PERANCANGAN DAN UJI KUALITAS *SOFTWARE* SIMULASI VIRTUAL "DIGICHIP" *PLATFORM* ANDROID PENUNJANG *MOBILE-VIRTUAL LEARNING* UNTUK KEJURUAN TEKNIK MEKATRONIKA

TITIH REJYASMITO HADI
14702251086

PROGRAM STUDI PENDIDIKAN TEKNOLOGI DAN KEJURUAN
PROGRAM PASCASARJANA
UNIVERSITAS NEGERI YOGYAKARTA
2018

ABSTRAK

TITIH REJYASMITO HADI : *Perancangan dan Uji Kualitas Software Simulasi Virtual "DigiChip" Platform Android Penunjang Mobile-Virtual Learning untuk Kejuruan Teknik Mekatronika.* **Tesis. Yogyakarta: Program Pascasarjana, Universitas Negeri Yogyakarta, 2018.**

Pembelajaran keilmuan teknik digital sangat memerlukan dukungan perangkat belajar berupa hardware memadai, secara operasional cukup membutuhkan biaya alat-bahan belajar yang tinggi. Siswa secara belajar mandiri kesulitan memiliki sarana belajar yang demikian. Siswa sangat terbatasi dengan permasalahan keterbatasan alat, bahan, waktu (jam belajar dikelas/lab/bengkel) dan sistem belajar yang harus bergilir/berkelompok. Penelitian ini bertujuan: (1) menghasilkan produk media belajar berupa *simulation software* yang bersifat *virtual learning* pada pembelajaran Gerbang Logika Dasar Teknik Digital, yang mampu beroperasi pada perangkat *mobile* sehingga memberikan kemudahan belajar berupa akses yang portabel, efektif, ekonomis dan operasional yang aman; (2) menguji kualitas produk *software* berdasarkan aspek *functional suitability, maintainability, portability* dan *usability* (ISO/IEC 25010).

Penelitian ini merupakan *research and development (R&D)* dengan proses perancangan dan pengujian *software* menggunakan metode *software development life cycle (SDLC)* jenis *v-model*, terdiri dari tahapan: (1) *requirement modeling*, (2) *architectural design*, (3) *component design*, (4) *code generation*, (5) *unit testing*, (6) *integration testing*, (7) *system testing*, (8) *acceptance testing*. Metode pengujian *unit* melalui teknik *white-box* dengan uji *basis-path, flowgraph, independent path* pada *software* "DigiChip". Pengujian *integration, system, acceptance* dengan teknik *black-box.* Aspek *functional suitability* diuji dengan kuesioner *run test* fitur, dan *test case*. Aspek *maintainability* diuji melalui pengukuran *maintainability index (MI), duplication source code, line of code (LoC), cyclomatic complexity (CC).* Aspek *portability* diuji dengan instalasi pada berbagai konfigurasi *hardware*, berbagai versi kernel OS Android. Uji materi dan media menggunakan kuesioner ahli materi dan ahli media. Aspek *usability* diuji menggunakan *USE Questionnaire* dan penghitungan *Cronbach's Alpha* melalui SPSS. Subyek penelitian ini siswa kelas XI Program Keahlian Teknik Mekatronika SMK-SMTI Yogyakarta.

Hasil penelitian menunjukkan: (1) perancangan *software* simulasi virtual "DigiChip" sebagai media *virtual learning* dan *mobile learning* telah dibuat untuk perangkat berbasis *mobile* komputer; (2) pengujian kualitas *software* berdasarkan ISO/IEC 25010, aspek *functional suitability* menunjukkan semua fitur beroperasi dengan baik (sangat layak), *maintainability* sebesar *(MI)* 84 (perawatan mudah), *portability* 100% dapat beroperasi pada semua *kernel OS* Android (sangat layak), dan *usability* 86,18% (sangat layak) dengan *Cronbach's Alpha* 0,841 (baik). Uji ahli media 90% (sangat layak), ahli materi 100% (sangat layak).

Kata Kunci: mekatronika, *mobile learning, virtual learning, virtual reality, simulation*

ABSTRACT

TITIH REJYASMITO HADI : *Designing and Testing the Quality of "DigiChip" Virtual Simulation Software of Android Platform for Mobile-Virtual Learning Supporting Vocational Mechatronics Engineering*. **Thesis. Yogyakarta: Postgraduate Program, Yogyakarta State University, 2018.**

Scientific learning of digital techniques really requires the support of learning devices in the form of adequate hardware, operationally enough to require the cost of high learning materials. Students who have independent learning difficulties have such learning tools. Students are very limited by the problems of limited tools, materials, time (study hours in class/lab/workshop) and learning systems that have to take turns. This study aims: (1) to produce learning media products in the form of simulation software that is virtual learning in the learning of Digital Engineering Basic Logic Gate, which is capable of operating on mobile devices so as to provide easy learning in the form of portable, effective, economical and operationally safe access; (2) testing the quality of software products based on functional aspects of suitability, maintainability, portability and usability (ISO/IEC 25010).

This research is research and development (R&D) with the process of designing and testing software using the software development life cycle (SDLC) type v-model, consisting of steps: (1) modeling requirements, (2) architectural design, (3) component design , (4) code generation, (5) unit testing, (6) integration testing, (7) system testing, (8) acceptance testing. Unit testing methods through the white-box technique with the base-path, flowgraph, independent path test on the "DigiChip" software. Testing for integration, system, acceptance with black-box techniques. Functional aspects of suitability were tested with a run test feature questionnaire, and a test case. Maintainability aspects are tested through measurements of maintainability index (MI), duplication of source code, line of code (LoC), cyclomatic complexity (CC). The portability aspect is tested by installing various hardware configurations, various kernel versions of the Android OS. Material and media tests use material expert questionnaires and media experts. The usability aspect was tested using the USE Questionnaire and Cronbach's Alpha calculation through SPSS. The subjects of this study were students of class XI of the SMTI Yogyakarta Mechatronics Expertise Program.

The results of the study show: (1) the design of "DigiChip" virtual simulation software as a medium of virtual learning and mobile learning has been made for mobile computer-based devices; (2) software quality testing based on ISO/IEC 25010, functional aspects suitability shows all features operating properly (very feasible), maintainability of (MI) 84 (easy maintenance), portability 100% can operate on all Android OS kernels (very feasible), and usability 86.18% (very feasible) with Cronbach's Alpha 0.841 (good). Test media experts 90% (very decent), material experts 100% (very feasible).

Keywords: *simulation, virtual reality, virtual learning, mobile learning, mechatronics*

PERNYATAAN KEASLIAN KARYA

Saya yang bertanda tangan dibawah ini :

Nama Mahasiswa	:	**Titih Rejyasmito Hadi**
Nomor Mahasiswa	:	**14702251086**
Program Studi	:	**Pendidikan Teknologi dan Kejuruan**

Menyatakan bahwa tesis ini merupakan hasil karya saya sendiri dan belum pernah diajukan untuk memperoleh gelar magister di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya dalam tesis ini tidak terdapat karya yang pernah ditulis atau diterbitkan oleh orang lain kecuali yang secara tertulis disadur dalam naskah dan disebut dalam daftar pustaka.

Yogyakarta, 28 Desember 2018

Pembuat pernyataan

Titih Rejyasmito Hadi
NIM. 14702251086

LEMBAR PENGESAHAN

PERANCANGAN DAN UJI KUALITAS *SOFTWARE* SIMULASI VIRTUAL "DIGICHIP" *PLATFORM* ANDROID PENUNJANG *MOBILE-VIRTUAL LEARNING* UNTUK KEJURUAN TEKNIK MEKATRONIKA

TITIH REJYASMITO HADI
NIM 14702251086

Dipertahankan di depan Tim Penguji Tesis
Program Pascasarjana Universitas Negeri Yogyakarta
Tanggal: 7 Januari 2019

TIM PENGUJI

Dr. Mochamad Bruri Triyono, M.Pd. (Ketua/Penguji)	.....	18/01/2019
Dr. Ratna Wardani, M.T. (Sekretaris/Penguji)	.....	22/01/2019
Dr. Eko Marpanaji, M.T. (Pembimbing/Penguji)	.....	17/01/2019
Dr. Fatchul Arifin, M.T. (Penguji Utama)	.....	29/01/2019

Yogyakarta, 28-2-2019
Program Pascasarjana
Universitas Negeri Yogyakarta
Direktur,

Prof. Dr. Marsigit, M.A.
NIP 19570719 198303 1 004

KATA PENGANTAR

Alhamdulillahirobbil'alamin,

Segala puji hanya milik Allah SWT, Tuhan pemilik semesta alam beserta seluruh isinya. Berkat limpahan rahmat dan karunia-Nya, saya dapat menyelesaikan kewajiban pembuatan tesis ini beserta seluruh rangkaian kegiatan di dalamnya dengan baik. Tesis dengan judul "*Perancangan dan Uji Kualitas Software Simulasi Virtual "DigiChip" Platform Android Penunjang Mobile-Virtual Learning untuk Kejuruan Teknik Mekatronika*", dapat disusun sesuai dengan harapan. Semoga menjadikan kepuasan bagi semua pihak dan saya sendiri tentunya.

Tugas Akhir Tesis ini tidak lepas dari bantuan dan dukungan semua pihak. Pada kesempatan ini, dengan segala kerendahan hati sebagai ungkapan rasa syukur atas segala bantuan yang telah diberikan perkenankan saya menyampaikan ucapan terima kasih kepada :

1. Prof. Dr. Sutrisna Wibawa, M.Pd., Rektor Universitas Negeri Yogyakarta.
2. Prof. Dr. Marsigit, M.A., selaku Direktur Program Pascasarjana Universitas Negeri Yogyakarta.
3. Dr. Mochamad Bruri Triyono, M.Pd. selaku Ketua Program Studi Pendidikan Teknologi dan Kejuruan - Program Pascasarjana Universitas Negeri Yogyakarta.
4. Dr. Eko Marpanaji, M.T., selaku Dosen Pembimbing Tesis.

5. Prof. Herman Dwi Surjono, Ph.D., Dr. Fatchul Arifin, M.T., Dr. Ratna Wardani, M.T., selaku Validator instrument penelitian dan Penguji Tesis.
6. Dr. Samsul Hadi, M.Pd., M.T., Muhammad Agung Wibowo, S.Si., selaku Penguji Ahli Media dan Penguji Ahli Materi.
7. Imam Suhudi, M.Ag., Wasilah, M.Ag., Fajar Swasono Hadi, M.H., selaku orang tua, adik, dan sponsor bantuan pendanaan riset :) .
8. Semua pihak, yang tidak dapat disebutkan semua di lembaran yang terbatas ini, atas segala bantuan dan perhatian, hingga terwujudnya penyusunan hasil riset ini, saya mengucapkan banyak terima kasih.

Akhirnya, semoga segala bantuan yang telah diberikan semua pihak di atas menjadi amalan yang bermanfaat dan mendapatkan keberkahan dari Allah SWT dan semoga Tugas Akhir Tesis ini tidak berhenti hanya di rak perpustakaan melainkan menjadi informasi dan inspirasi yang bermanfaat bagi pembaca dan semua pihak yang membutuhkannya.

Yogyakarta, 20 Desember 2018

Penyusun,

Titih Rejyasmito Hadi

NIM. 14702251086

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

A. Latar Belakang

Perkembangan ilmu pengetahuan dan teknologi (IPTEK) yang semakin kompleks saat ini, teknologi memiliki peranan penting hampir disemua aspek kebutuhan hidup manusia. Demikian pula perkembangan dunia teknologi digital, teknologi *mobile,* dan perkembangan teknologi komputer yang kini telah dikembangkan dalam bentuk *handheld.* Perkembangan teknologi perangkat komputer dalam bentuk yang semakin praktis, membuat berbagai fitur komputer dapat dinikmati hanya dalam genggaman tangan. Permasalahan di sini adalah seiring dengan perkembangan teknologi *handheld* ke arah yang lebih *expert*, tentu perlu ada harmonisasi dan adaptasi dalam menguasai dan mengoperasikan.

Kemajuan sebuah bangsa terletak pada seberapa tingkat kemajuan dari ilmu pengetahuan dan teknologi dibandingkan dengan bangsa lain. Masyarakat pengguna teknologi tentu harus memiliki kemampuan untuk menggunakan, terlebih ketika memilih sebagai masyarakat pencipta teknologi. Tentu dibutuhkan kemampuan lebih lagi dalam penguasaan ilmu pengetahuan dan teknologi. Hal ini tentu akan mengarah pada tuntutan kualitas sumber daya manusia (SDM) dan berbagai sumber daya pendukung yang lain.

Teknologi dimanfaatkan sebagai alat belajar dan media belajar, dalam dunia pendidikan akan memberikan nuansa dan citra rasa baru. Secara otomatis tentu harus didukung dengan perilaku bijak dalam menggunakan teknologi *mobile*/*handheld* ke arah yang produktif, edukatif dan bukan malah sebaliknya,

yaitu ke arah yang kontraproduktif. Teknologi sejatinya dilahirkan untuk memberikan sesuatu yang bermanfaat dan memudahkan segala kebutuhan hidup umat manusia. Perilaku bijak dalam menggunakan teknologi haruslah menjadi prioritas utama dalam melahirkan generasi dan sumber daya manusia (SDM) yang memiliki tingkat kualitas tinggi dalam suatu bangsa.

Pendidik dan peserta didik selama ini banyak memiliki pandangan tentang perangkat *mobile*/*handheld* hanya dapat digunakan sebagai sarana hiburan dan komunikasi saja. Padahal penggunaan teknologi perangkat *mobile* ke arah yang lebih edukatif sangatlah perlu untuk dimunculkan. Salah satunya sebagai sarana media belajar berbasis simulasi, atau biasa disebut teknologi media simulator. Tingkat kepraktisan, efisien, ekonomis dan aman, tentu menjadi poin lebih pada media belajar berbasis simulasi. Hal ini diharapkan akan sangat membantu mengatasi motivasi belajar siswa yang rendah ketika dihadapkan pada permasalahan keterbatasan alat, bahan, waktu (jam belajar dikelas/lab/bengkel) dan sistem belajar yang harus bergilir/berkelompok.

Perangkat *mobile* yang merupakan sebuah perangkat *Personal Computer (PC)* tentu dimiliki secara personal oleh masing-masing siswa. Aktivitas belajar secara personal dan mandiri tanpa harus tergantung pada jam praktikum, alat, serta bahan praktikum yang disediakan sekolah bisa diatasi. Siswa akan memiliki porsi jam belajar lebih dan dapat belajar kapan pun dimana pun. Demikian pula dalam permasalahan motivasi siswa yang rendah dalam membaca buku dan pengadaan sumber belajar berupa buku. Hal ini bisa diatasi pula dengan menggunaan perangkat *mobile* sebagai sarana membaca yang dikemas dalam

bentuk *e-book, digital book* maupun hasil foto buku, sehingga akan lebih praktis, ekonomis dan memberikan daya tarik lebih dalam membaca. Perangkat *mobile* dalam kenyataannya selalu dalam saku dan genggaman para penggunanya, sehingga sebagai siswa dapat melakukan aktivitas belajar lebih mudah, kapan pun, dimana pun dalam membaca.

Penelitian ini diharapkan dapat memberikan edukasi masyarakat secara luas dan kalangan pengguna perangkat *handheld* sebagai media yang mampu digunakan ke arah yang lebih produktif, dalam hal ini adalah sebagai media untuk belajar. Para pendidik dan peserta didik sering menggunakan peralatan seperti; ponsel, *smartphone, tablet-PC,* yang akhir-akhir ini telah menjadi perangkat yang begitu akrab dalam kehidupan sehari-hari. Perangkat tersebut merupakan pengembangan dari perangkat teknologi komputer yang telah menjadi dalam bentuk teknologi komputer genggam *(handheld),* dan teknologi perangkat *mobile.* Teknologi komputer mengalami berbagai pengembangan tentu bertujuan untuk keberadaan teknologi komputer agar dapat dinikmati sepraktis dan semudah mungkin oleh semua orang. Perkembangan teknologi komputer hingga seperti sekarang ini, melahirkan inovasi cara belajar baru yang berbasis pada perangkat *mobile* ini *(Mobile Learning).*

Aktivitas belajar menggunakan perangkat *mobile* dapat digunakan sebagai media alternatif dalam permasalahan proses pembelajaran teknik digital, berupa permasalahan ketersediaan peralatan dan bahan praktikum. Peralatan yang utama tentunya sebuah *I/O Digital Logic Board,* yaitu sebuah *hardware* elektronik yang berfungi untuk memproses chip *logic gate* menjadi keluaran *(output)* berupa

tampilan visual yang bisa dipahami manusia. Biasanya berupa lampu indikator LED maupun dalam bentuk *sevensegment*. Peralatan *I/O Digital Logic Board*, merupakan peralatan utama praktikum teknik digital yang bisa dikatakan tidak murah. Saat pembelajaran teknik digital idealnya satu pebelajar adalah satu peralatan *I/O Digital Logic Board*.

Ketersediaan bahan praktikum juga menjadi permasalahan tersendiri saat proses pembelajaran teknik digital. Bahan praktikum yang berupa chip *logic gate* ini bersifat *non-reusable*, atau ketika terjadi kerusakan sulit untuk diperbaiki. Kerusakan pada sebuah chip *logic gate* hanya dapat diatasi dengan mengganti sebuah chip *logic gate* baru. Karena sifat chip *logic gate* dalam pemakaiannya saat pembelajaran harus beroperasi *on-off* terus menerus dalam intensitas yang tinggi. Padahal saat pembelajaran dimungkinkan setiap peserta didik menghabiskan bahan praktik berupa chip *logic gate* dalam jumlah lebih dari 5 atau bahkan lebih.

Media pembelajaran berbasis simulasi akan lebih memberikan nilai ekonomis, praktis, portabel dan lebih aman dioperasikan. Demikian pula dalam penelitian pengembangan ini dipilih ranah pembelajaran berbasis *mobile* karena peralatan teknologi *mobile* berupa *smartphone*, *tablet-PC,* merupakan peralatan personal para peserta didik yang mayoritas dari mereka menggunakan dan memiliki. Nilai efektif pembelajaran bahwa satu peserta didik memiliki satu media belajar tentu akan menjadi terwujud. Penelitian pengembangan ini merancang dan menguji kualitas sebuah media pembelajaran yang memiliki kemampuan selayaknya media pembelajaran teknik digital dalam bentuk fisik

aslinya, menjadi dalam bentuk simulasi virtual. Tentu *software* simulasi tersebut dirancang dan diuji untuk mampu beroperasi pada teknologi perangkat *mobile.*

Konsep inovasi pengembangan media belajar ini untuk membantu para peserta didik dalam belajar mengoperasikan teknik digital kapan saja dan dimana saja akan terwujud. Faktor kelebihan lain yang ditawarkan dalam produk media belajar simulasi ini adalah karakter produk yang bersifat *off-line*, sehingga siswa lebih termotivasi belajar karena tanpa harus bergantung pada biaya dan koneksi internet. Penelitian pengembangan ini sebagaimana tujuan utamanya adalah untuk memberikan inovasi sebuah konversi aktivitas belajar teknik digital yang membutuhkan biaya tinggi, menjadi tanpa biaya sama sekali, serta kemudahan operasional yang praktis lainnya.

B. Identifikasi Masalah

Berdasar pada latar belakang yang telah dijelaskan pada bahasan sebelumnya dapat diidentifikasi berbagai permasalahan menjadi uraian singkat sebagai berikut:

1. Pembelajaran pada keilmuan teknik digital sangat memerlukan dukungan perangkat belajar berupa hardware memadai, secara operasional cukup membutuhkan biaya alat-bahan belajar yang tinggi.
2. Siswa secara belajar mandiri diluar sekolah atau jam pelajaran, kesulitan memiliki sarana belajar gerbang logika dasar teknik digital. Siswa sangat terbatasi dengan permasalahan keterbatasan alat, bahan, waktu (jam belajar dikelas/lab/bengkel) dan sistem belajar yang harus bergilir/berkelompok.

3. Pendidik dan peserta didik memerlukan adanya media pembelajaran yang dapat mengatasi permasalahan belajar pada materi pembelajaran gerbang logika dasar teknik digital, serta pengenalan tentang inovasi dalam penggunanaan teknologi perangkat *mobile* komputer ke arah yang lebih edukatif dan produktif.
4. Media belajar dan praktikum gerbang logika dasar teknik digital dalam bentuk fisik *(hardware)* memiliki banyak kelemahan, mulai dari pengadaan peralatan yang mahal, biaya operasional bahan praktikum, resiko kurang aman dalam praktik merangkai komponen elektronik maupun kelistrikan lain, dan terbatasnya akses peserta didik ketika ingin menggunakannya secara belajar mandiri diluar jam sekolah.

C. Pembatasan Masalah

Pembuatan produk perangkat lunak *(software)* hanya terbatas pada aplikasi simulasi virtual untuk materi pembelajaran gerbang logika dasar teknik digital. Produk perangkat lunak dalam penelitian ini diberi nama "DigiChip", sebab hanya menyediakan fitur simulasi untuk IC *chip* gerbang logika dasar teknik digital. Desain produk perangkat lunak "DigiChip" berbentuk *software* simulasi yang menampilkan modul *logic gate* fisik dalam bentuk modul virtual. Pemilihan *platform OS (Operating System)* hanya dibatasi pada sistem operasi Android, karena disasarkan untuk segmen pengguna perangkat *mobile (handheld PC)* yang memiliki jumlah dominan. Produk perangkat lunak "DigiChip" tidak dirancang untuk sistem operasi *iOS*, *Windows Phone*, *Tizen*, *Blackberry OS*, dll. Proses pengujian kualitas produk perangkat lunak simulasi "DigiChip" mengacu pada

standar ISO/IEC 25010 meliputi kualitas *functional suitability, maintainability, portability, usability*. Penelitian dibatasi hanya dilakukan pada materi pembelajaran gerbang logika dasar teknik digital, sesuai dengan standar Kompetensi Dasar (KD) pada Program Keahlian Teknik Mekatronika di Sekolah Menengah Kejuruan Teknik Industri (SMTI) Yogyakarta.

D. Rumusan Masalah

Berdasarkan uraian pada pembatasan masalah dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana rancangan perangkat lunak *(software)* simulasi "DigiChip" sebagai media belajar virtual Teknik Digital pada *platform* Android?
2. Bagaimanakah pengujian kualitas dan analisis hasil produk *software* simulasi virtual “DigiChip” yang dirancang sesuai standar ISO/IEC 25010 meliputi kualitas *functional suitability, maintainability, portability,* dan *usability*?

E. Tujuan Penelitian

Tujuan yang diharapkan dalam penelitian ini:

1. Merancang dan menghasilkan sebuah produk media belajar berupa perangkat lunak simulasi bersifat *virtual learning* pada pembelajaran Gerbang Logika Dasar *(Logic Gate)* ilmu Teknik Digital, yang mampu beroperasi pada perangkat *mobile* sehingga memberikan kemudahan belajar berupa *mobile learning*, dengan akses yang portabel, efektif dan ekonomis.

2. Menguji kualitas *software* "DigiChip" dan analisis hasil, berdasarkan standar ISO/IEC 25010 yang meliputi aspek *functional suitability*, *maintainability*, *portability* dan *usability*.

F. Manfaat Penelitian

Hasil penelitian diharapkan bermanfaat bagi:

1. Guru dan Siswa

   Dapat digunakan sebagai bahan masukan khususnya bagi guru, tentang penggunaan *mobile learning* dan *virtual learning* sebagai inovasi dan solusi permasalahan sarana prasarana pembelajaran siswa. Bagi siswa, sebagai saran inspirasi untuk meningkatkan kualitas belajar serta motivasi dalam belajar.

2. Peneliti

   Menambah wawasan ilmu pengetahuan dan riset, serta sebagai media untuk menerapkan keilmuan yang telah dimiliki. Sebagai sumber referensi bagi peneliti berikutnya dalam melakukan penelitian serupa dimasa akan datang.

BAB II
KAJIAN PUSTAKA

A. Kajian Teori

1. Media Pembelajaran

Sistem-sistem komputer dapat menyampaikan pembelajaran secara langsung kepada para siswa melalui cara berinteraksi dengan mata pelajaran yang diprogramkan ke dalam sistem, dan inilah yang disebut pengajaran dengan bantuan komputer. Kegiatan pembelajaran dengan komputer, atau yang lebih dikenal dengan *Computer-Based Instruction (CBI)*, adalah istilah umum yang digunakan untuk segala kegiatan belajar yang menggunakan komputer baik sebagian maupun keseluruhan. Pembelajaran berbasis komputer (CBI) merupakan sebuah konsep pembelajaran baru yang hingga saat ini banyak sekali ragam desain dan implementasinya dalam dunia pendidikan dan pembelajaran (Indriana, 2011:107).

Model pembelajaran CBI saat ini telah berkembang menjadi berbagai ragam model, seperti: *Computer-Assisted Instruction (CAI)* dan mengalami perbaikan menjadi *Intelligent Computer Assisted Instruction (ICAI)*. Dengan karakteristik fungsi yang berbeda muncul pula *Computer Assisted Learning (CAL)*, *Computer Based Learning (CBL)*, *Computer Assisted Personalized Assignment (CAPA)*, dan *Intelligent Tutoring System (ITS)* (Indriana, 2011:103). Berbagai bentuk program tersebut dapat bersifat tutorial, latihan dan perulangan pada materi pelajaran yang telah dipelajari sebelumnya, bersifat permainan dan simulasi.

Menurut jenisnya, sumber pembelajaran dapat diklasifikasikan sebagai berikut: (1) Bersumber dari manusia (pengajar, teman sebaya, pakar, produser media); (2) Sumber dari bahan cetak (buku, majalah, jurnal, ensiklopedia, surat kabar, katalog, kamus, buku teks, dsj); (3) Media audio dan visual (televisi, radio, OHP, LCD, tape recorder, video, CD/VCD/DVD player, film, dsj); (4) Media komputer (disket, CD, VCD, DVD, hardware & software komputer, internet); (Sanaky, 2013:21). Teknologi media pembelajaran berbasis komputer rnerupakan cara-cara mernproduksi dan rnenyampaikan bahan dengan rnenggunakan perangkat yang bersurnber pada *mikroprosesor*. Teknologi berbasis komputer dibedakan dari teknologi lain karena menyirnpan informasi secara elektronis dalam bentuk digital, bukan sebagai bahan cetak atau visual. Media pembelajaran berbasis komputer penekanannya terletak pada upaya yang berkesinambungan untuk memaksimalkan aktivitas belajar dan rnengajar sebagai interaksi kognitif antara siswa, materi pelajaran, dan instruktur (dalam hal ini komputer yang telah diprogram).

Media pembelajaran memiliki pengertian lain menurut Indriana, bahwa yang dimaksudkan dengan media pembelajaran adalah semua bahan dan alat fisik yang mungkin digunakan untuk mengimplementasikan pengajaran dan memfasilitasi prestasi siswa terhadap sasaran atau tujuan pembelajaran. Media pembelajaran mencakup bahan-bahan tradisional seperti papan tulis, buku pegangan, bagan, slide, OHP/OHT, objek-objek nyata, dan rekaman video atau film. Selain itu dalam bentuk berupa bahan-bahan dan

beberapa metode mutakhir seperti komputer, DVD, CD-Room, internet, dan penggunaan fasilitas konferensi video secara interaktif (Indriana, 2011:16).

Media Pembelajaran adalah sebuah alat yang berfungsi dan dapat digunakan untuk menyampaikan pesan pembelajaran, sebagai perantara dalam proses pembelajaran untuk mempertinggi efektifitas dan efisiensi dalam mencapai tujuan pengajaran. Pembelajaran merupakan proses komunikasi antara pembelajar, pengajar, dan bahan ajar. Komunikasi tidak akan berjalan dengan baik tanpa bantuan sarana untuk menyampaikan pesan/materi (Sanaky, 2013:3). Bentuk-bentuk stimulus dapat dipergunakan sebagai media, diantaranya adalah hubungan atau interaksi manusia, realitas, gambar bergerak atau tidak, tulisan dan suara yang direkam. Dengan kelima bentuk stimulus ini, akan membantu peserta didik mempelajarai bahan pelajaran. Substansi dan inti dari media pembelajaran adalah: (1) bentuk saluran, yang digunakan untuk menyalurkan pesan, informasi dan bahan pelajaran kepada penerima pesan; (2) berbagai jenis komponen dalam lingkungan pembelajar yang dapat merangsang pembelajar untuk belajar; (3) bentuk alat fisik yang dapat menyajikan pesan serta merangsang pembelajar untuk belajar; dan (4) bentuk-bentuk komunikasi dan metode yang dapat merangsang pembelajar untuk belajar, baik cetakmaupun audio, visual, dan audio-visual.

Media pembelajaran hingga saat ini berkembang menjadi berbagai bentuk ragam dan tingkat teknologi yang digunakan. Perkembangan teknologi media pembelajaran dalam pendidikan tinggi maupun pada aktivitas

pengembangan sumber daya manusia yang lain saat ini mengacu pada penggunaan media pembelajaran yang praktis dan fleksibel. Dampak perkembangan teknologi akan memberikan perubahan media pembelajaran dari beberapa ragam menjadi berbagai varian media pembelajaran. Hal tersebut selaras dengan apa yang telah dijelaskan pada kutipan berikut ini;

> *As higher education and human resource development adopt flexible and online approaches to learning, the role of technologies changes from one of being an adjunct to the process of learning. Their new role is to be the mediators of communications between learners and between learners and teachers and to provide learners with access to the resources they need.* (Caladine, 2008:40).

Dampak perkembangan teknologi memang akan memberikan budaya baru dalam belajar, serta memberikan ragam yang lebih pada berbagai media pembelajaran. Perkembangan media pembelajaran yang semakin beraneka ragam tentunya tidak akan merubah filosofi dari sebuah pembelajaran yang merupakan sebuah aktivitas transfer ilmu pengetahuan dari pendidik *(teachers)* kepada peserta didik *(learners)*. Kita memahami bahwa teknologi baru dan keragaman teknologi dalam media pembelajaran hanya memiliki satu tujuan yang satu, yaitu sebagai media komunikasi dalam proses transfer ilmu pengetahuan antar peserta didik dan antara pendidik ke peserta didik. Seorang pendidik harus mampu menggunakan media yang terbaik untuk memfasilitasi pembelajaran atau meningkatkan pemahaman siswa terhadap bahan pelajaran. Proses komunikasi untuk memfasilitasi pembelajaran bisa menjadi sebuah proses yang menantang, yang sering kali membutuhkan usaha-usaha kreatif dan inovatif untuk mencapai sebuah ragam tujuan-tujuan pengajaran yang implisit.

Seorang pendidik secara tidak langsung dituntut mampu beradaptasi dalam budaya dan perilaku belajar baru dari siswa, yang tidak lain adalah produk baru dan cara belajar baru dari dampak perubahan era. Tempat pendidikan dan para insan pendidik tentu memiliki fungsi mengakomodasi berbagai cara belajar dan sekaligus sebagai filter dalam kualitas belajar yang terbaik. Proses akomodasi yang kurang siap tentu akan menjadi kendala dalam keberlangsungan proses pembelajaran dengan mediasi berbasis teknologi. Kasus demikian ini sejalan dengan apa yang banyak dialami oleh para pendidik, seperti apa yang ada pada kutipan berikut ini;

> *While these teachers may have been "tech savvy" they may have also been at a loss to determine what the curriculum would be, how to deal with technology problems, and how to organize and assess student projects. Many of the experienced teachers I have known have been reluctant to disrupt the smooth and efficient flow of their teaching to integrate technology or a new methodology into their practice*. (Martinez, 2011:13).

Perkembangan teknologi tentu akan menjadi sarana yang menunjang proses mediasi antara pendidik dan peserta didik dalam proses pembelajaran, apabila hal ini didesain secara tepat guna. Aktivitas penyalahgunaan teknologi dalam proses pembelajaran tentu bertentangan dengan, tujuan utama teknologi media pembelajaran sebagai sarana pendukung proses transfer ilmu pengetahuan menjadi lebih baik.

Media pengajaran itu selalu terdiri atas dua unsur, yaitu unsur peralatan atau perangkat keras *(hardware)* dan unsur materi yang dibawanya *(software)*. Dengan demikian, media pengajaran memerlukan peralatan untuk menyajikan materi. Namun, yang terpenting bukanlah media, tetapi materi

atau informasi belajar yang dibawakan oleh media tersebut (Indriana, 2011:21). Tujuan utama media pembelajaran adalah untuk memadukan aspek afektif, kognitif, dan psikomotor, yang sangat penting dalam proses pembelajaran. Tiga aspek tersebut menjadi indikator keberhasilan proses pembelajaran yang sesuai dengan apa yang menjadi harapan. Pada ranah kognitif, kemampuan yang diharapkan bisa didapat melalui media pembelajaran adalah kemampuan yang bersifat intelekfual atau kognitif.

Kemampuan yang bersifat intelektual ini terdiri atas pengetahuan *(knowledge)*, pemahaman *(comprehension)*, penerapan *(application)*, penguraian/analisis *(analysis)*, sintesis *(syntesis)*, dan penilaian. Sedangkan pada ranah afektif, kemampuan yang dituju dari penggunaan media adalah berkaitan dengan rasa, sikap, dan tingkah laku. Ranah afektif ini terdiri atas penerimaan *(receiving)*, tanggapan *(recording)*, penghargaan *(valuing)*, pengaturan *(organization)*, dan karakterisasi *(characterization)*. Pada ranah psikomotorik, kemampuan yang ditekankan melalui media pembelajaran adalah kemampuan yang bersifat jasmaniah atau fisik. Ranah psikomotorik ini terdiri atas persepsi *(perception)*, mekanisme *(mechanism)* dan kemampuan untuk menyelesaikan.

Pembelajaran merupakan wadah dari materi pembelajaran yang ingin disampaikan oleh pendidik kepada para peserta didik, yang bertujuan untuk mencapai proses pembelajaran yang efektif, efisien dan sesuai dengan apa yang menjadi harapan. Jika seorang pendidik mampu memanfaatkan dan menggunakan media pembelajaran secara maksimal, maka peserta didik akan

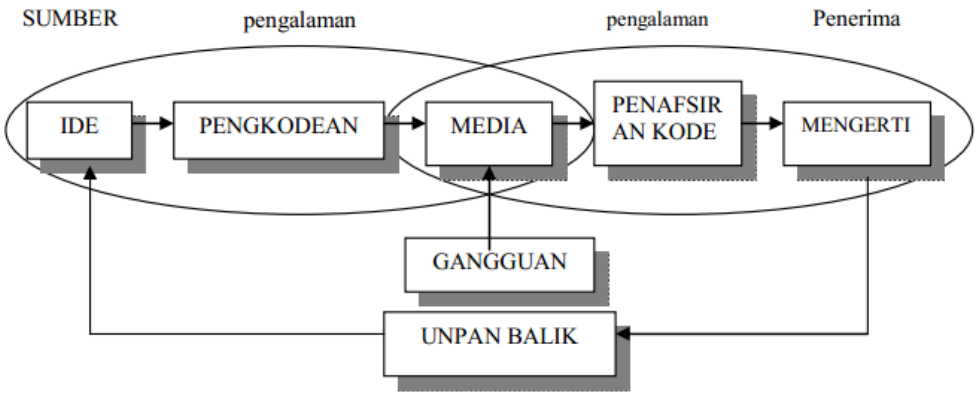
mampu rnenyerap materi yang disampaikan, belajar lebih banyak, menyerap sesuatu yang dipelajari dengan lebih baik, serta ketrampilan sesuai dengan tujuan proses belajar dan mengajar. Setiap guru/pendidik harus dapat memiliki pengetahuan dan pemahaman yang baik tentang media pendidikan/pengajaran. Hal ini untuk terciptanya kondisi dan kualitas belajar yang cukup baik pula.

> *Some of the knowledge and expertise are embedded in the technology and media, and some of it is available in the social lives of the students. Students bring their own knowledge of media and culture to the learning environment and collectively they have a great deal of access to the different types of media and technology...* (Martinez, 2011:66).

Beberapa ilmu pengetahuan dan keahlian tertentu memang tertanam dalam perangkat teknologi dan media pembelajaran, dan beberapa di antaranya tersedia dalam kehidupan sosial para siswa. Siswa membawa pengetahuan media pembelajaran dan budaya mereka sendiri ke lingkungan belajar dan secara kolektif mereka memiliki banyak akses terhadap berbagai jenis media pembelajaran dan perangkat teknologi. Kondisi dari perkembangan cara belajar yang demikian tentu sangat memerlukan akomodasi dari tempat belajar maupun tenaga pengajar yang sekaligus sebagai filter dan pembimbing, agar aktivitas pembelajaran berlangsung sesuai tujuan.

Media pembelajaran memiliki posisi penting dalam sebuah pembelajaran. Proses pembelajaran merupakan proses komunikasi dan berlangsung dalam suatu sistem, tanpa media untuk berkomunikasi tentu tidak akan terjadi proses komunikasi dan pembelajaran tidak mampu berlangsung secara optimal. Media pembelajaran adalah komponen integral

dari sistem pembelajaran. Menurut Daryanto (2013:7), posisi media dalam pembelajaran dapat dilihat pada gambar berikut;



Gambar 1. Posisi Media dalam Pembelajaran
(sumber: Buku *"Media Pembelajaran"* Daryanto, 2013, hal. 7)

Media pembelajaran berbasis modul elektronik memiliki sifat; (1) sebagai media yang menunjukkan, (2) sebagai pendukung untuk praktikum, (3) sebagai media belajar interaktif dan bersifat individu (Munadi, 2010:48). Tujuan media pembelajaran sebagai alat bantu pembelajaran adalah diantaranya sebagai media untuk: (a) mempermudah proses pembelajaran di kelas; (b) meningkatkan efisiensi proses pembelajaran; (c) menjaga relevansi antara materi pelajaran dengan tujuan belajar; (d) membantu konsentrasi pembelajar dalam proses pembelajaran (Sanaky, 2013:5).

Manfaat media pembelajaran baik secara umum maupun khusus sebagai alat bantu pembelajaaran bagi pengajar dan pembelajar diantaranya adalah: (1) pengajaran lebih menarik perhatian pembelajar sehingga dapat menumbuhkan motivasi belajar; (2) bahan pengajaran akan lebih jelas

maknanya, sehingga dapat lebih difahami pembelajar serta memungkinkan pembelajar menguasai tujuan pengajaran dengan baik; (3) metode pembelajaran bervariasi, tidak semata-mata hanya komunikasi verbal melalui penuturan kata-kata lisan pengajar, pembelajar tidak bosan, dan pengajar tidak kehabisan tenaga, (4) pembelajar lebih banyak melakukan kegiatan belajar, sebab tidak hanya mendengarkan penjelasan dari pengajar saja, tetapi juga aktivitas lain yang dilakukan seperti: mengamati, melakukan, mendemonstrasikan, dan lain-lain (Sanaky, 2013:5). Kesimpulan dari uraian diatas adalah suatu media pembelajaran harus bersifat menumbuhkan motivaasi belajar dari peserta didik. Oleh karena itu media pembelajaran sebagai alat bantu belajar harus memiliki nilai-nilai daya tarik, memudahkan dalam mahami suatu ilmu, kreatif dan tidak membosankan.

Media pembelajaran menurut Sanaky mampu memberikan manfaat bagi pengajar antara lain; (1) memberikan pedoman, arah untuk mencapai tujuan pembelajaran, (2) menjelaskan struktur dan urutan pengajaran secara baik, (3) memberikan kerangka sistematis mengajar secara baik, (4) memudahkan kendali pengajar terhadap materi pelajaran, (5) membantu kecermatan, ketelitian dalam penyajian materi pelajaran, (6) membangkitkan rasa percaya diri seorang pengajar, (7) meningkatkan kualitas pengajaran, (8) memberikan dan meningkatkan variasi belajar, (9) menyajikan inti informasi, pokok-pokok secara sistematik, sehingga memudahkan penyampaian, (10) menciptakan kondisi dan situasi belajar yang menyenagkan dan tanpa tekanan.

Manfaat media pembelajaran bagi pembelajar diataranya adalah; (1) meningkatkan motivasi belajar pembelajar, (2) memberikan dan meningkatkan variasi belajar bagi pembelajar, (3) memudahkan pembelajar untuk belajar, (4) merangsang pembelajar untuk berfikir dan beranalisis, (5) pembelajaran dalam kondisi dan situasi belajar yang menyenagkan dan tanpa tekanan, (6) pembelajar dapat memahami materi pelajaran secara sistematis yang disajikan. Kesimpulan dari uraian tersebut adalah sebuah inovasi media pembelajaran bermanfaat untuk meningkatkan kualitas pembelajaran, memberikan kemudahan dalam menyampaikan konten materi tertentu sehingga mudah diserap peserta didik dan merangsang motivasi belajar peserta didik.

Pertimbangan pemilihan media pembelajaran yang akan digunakan tentunya harus berorientasi pada berbagai poin berikut ini, menurut (Sanaky, 2013:6), diantaranya adalah: (1) tujuan pengajaran; (2) bahan pelajaran; (3) metode penyampaian materi ajar; (4) ketersediaan alat yang dibutuhkan; (5) pribadi pengajar; (6) kondisi minat dan kemampuan pembelajar; (7) situasi pengajaran yang sedang berlangsung. Kesimpulannya adalah dalam pemilihan dan pembuatan media pembelajaran harus meliputi aspek-aspek yang utama, yaitu: tujuan belajar, kemampuan belajar, situasi belajar dan pribadi pengajar, dengan kata lain sebuah media pembelajaran tidak hanya dirancang hanya berdasarkan pertimbangan menarik dan berteknologi tinggi saja. Secara fungsi, media pembelajaran digunakan sebagai upaya untuk merangsang proses pembelajaran dengan cara: (1) menghadirkan objek

sebenarnya dan objek yang langka; (2) membuat duplikasi dari objek yang sebenarnya; (3) membuat konsep abstrak ke konsep kongkret; (4) memberi kesamaan persepsi; (5) mengatasi hambatan waktu, tempat, jumlah, dan jarak; (6) menyajikan ulang informasi secara konsisten; (7) memberi suasana belajar yang menyenangkan, tidak tertekan, santai, dan menarik, sehingga dapat mencapai tujuan pembelajaran.

Kesimpulannya bahwa berdasarkan berbagai pengertian mengenai media pembelajaran yang telah dijelaskan sebelumnya, bisa dipahami bahwa media merupakan alat bantu yang sangat bermanfaat bagi para siswa dan pendidik dalam proses belajar dan mengajar. Keberadaan media pengajaran, menjadikan peran guru semakin luas. Para anak didik akan terbantu untuk belajar dengan lebih baik, serta terangsang untuk memahami subjek yang sedangkan diajarkan dalam bentuk komunikasi penyampaian materi yang lebih efektif dan efisien. Media pengajaran merupakan salah satu alat komunikasi dalam proses pembelajaran. Dikatakan demikian karena di dalam media pengajaran terdapat proses penyampaian materi dari pendidik kepada anak didik. Sedangkan materi yang dikirimkan, biasanya, berupa informasi atau keterangan dari pengirim materi. Materi tersebut adakalanya disampaikan dalam bentuk sandi-sandi atau larnbang-lambang, seperti kata-kata, bunyi, gambar, dan lain sebagainya. Melalui saluran seperti radio, televisi, OHP, film, materi diterima oleh penerima materi melalui indra untuk diolah, sehingga materi yang disampaikan dapat diterima dan dipahami oleh penerima materi.

2. Pembelajaran Simulasi

Simulasi merupakan suatu teknik meniru operasi-operasi atau proses-proses yang terjadi dalam suatu sistem dengan bantuan perangkat komputer dan dilandasi oleh beberapa asumsi tertentu sehingga sistem tersebut bisa dipelajari secara ilmiah (Mahendra, 2014:169). Simulasi dapat diartikan sebagai meniru suatu sistem nyata yang kompleks dengan penuh dengan sifat probabilistik, tanpa harus mengalami keadaan yang sesungguhnya (Rosadi, 2012:36). Simulasi dalam pembelajaran adalah suatu metode yang digunakan untuk melakukan pengamatan/penelitian dan analisis dari perilaku dan kemampuan suatu sistem nyata atau sistem secara teoritis (Purwoko, 2011:357). *Software* simulasi melibatkan peserta didik dalam satu realitas virtual dan terkomputerisasi. Peserta didik belajar menunjukkan fungsi dan perannya dalam bersimulasi. Lingkungan seperti ini, peserta didik mampu belajar dari dalam diri mereka (Indriana, 2011:103).

Simulasi dapat dimanfaatkan peserta didik sebagai sarana penyelidikan dan percobaan sebelum mereka masuk ke tahap pembelajaran dengan *real equipment* atau peralatan yang nyata (Dobrzański, 2010:197). Teknologi simulasi juga digunakan dalam industri manufaktur untuk mengintegrasikan berbagai aktivitas proses dan aspek yang terkait dengan *manufacturing*, untuk menghasilkan sebuah *cost reduction* dan *profitability* dalam sebuah industri. Berbagai perusahaan saat ini menggunakan teknologi simulasi untuk mengoptimalkan berbagai faktor yang memengaruhi *profitability* dari produk yang terkait dengan perusahaan tersebut. Teknologi

simulasi bertindak seperti agen yang menjelaskan berbagai faktor yang menghasilkan *cost reduction*, penggunaan material yang lebih efisien, dan sebagainya (Talekar, 2017:148).

Model simulasi dalam CBI pada dasarnya merupakan salah satu dari strategi pembelajaran yang memiliki tujuan untuk memberikan pengalaman belajar yang lebih realis, melalui berbagai tiruan bentuk yang mendekati suasana yang sebenarnya. Biasanya *software* simulasi didesain menarik, imajinatif dan menghibur. Kemampuan multimedia pada media pembelajaran berbasis komputer dapat digunakan sebagai sarana dalam melakukan model pembelajaran berbasis simulasi untuk melatih ketrampilan skill dan kompetensi tertentu. Penggunaan media berupa simulator kokpit pesawat terbang pada sekolah kejuruan dan akademi penerbangan adalah salah satu contohnya, sebagai upaya meminimalisir resiko dalam kegiatan praktikum. Proses pelatihan ketrampilan berlangsung lebih aman, efisien dan representatif. Pembelajaran berbasis simulasi seperti CAD *(computer-aided design)* agar lebih efektif dan efisien harus ada media yang dapat digunakan oleh siswa secara individu (Ramadhani, 2016:38).

Pembelajaran untuk memahami kinerja dari sebuah sistem yang sulit diamati secara langsung tentu akan lebih efektif menggunakan teknik pemodelan dan simulasi *(modeling & simulation)*. Teknik pemodelan dan simulasi pada hakikatnya memiliki pengertian sebagai berikut;

> *Modeling and Simulation is the process of representing the behavior of a real system by a collection of mathematical equations and logic. The term real system is synonymous with physical system —that is, a system whose behavior is based on matter and energy. Models can be*

> *broadly categorized as either staticor dynamic. In a static model, there is no energy transfer. Systems, which are static produce no motion, heat transfer, fluid flow, traveling waves, or any other changes... Simulation is the process of solving the model and is performed on a computer.* (Shetty, 2011:10)

Teknik pemodelan dan simulasi *(Modeling & Simulation)* adalah proses representasi dari sebuah sistem yang sesungguhnya atau piranti fisik yang sebenarnya dan dikemas dalam sebuah proses penyelesaian dalam kinerja komputer. Pengertian secara mudah dapat kita pahami bahwa teknik pemodelan dan simulasi merupakan diversifikasi sebuah kinerja sistem fisik menjadi kemasan dalam bentuk virtual pada kinerja komputer. Teknik alih teknologi semacam ini tentu akan memiliki kelebihan dari sisi kepraktisan, ekonomis, dan keamanan dalam pengoperasiannya, karena tidak memiliki resiko seperti halnya pada sistem fisik yang sebenarnya. Salah satu permasalahan yang sulit dalam simulasi adalah dalam menentukan suatu model simulasi apakah merupakan penyajian yang nyata dan dapat dipelajari secara akurat, serta apakah model tersebut sesuai dan telah mewakili dari sistem yang nyata (Joni, 2012:111).

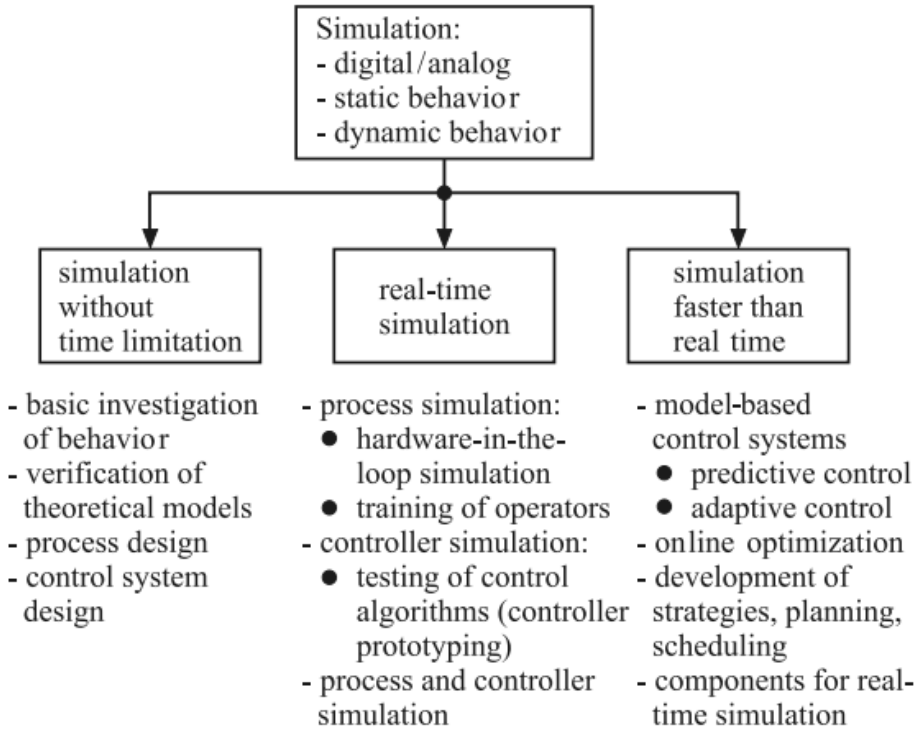
Inovasi pembelajaran teknologi yang ekonomis dan efektif yang ditonjolkan pada penelitian ini mengacu pada konsep efektifitas dalam pengadaan sarana-prasarana pembelajaran berbasis teknologi pada pendidikan berbasis industri, seperti Teknik Mekatronika, Elektronika Industri, dan sejenisnya. Konsep teknik simulasi sebagai diversifikasi *resources* juga lazim telah lama dilakukan oleh para praktisi industri. Inovasi teknologi yang

demikian tentu menjadi referensi pembelajaran yang berharga dan bermanfaat bagi para praktisi akademisi.

> *Besonders im Anlagen- und Fahrzeugbau, aber auch in vielen anderen Branchen wird die Visualisierung von Einbau- und Kollisionsuntersuchungen direkt am Bildschirm eingesetzt. Mithilfe einer schattierten Bildausgabe können viele Verständnisprobleme vermieden werden. Durch die Bereitstellung eines 3D-Produktmodells kann der Einbau simuliert werden, wobei neben Kollisions betrachtungen Einbauraum, Zugänglichkeit, Bauteilanordnung und andere Montageaspekte überprüfbar sind.*
> *..., gewinnen visuelle Darstellungen von Ergebnissen in Zukunft eine bedeutende Rolle, da sie allen angeschlossenen Interessenten sehr schnell zur Verfügung gestellt werden können.* (Vajna, 2018:293)

Teknik simulasi pada berbagai industri khususnya industri konstruksi, *sparepart* dan industri kendaraan, digunakan sebagai teknik diversifikasi *resources* untuk menekan angka resiko bahaya maupun angka biaya. Seperti pada aktivitas pembuatan visualisasi untuk mempelajari kualitas komponen kendaraan pada adegan tabrakan yang dimodelkan pada layar komputer. Menggunakan teknik tersebut, semua parameter, pengaturan komponen, adegan, aksesibilitas dan aspek pemasangan lainnya yang ingin diteliti dan dipelajari dapat dimasukkan dan disimulasikan. Teknik simulasi, representasi visual di masa depan akan memainkan peran penting, karena sangat membantu semua pihak yang tertarik dengan efisiensi (Vajna, 2018:293). Pembelajaran menggunakan media simulasi berbasis teknologi informasi dapat mengembangkan proses berpikir kritis dan kreatif siswa (Larasati, 2014:51). Keandalan dari sebuah simulasi ditentukan oleh tingkat kualitas *software* dan tingkat keakurasian model/perangkat (Surendro, 2010:71).

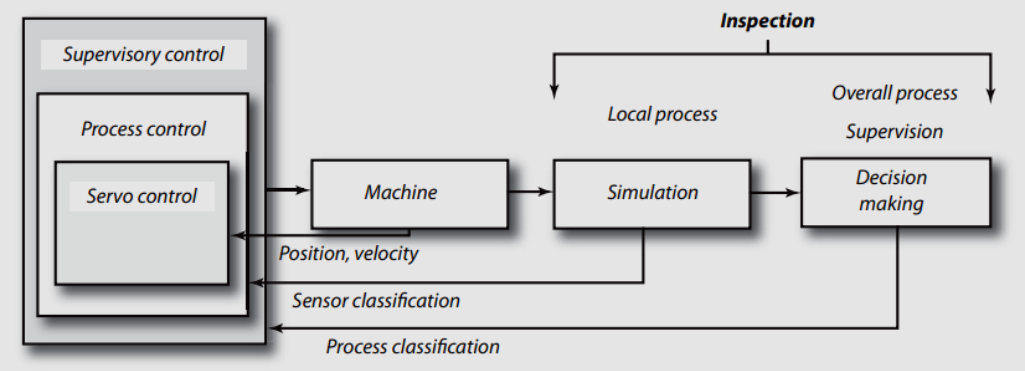
Teknik simulasi bekerja penuh pada perangkat berbasis komputer. Metode simulasi terdiri dari tiga jenis berdasarkan tingkat kebutuhan waktu dalam pemrosesan datanya (Bishop, 2006:2-12).



Gambar 2. Klasifikasi Teknik Simulasi dan Aplikasinya
(sumber: Buku *"Mechatronics - An Introduction"*, Robert H. B.)

Semua aplikasi teknologi terapan akan lebih jelas dipelajari dan diamati apabila ada suatu media pembelajaran yang inovatif yang dapat menyimulasikan fungsi dari masing-masing teknologi terapan tersebut (Rochayati, 2012:90). Menurut Deni Darmawan pembelajaran menggunakan alat bantu komputer model simulasi didesain untuk materi-materi yang banyak menampilkan proses, mekanisme, alur, sistem kerja yang perlu visualisasi berupa simulasi dengan animasi/video yang mendekati kondisi yang sebenarnya (Darmawan, 2012:60).

Teknik simulasi digunakan pula untuk fungsi yang lain, seperti digunakan sebagai sarana pemantauan dan pemeriksaan pada perangkat keras *(hardware)*. Proses kinerja perangkat lunak simulasi yang semacam ini sering diklasifikasikan sebagai tipe simulasi *real-time*, karena membutuhkan waktunya pemrosesan data sesuai dengan waktu yang sebenarnya *(real-time)*. Unit perangkat lunak simulasi semacam ini sering diterapkan pada proses produksi sebuah aktivitas manufaktur, yang terintegrasi dengan perangkat lunak CAD-CAM dan mesin CNC. Proses pengawasan *(supervisory control)* dan pemeriksaan *(inspection)* dalam kegiatan manufaktur sangat diperlukan guna menghasilkan sebuah efektivitas kerja dan orientasi kualitas pada produk yang dihasilkan, berikut skema kerja dari jenis simulasi tersebut;



Gambar 3. Teknik Simulasi Sebagai Pengawasan dan Pemeriksaan
(sumber: Buku *"Mechatronics System Design"*, Devdas Shetty)

Simulasi dapat digolongkan menjadi tiga macam (Purwoko, 2009:9); *(1) Simulasi interaktif dan fisikal*, simulasi yang lebih mengacu pada simulasi objek fisik untuk menggantikan sistem nyata. Objek fisik ini sering dipilih sebab mereka lebih kecil atau lebih murah dibanding

sistem atau obyek yang nyata. *(2) Simulasi komputer,* simulasi yang menyerupai situasi nyata dalam suatu komputer, sehingga situasi dapat dipelajari untuk dilihat bagaimana sistem itu bekerja. Dengan mengubah variabel, kemungkinan peramalan dapat dibuat tentang perilaku sistem itu. *(3) Simulasi dalam pelatihan,* simulasi yang sering digunakan pada pelatihan sipil dan personil militer. Simulasi dalam pelatihan umumnya dilakukan ketika terhalang oleh harga yang mahal atau terlalu berbahaya untuk mengijinkan anggota pelatihan untuk menggunakan peralatan yang nyata dalam dunia nyata.

Media simulasi yang baik adalah media yang mampu mencakup aspek diantaranya; *(1) Dapat merepresentasikan cara kerja dari sistem yang sebenarnya. (2) Dapat memberikan informasi yang sulit didapat dari kinerja sistem yang nyata. (3) Memiliki tingkat keamanan yang lebih daripada sistem yang sebenarnya. (4) Bersifat cost-effectiveness dari sistem yang sebenarnya* (Landriscina, 2013:6-7). Karakteristik media yang baik dalam *mobile learning* adalah memiliki karakteristik yang sesuai dengan perangkat yang bersangkutan *(mobile tools),* yaitu perangkat *mobile (mobile tools)* yang memiliki karakteristik berupa ukurannya yang kecil dan resolusi layar *(screen)* yang kecil pula, tentu memerlukan perangkat lunak yang mampu menyesuaikan (Pachler, 2010:68). Hal tersebut tentu mengarah pada spesifikasi media perangkat lunak yang harus dibuat berorientasi pada penyesuaian desain visualnya. Tentu berupa gambar, warna, tombol, animasi, huruf, yang sesuai dengan karakteristik perangkat *mobile* yang mayoritas

berukuran kecil tersebut. Penggunaan kombinasi antara simulasi dan aktifitas laboratorium, secara signifikan meningkatkan perolehan hasil pembelajaran siswa dibandingkan dengan perolehan pembelajaran yang hanya menggunakan salah satu aktifitas saja. Simulasi juga dapat menstimulasi siswa agar tertarik pada subyek pembelajaran, ketika simulasi komputer meniru aktifitas nyata. Simulasi yang tidak beresiko merusak peralatan diharapkan dapat meningkatkan kreatifitas peserta didik dalam berinovasi (Hariwibowo, 2014:63-64).

Konsep simulasi merupakan representasi dari sebuah kondisi yang nyata, baik berupa visual aktivitas sekitar kita, sebuah sistem kerja, maupun hal lain yang tidak mampu tertangkap langsung oleh indra visual kita. Definisi lain menjelaskan bahwa;

> *the term simulation is defined as models that have been implemented in a temporal manner. The implementation may take one of three forms: live, virtual, and constructive simulation.* (Sokolowski, 2011:24).

Istilah dari simulasi merupakan sebuah gambaran pemodelan yang berada dalam tatanan lingkungan yang tidak nyata. Sebuah simulasi dalam penerapannya bisa saja dalam berbagai bentuk yang bervariatif, seperti *live simulation, virtual simulation* dan *constructive simulation.* Bentuk *live simulation* adalah representasi dari visual kehidupan manusia secara mirip dengan kenyataan, beserta segala kelengkapan dan lingkungan sekitarnya. Sebuah *live simulation* merupakan gambaran dari suasana yang memiliki konsep *real-world.* Contoh dari bentuk *live simulation* adalah teknik rekayasa

pembuatan situasi pengamanan tamu negara, pengamanan unjuk rasa, training agen intelejen negara dan sebagainya.

Karakteristik dari *virtual simulation* adalah representasi dari visual kehidupan manusia secara mirip dengan kenyataan, beserta segala kelengkapan dan lingkungan sekitarnya namun dikemas dalam bentuk lain, umumnya selalu dalam bentuk perangkat audio visual digital. Contoh dari bentuk *virtual simulation* adalah *layout* 3D mesin CNC, *monitoring* sistem SCADA, simulator cockpit pesawat sebagai sarana pendidikan pilot penerbang. Semua kelengkapan dibuat sedemikian rupa mirip seperti sistem aslinya, namun untuk biaya dan resiko insiden, jauh lebih kecil. Selanjutnya adalah bentuk *constructive simulation* memiliki pengertian representasi dari visual kehidupan manusia secara mirip dengan kenyataan, beserta segala kelengkapan dan lingkungan sekitarnya namun dikemas lebih animatif dan imajinatif. Sehingga memiliki kesan berlebihan dibandingkan dengan kenyataan yang sesungguhnya. Umumnya simulasi ini sering kita jumpai pada berbagai game berbasis komputer dengan jenis simulasi. Sehingga dikemas dengan efek visual yang melebihi dari logika yang sesungguhnya sebagai nilai tambah dari daya jual produk game itu sendiri.

Teknik simulasi dan pemodelan merupakan bentuk yang komplek dari peran perangkat keras *(hardware)* dan perangkat lunak *(software)*. Perangkat keras terdiri dari komponen inti pendukung seperti *chip* prosesor, *integrated circuit (IC),* dan berbagai komponen elektronik pendukung lainnya. Pada aspek perangkat lunak terdiri dari perangkat desain tampilan *(user*

interface/UI), bahasa instruksi untuk pemprogramman, dan sistem instruksi *(command)* perangkat lunak yang ditanamkan pada perangkat keras *(embedded system)*. Semua gambaran tersebut merupakan rangkaian dari sebuah sistem yang bernama *'digital system'*. Hal ini seperti yang telah dijelaskan dalam sebuah buku berjudul *'Introduction to Digital Systems - Modeling, Synthesis & Simulation Using VHDL',* yang menjelaskan bahwa;

> *Modeling and simulation have their roots in digital systems. Long before they became the basis of an interdisciplinary field, modeling and simulation were used extensively in digital system design. As electronic and computer technology advanced , so did modeling and simulation concepts. Today, the many computer-aided design (CAD) tools are pushing the limit of modeling, synthesis, and simulation technology.* (Ferdjallah, 2011:1).

Teknik simulasi dan pemodelan memiliki akar dalam sistem digital. Jauh sebelum teknik simulasi menjadi dasar dari bidang interdisipliner ilmu, pemodelan dan simulasi digunakan secara luas dalam desain sistem digital. Sebagai teknologi elektronik dan komputer canggih, begitu pula konsep dari sebuah teknik simulasi dan pemodelan. Hingga saat ini teknik simulasi telah hadir dengan sebuah teknologi CAD *(computer-aided design)* yang menjadi alat yang utama dalam aktivitas pemodelan, sintesis, dan teknologi simulasi.

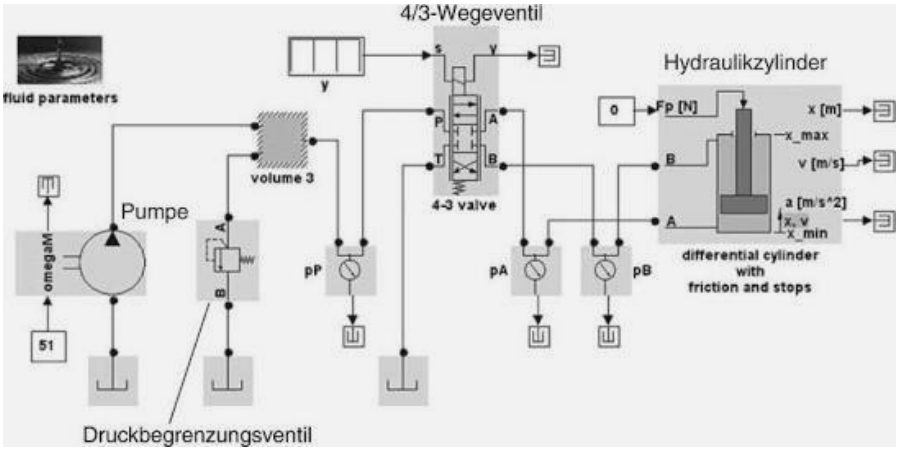
Teknologi CAD *(computer-aided design)* merupakan teknologi alat bantu yang hingga saat ini belum bisa tergantikan. Bersama dengan teknologi CAM *(computer-aided manufacturing),* teknologi CAD berperan dominan dalam semua aktivitas pembuatan berbagai produk industri, mulai proses desain hingga proses manufaktur. Produk industri mulai dari bolpoin, buku, kaleng minuman dan makanan, ponsel, laptop, komputer, mobil hingga

pesawat, semua produk tersebut memerlukan peran dari teknologi CAD-CAM dalam proses pembuatannya. Teknik CAD-CAM merupakan cabang keilmuan teknik yang masuk didalam teknologi keilmuan CAx, berikut adalah daftar berbagai keilmuan teknik yang masuk didalam keluarga teknik CAx (Vajna, 2018:15);

Tabel 1. Lingkup Keluarga CAx
(sumber: Buku *"CAx für Ingenieure - Eine praxisbezogene Einführung"* Sandor Vajna)

CAD	*Computer-Aided Design*
CAE	*Computer-Aided Engineering*
CAID	*Computer-Aided Industrial Design*
CAO	*Computer-Aided Optimisation*
CAT	*Computer-Aided Testing*
DMU	*Digital Mock-up*
VR	*Virtual Reality*
KBE	*Knowledge-based Engineering*
RP, RT, RPT	*Rapid Prototyping/Tooling*
CA(P)P	*Computer-Aided (Process) Planning*
PDM	*Product Data Management*
NC	*NC-Programmierung*
MES	*Manufacturing Execution System*
CAM	*Computer-Aided Manufacturing*
CAQ	*Computer-Aided Quality Assurance*
ERP	*Enterprise Resource Planning*
PPS	*Produktionsplanung und –steuerung*
BDE	*Betriebsdatenerfassung*
PLM	*Product Lifecycle Management*

Teknik *modeling* dan *simulation* sistem CAx digunakan hampir disemua bidang keilmuan teknik industri, seperti pada teknik mekanika, hidrolika, teknik elektro dan elektronik, serta teknik mekatronik (Vajna, 2018:375). Berikut ini contoh salah satu *software* simulasi pada teknik hidrolika.



Gambar 4. *Hydraulic Simulation Software*

Kelebihan simulasi antara lain (Rosadi, 2012:37); (1) Sistem nyata sulit diamati secara langsung, (2) Mampu memberikan perkiraan sistem yang lebih nyata sesuai operasional dari kumpulan pekerjaan, (3) Pengamatan sistem secara langsung tidak dimungkinkan karena sangat mahal, memakan waktu yang terlalu lama, akan merusak sistem yang sedang berjalan, (4) Solusi analitik tidak dapat dikembangkan, karena sistem yang digunakan di dunia kerja sangat kompleks, simulasi dapat memberi solusi apabila model analitik gagal, (5) Memudahkan pengontrolan lebih banyak kondisi dari suatu percobaan sehingga dimungkinkan untuk dicoba diterapkan secara nyata pada sistem tersebut, (6) Menyediakan sarana untuk mempelajari sistem dalam waktu yang cukup lama (lebih ekonomis) dengan proses yang membutuhkan waktu cukup singkat ataupun sebagai alternatif pembelajaran yang lebih rinci dan jelas tentang perilaku suatu sistem nyata yang prosesnya lebih panjang.

Membangun sebuah media belajar yang baik berbasis simulasi dan pemodelan *(simulation & modelling)* membutuhkan berbagai aspek. Sumber

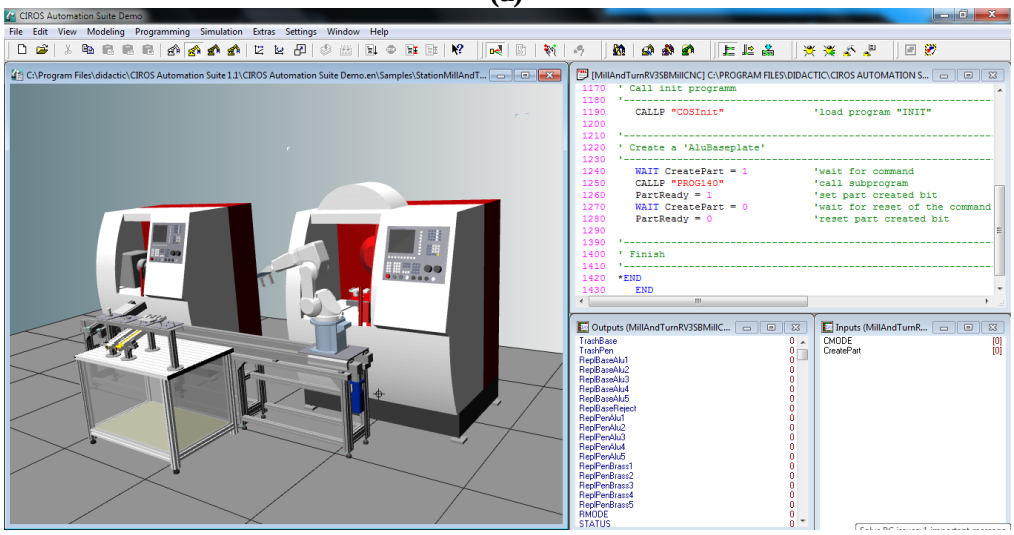
dari sebuah buku berjudul *'Building Software for Simulation; Theory and Algorithms',* menyatakan bahwa;

> *Modeling and simulation has many dimensions; software design and programming are two of these and are certainly important for building simulators. Two other facets are discussed above: (1) simulation tools, which include special programming languages and (2) the many theories of discrete-event systems, which link simulation models with other methods of analysis.* (Nutaro, 2011:277).

Perangkat lunak *builder* dan *desainer*, serta perangkat lunak untuk memberikan instruksi perintah program *(software programming)* merupakan dua aspek yang dibutuhkan dalam membangun sebuah perangkat lunak berbasis simulasi. Perangkat lunak utama yang memiliki spesifikasi sebagai *software designer* tentu memiliki peran sebagai alat utama untuk membangun sebuah perangkat lunak simulasi yang akan dibuat. *Software designer* memiliki peran utama pada semua proses produksi yang berhubungan dengan *UI (user interface)*, tampilan, warna, *layout*, fitur menu, tombol dan sejenisnya. Pembuatan desain dan semua tampilan perangkat lunak tentu mengacu pada kerangka kerja, algoritma dan karakteristik sebuah perangkat lunak simulasi yang diharapkan. Tahap berikutnya dalam pembuatan sebuah perangkat lunak simulasi adalah pemberian instruksi kerja pada masing-masing fitur perangkat lunak, atau biasa dikenal dengan istilah *'programming'*. Tahap *programming* menggunakan berbagai bahasa instruksi *(programming languages)* yang memiliki karakteristik dan fungsi yang beraneka ragam. Ada bahasa pemrograman yang mudah dimengerti karena memakai bahasa instruksi yang lazim digunakan manusia, dan ada pula bahasa pemrograman yang hanya menggunakan simbol, angka, huruf, yang

hanya berupa kode *(source code)*, serta bahasa pemrograman berupa gabung antara dua karakteristik tersebut.

(a)



(b)

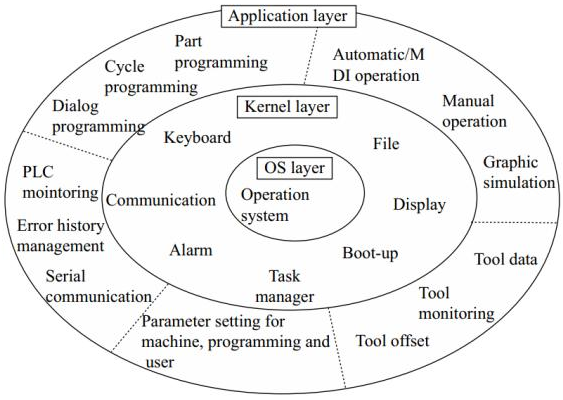
Gambar 5. Unit Mesin CNC : (a) Dalam Bentuk Sesungguhnya, (b) Bentuk Simulasi
(sumber: *"FESTO Technology for education and science"*, 2014)

Metode simulasi digunakan dalam aktivitas kinerja mesin CNC karena memiliki kelebihan berupa kemampuan fungsi untuk mendeteksi kesalahan dan memodifikasi pada proses *machining* sebelum proses *machining* sebenarnya dimulai.

> *The automatic programming method makes it easy to machine parts with complicated or 3D shapes. It also makes it possible to generate the large part programs in a short time. In addition, with computer*

simulation, it makes it possible to detect and modify machining errors before actual machining begins. (Suh, 2008:280).

Proses manufaktur sebuah produk industri menggunakan mesin CNC pada awalnya menggunakan bahasa perintah berupa kode-kode angka dan huruf. Seiring perkembangan ilmu pengetahuan muncullah teknik perintah pada mesin CNC berbasis metode simulasi, yang biasa dikenal dengan teknologi teknologi CAD-CAM *(Computer Aided Design – Computer Aided Manufacturing).* Teknologi CAD-CAM merupakan unit perangkat lunak *(software)* yang masuk dalam ranah MMI *(Man-Machine Interface)*, sebuah teknologi yang menghubungkan antara komunikasi manusia dengan mesin, dan bahasa manusia dengan bahasa mesin (Suh, 2008:271). Berikut gambaran dari struktur sebuah sistem MMI *(Man-Machine Interface)*;



Gambar 6. Struktur dari Sistem MMI *(Man-Machine Interface)*

Teknologi untuk menjembatani komunikasi antara manusia dengan mesin terus mengalami perubahan, hingga munculnya teknologi GUI *(Graphical User Interface)*. Teknologi GUI merupakan bahasa perintah *(language-type programming)* manusia pada mesin berbasis grafik bentuk *(shape)*. Bahasa

dalam gambar bentuk akan lebih mudah dimengerti oleh manusia dibanding bahasa kode mesin CNC (Suh, 2008:280).

Penggunaan teknik simulasi memiliki berbagai manfaatnya diantaranya adalah, (Pelz, 2003:5); (1) Dibandingkan dengan eksperimen nyata, teknik simulasi seringkali memerlukan lebih rendah biaya dan waktu, karena pada umumnya lebih murah untuk merancang prototipe virtual/simulasi daripada prototipe nyata; (2) Beberapa sistem yang mengandung resiko dan kerusakan pada sistem yang nyata tidak terdapat pada sistem simulasi; (3) Eksperimen dalam sistem simulasi dapat dilakukan berulang-ulang, yang tidak dapat dilakukan pada sistem yang sebenarnya, atau akan memerlukan biaya yang besar; (4) Model simulasi biasanya terkontrol sepenuhnya. Jadi semua variabel masukan dan parameter sistemnya dapat ditentukan sebelumnya, hal ini tidak dapat dilakukan pada sistem yang sebenarnya; (5) Model simulasi umumnya dapat dipantau sepenuhnya. Semua variabel *output* dan keadaan internal tersedia, sedangkan pada sistem sebenarnya setiap variabel yang akan dipantau melibatkan setidaknya biaya pengukuran yang signifikan. Selain itu, setiap pengukuran yang diambil mempengaruhi perilaku sistem; (6) Dalam beberapa kasus 'konstanta waktu' eksperimen dan pengamat tidak sesuai, seperti investigasi partikel elementer atau galaksi; (7) Dalam beberapa kasus, percobaan dikesampingkan karena alasan kode etik, misalnya eksperimen pada manusia di bidang teknologi medis.

3. Teknik Digital

Teknik digital dalam penelitian tesis ini bukanlah sebuah teknik digital visual atau biasa kita kenal dengan teknik digital dalam lingkup ilmu multimedia. Teknik digital yang dimaksud pada sub-bab ini adalah sebuah teknik dalam rumpun ilmu kelistrikan yang lazim digunakan sebagai teknik untuk berkomunikasi antara manusia dengan mesin. Karakteristik mesin yang dimaksud tentunya berupa mesin yang memiliki sebuah sistem kontrol elektris yang tertanam didalamnya. Teknik digital hingga saat ini telah digunakan dalam berbagai keperluan dan ditanamkan dalam berbagai perangkat elektronik yang biasa kita kenal, seperti pada kalkulator, televisi, telepon, piano, *disc jokey (DJ equipment), sound system, PC, tablet PC, smartphone* , dan berbagai produk turunan komputer lainnya.

Teknik digital berkembang seiring kebutuhan manusia akan sebuah perangkat pembantu dalam mengatasi kesulitan pada berbagai aktivitas sehari-hari, demikian pula kesulitan dalam sebuah industri manufaktur dalam aktivitas memproduksi berbagai produk kebutuhan hidup manusia. Perkembangan teknik digital menjadi pesat dan variatif seiring posisinya sebagai alat bantu dalam industri manufaktur, hingga terlahir produk turunan dari teknik digital seperti *microcontroller, microprocessor, Embedded System, PLC (Programmable Logic Controller), HMI (Human-Machine Interface), SCADA (Supervisory Control And Data Acquisition), CAD (Computer-aided Design), CAM (Computer-aided Manufacturing)* dan lain sebagainya.

Ilmu teknik digital lahir dari sebuah penciptaan logika simbolik yang sekarang dikenal dengan Aljabar Boole. Ilmu Aljabar berasal dari buku *"Al-jabr wa al-Muqabala" (The Compendious Book on Calculation by Completion and Balancing)* yang ditulis ilmuwan Persia, Musa Al-Khawarizmi sekitar tahun 790M. Kata Aljabar sendiri berasal dari bahasa *"al-jabr"* yang berarti "perampungan" atau "penyelesaian". Sistem Aljabar merupakan pengembangan matematika yang cukup rumit yang dipergunakan untuk menghitung solusi dari sebuah nilai yang tidak diketahui (Hidayani, 2012:01-02).

Aljabar Boole diciptakan pada tahun 1854 oleh George Boole sebagai pengembangan dari keilmuan induk Aljabar. Setiap variabel dalam Aljabar Boole hanya memiliki dua keadaan atau dua nilai, yaitu *'keadaan benar'* yang dinyatakan dengan (1), dan *'keadaan salah'* yang dinyatakan dengan (0). Aljabar Boole yang memiliki dua keadaan ini digunakan untuk menyelesaikan persoalan-persoalan logika (Widjanarka, 2006:22). Pada tahun 1938, Calude Shannon menggunakan Aljabar Boole sebagai praktik rangkaian penyaklaran *(switching)* telepon. Shannon menggunakannya untuk menyatakan terbuka dan tertutupnya saklar relay (saklar elektromagnetik), sehingga Aljabar Boole diterapkan pada ilmu teknologi elektronika dan elektronika komputer.

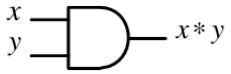
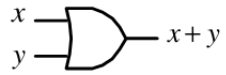
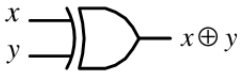
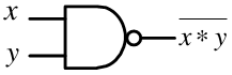
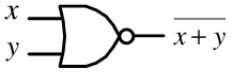
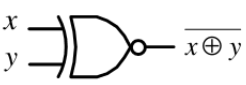
Aljabar Boole kemudian diwujudkan dalam sebuah piranti atau sistem yang disebut dengan "Gerbang Logika" (Widjanarka, 2006:22). Gerbang Logika *(Logic Gate)* adalah blok bangunan dasar untuk membentuk rangkaian

elektronika digital, yang digambarkan dengan simbol yang telah ditetapkan. Sebuah gerbang logika memiliki beberapa masukan *(input)*, tetapi hanya memiliki satu keluaran *(output)*. Keluarannya akan *HIGH* (1) atau *LOW* (0) tergantung pada level digital terminal masukan. Gerbang Logika digunakan untuk merancang dan mendesain suatu sistem yang akan dikendalikan level masukannya dan menghasilkan tanggapan keluaran tertentu, berdasarkan rancangan logika yang diinginkan. Gerbang Logika dapat diartikan sebagai elemen pengambil keputusan pada rangkaian elektronika digital. Gerbang Logika beroperasi pada sistem bilangan Biner, karena itu disebut Gerbang Logika Biner.

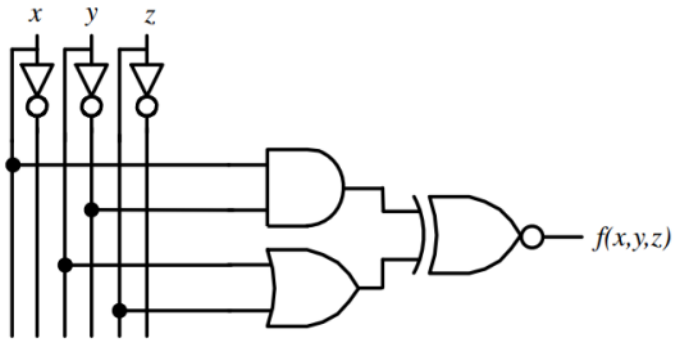
Mekanisme kerja pada sistem teknik digital di berbagai perangkat elektris merupakan hasil rekayasa teknologi para ilmuwan dalam mengembangkan bahan *chalcogenide-based* seperti *germanium*, *antimony*, dan *tellurium* (GeSbTe, atau GST). Material GST dirancang untuk kondisi *amorphous* dan *crystalline* sehingga dapat menghasilkan sinyal digital "0" atau "1". Material GST dapat dialihkan antara fase *crystalline* dan fase *amorphous* ketika GST mendingin dari suhu pemanasan yang berbeda, yang dapat dikontrol dengan menerapkan arus yang berbeda (Xiao, 2016:163).

Sistem digital merupakan sistem berupa Gerbang Logika *(Logic Gate)* yang terdiri dari NOT gate *(inverter)*, AND gate, NAND gate, OR gate, NOR gate, XOR gate *(Exclusive-OR)*, XNOR *(Exclusive-NOR)* (Ferdjallah, 2011:27). Simbol diagram dari berbagai *logic gate* tersebut adalah sebagai berikut:

Tabel 2. *Logic Gate*
(*"Introduction to Digital Systems; Modeling, Synthesis & Simulation VHDL"*, Mohammed Ferdjallah)

Truth Table	*Function*	*Symbol*
x | $\overline{x}$ 0 | 1 1 | 0	NOT	
x | y | $x*y$ 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	AND	
x | y | $x+y$ 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1	OR	
x | y | $x \oplus y$ 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	XOR	
x | y | $\overline{x*y}$ 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	NAND	
x | y | $\overline{x+y}$ 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 0	NOR	
x | y | $\overline{x \oplus y}$ 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	NXOR	

Proses perancangan sebuah sistem digital terdiri dari aktivitas meracik berbagai fungsi *logic gate* menjadi sebuah rangkaian sistem yang memiliki fungsi tertentu, rangkaian tersebut berupa bentuk *"Circuit Diagram"* (Ferdjallah, 2011:28). Suatu *circuit diagram* memiliki karakteristik kinerja sistem tertentu sesuai dengan komposisi *logic gate* yang ada didalamnya. Hasil keluaran sistem *(output)* yang dihasilkan merupakan representasi dari karakter *circuit diagram* yang dirancang, sehingga input yang diberikan pada sistem akan menghasilkan *output* yang sesuai dengan ketetapan sistem yang dirancang. Pada ilmu teknik digital, perancangan sebuah sistem digital dalam bentuk *circuit diagram* yang sudah disepakati secara internasional salah satu contohnya adalah sebagai berikut:



Gambar 7. *Logic Gate Circuit Diagram*
(*"Introduction to Digital Systems; Modeling, Synthesis & Simulation VHDL"*, Mohammed Ferdjallah)

Pola rangkaian *circuit diagram* tersebut membentuk sebuah sistem kinerja yang lazim disebut dengan *digital system*. Rangkaian sistem digital sangat berguna dalam banyak aplikasi ataupun perancangan kontrol

digital/komputer. Perancangan sistem biasanya menentukan spesifikasi yang dikeluarkan dalam bentuk fungsi dengan tabel kebenarannya (Polosoro, 2009:95). Aljabar Boole merupakan sebuah metode yang sangat berguna untuk mentransformasikan dari tabel kebenaran *(truth tabel)* ke rangkaian praktis. Sistem digital dapat pula digambarkan dalam sebuah bentuk persamaan matematika. Pada rangkaian diagram **(Gambar 7)** memiliki sebuah notasi dalam persamaan matematika sebagai berikut:

$$f(x, y, z) = (x \cdot \bar{y}) \oplus (y + z)$$

Nilai biner yang akan diberikan pada *input* (x,y,z) akan menghasilkan sebuah nilai *output* biner berupa $f(x, y, z) = (x \cdot \bar{y}) \oplus (y + z)$. Pada rangkaian diagram tersebut akan membentuk sebuah sistem digital yang memiliki karakter apabila *input* (x,y,z) diberikan kombinasi angka; x=0, y=0 dan z=0, maka akan menghasilkan (f=1).

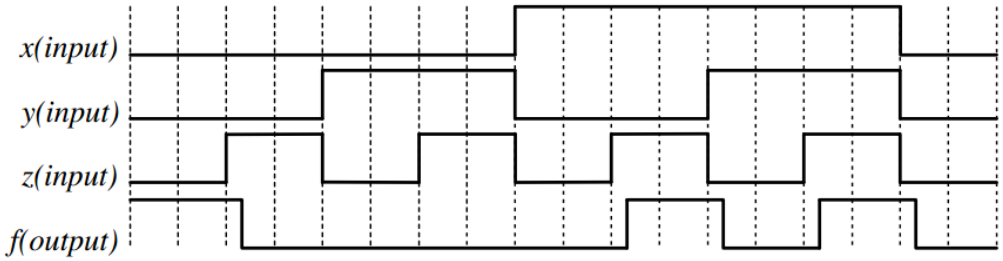
Bila *input* (x,y,z) diberikan kombinasi angka x=1, y=1 dan z=1, maka akan menghasilkan (f=1). Hal ini dapat dilakukan dalam berbagai bentuk komposisi angka biner yang ingin diberikan pada *input* (x,y,z). Hasil *output* (f) yang dihasilkan juga akan bervariasi sesuai dengan komposisi angka biner yang diberikan pada *input*. Pemberian angka biner pada *input* yang hanya terdiri dari satu digit ini lazim disebut dengan *single bit* (1bit). Pada rangkaian sistem digital tersebut dapat dipahami bahwa sistem tersebut bekerja pada operasi sistem 1bit *input* dan 1bit *output*.

Karakteristik kinerja sistem saat diberikan instruksi pada *input* dan menghasilkan sebuah *output* dapat diuraikan dalam bentuk tabel, yang lazim disebut *"Truth Tabel"*. Pada rangkaian sistem digital **(Gambar 7)** dapat konversikan dalam bentuk tabel sebagai berikut:

Tabel 3. *Truth Tabel*

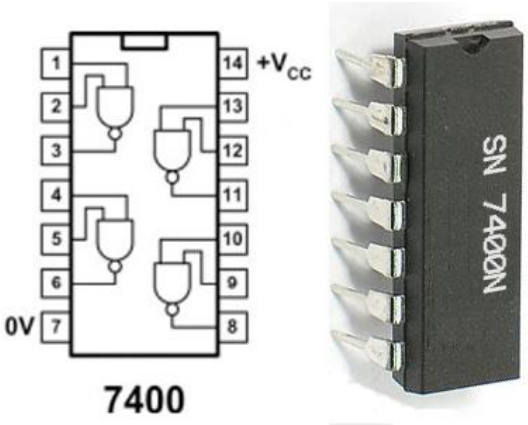
Row	*x*	*y*	*z*	*f*
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

Sebuah *circuit diagram* sistem digital dapat diinformasikan dalam bentuk *"timing diagram"*. Sebuah diagram waktu *(timing diagram)* merupakan representasi secara grafis perjalanan kinerja *input* dan *output* dalam suatu satuan waktu (Ferdjallah, 2011:31). Diagram waktu menggambarkan kondisi *input* dan *output* saat terdapat aktivitas berstatus nilai (0) atau (1). Pada komponen elektronik *logic gate* yang sebenarnya, aktivitas ini hanya dapat dilihat menggunakan alat bantu lain seperti *oscilloscope*, *logic analyzer* dan sejenisnya. Pada perangkat teknologi seperti *Computer-aided Desain* (CAD), *Computer-aided Manufacturing* (CAM), memiliki perangkat lunak *(sofware)* yang dapat menghasilkan simulasi *timing diagram* dari komponen elektronik *logic gate*. Bentuk *timing diagram* secara umum dari *circuit diagram* diatas adalah sebagai berikut:



Gambar 8. *Timing Diagram*

Konsep *logic gate* dalam ilmu teknik digital, dikemas dalam sebuah piranti elektronik kecil yang bernama *IC (Integrated Circuit)*. Piranti *IC* merupakan komponen elektronik yang terdiri dari perangkat transistor dan bermaterial semikonduktor yang dikemas dalam sebuah ukuran yang diperkecil. Sebuah *IC* sering disebut pula dengan *(micro) chips*. Penciptaan *IC* merupakan wujud dari inovasi komponen elektronik yang secara kuantitasnya lebih dirancang efisien dan praktis. Sebagai contohnya adalah sebuah *logic gate, flip-flop, counter,* berbagai variasi dari keluarga IC 7400, serta *RAM (Random Access Memory), ROM (Read-Only Memory), Microcomputer, DSP (Digital Signal Processors)* (Kaeslin, 2008:4-5).



Gambar 9. *Integrated Circuit (IC)* NAND 7400 *SSI*

Rangkaian digital atau sering disebut *logic gate* memiliki berbagai macam tingkat kepadatan masing-masing komponen *gate*nya dalam setiap *chip* (Ayers, 2005:4). Berbagai tingkatan teknologi integrasi tersebut secara berurutan dari yang rendah hingga yang tinggi adalah; *SSI*, *MSI*, *LSI*, *VLSI*.

Tabel 4. Tingkatan Kepadatan *IC*
(sumber: *"Digital Integrated Circuits - Analysis and Design"*, *John E. Ayers*)

Levels of Integration

Level of Integration		**Gates/chip**
Small-scale integration	SSI	1–10
Medium-scale integration	MSI	10–100
Large-scale integration	LSI	$100–10^4$
Very large-scale integration	VLSI	$>10^4$

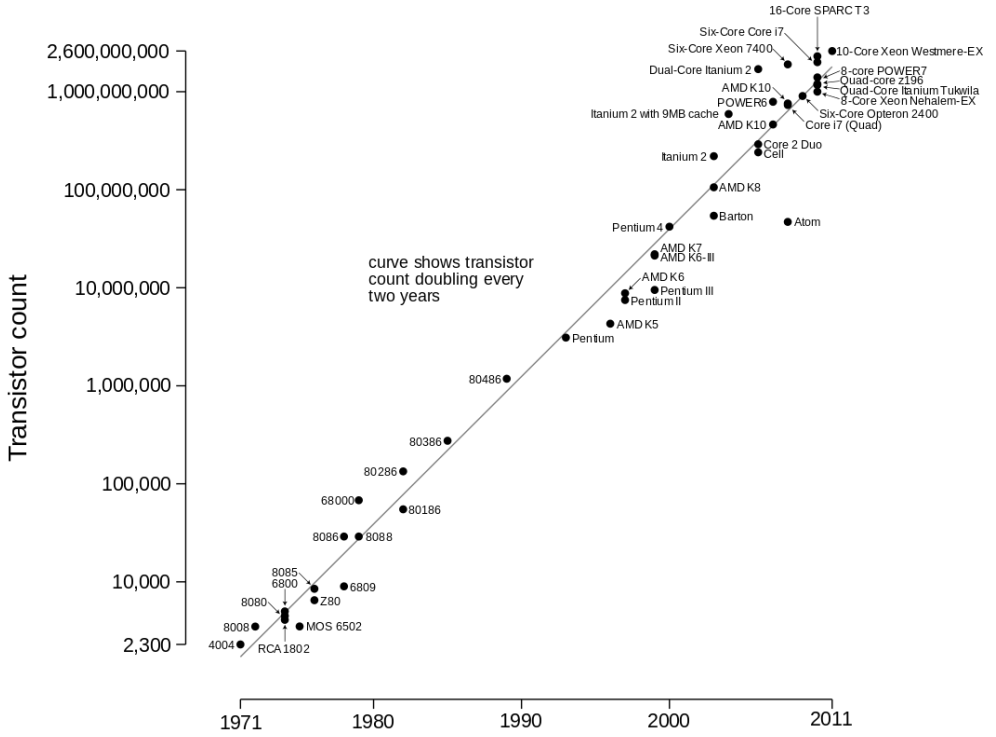
Jenis *SSI (Small-scale Integration)* merupakan teknologi integrasi *logic gate* yang dalam satu kemasan *chip* berisi 1-10 *gate*, seperti pada *IC* dari keluarga 7400 *TTL* **(Gambar 9)**. Jenis *MSI (Medium-scale Integration)* dalam satu kemasan *chip* berisi 10-100 *gate*. Pada jenis *LSI (Large-scale Integration)* dalam satu kemasan *chip* berisi 100-10.000 *gate*. *VLSI (Very Large-scale Integration)* memiliki lebih dari 10.000 gate setiap satu ke masan *chip*. Pada tahun 1995, *IC VSLI* teknologinya rata-rata sudah terdiri dari 5.500.000 transistor dalam satu *chip* kemasan (Widjanarka, 2006:124). Teknologi *LSI* dan *VLSI* biasa digunakan untuk keperluan sistem yang lebih besar dan lengkap seperti *microprocessor*, *single-chip microcomputer*, dan *memory chip* (Ibrahim, 1996:49).

Perkembangan teknologi pembuatan *IC* dalam bentuk *chip* yang terintegrasi, dari tahun ke tahun berkembang sangat pesat. Perkembangan tersebut dapat disaksikan dalam grafik Moore pada halaman selanjutnya **(Gambar 10)**. Teori Moore atau biasa disebut "Hukum Moore" mengatakan bahwa, perkembangan teknologi integrasi pada satu kemasan *chip* meningkatkan dua kali lipat jumlah transistor didalamnya setiap 18 bulannya (Ayers, 2005:2). Pada akhirnya membuat saat ini banyak kita jumpai berbagai varian teknologi turunan dari *chip* melimpah disekitar kita, baik dalam berbagai keperluan dan merek.

Perkembangan teknologi *chip* dari masa ke masa secara fisik semakin kecil dan ramping, seiring kebutuhan manusia akan perangkat turunan dari komputer yang lebih praktis dan ramping pula. Perkembangan teknologi integrasi *chip* selalu melibatkan banyak para ahli, karena dari masa ke masa kebutuhan manusia menuntut sebuah produk *chip* yang secara kuantitas semakin besar fungsinya, namun secara fisik bentuknya menuntut semakin kecil. Pada proses merancang produk *chip* yang secara fisik semakin kecil, namun dituntut mampu mengakomodasi jumlah transistor yang banyak, akan melibatkan banyak para ahli peneliti baik dari ranah fungsi elektronis, desain, maupun ketahanan material yang digunakan.

Berbagai riset kearah inovasi dari aspek desain dan material sebuah *chip IC (Integrated Circuit)* terus dilakukan. Seperti riset inovasi *IC* menggunakan material *Graphene* dengan desain *Wafer-Scale*. Sirkuit *wafer-scale graphene* memperlihatkan di mana semua komponen sirkuit, termasuk

graphene field-effect transistor dan induktor, secara monolitis terintegrasi dalam sebuah *single silicon carbide wafer*. *Graphene circuits* menunjukkan stabilitas termal yang memiliki reduksi sedikit dalam kinerja (kurang dari 1 desibel) antara 300 dan 400 kelvin. Hasil ini membuka kemungkinan untuk mencapai teknologi *practical graphene* dengan fungsionalitas dan kinerja yang lebih kompleks (Lin, 2011:1294).

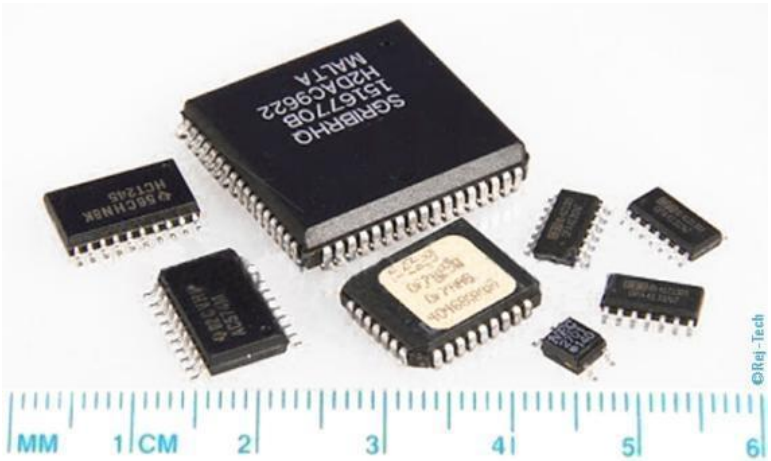


Gambar 10. Grafik Perkembangan Mikroprosesor

Piranti *logic gate* (gerbang logika) pada bidang ilmu teknik digital yang dikemas dalam bentuk *IC*, secara material dan desain tata letak internal ada beberapa teknologi. Teknologi *IC* tersebut antara lain; *TTL (Transistor-Transistor Logic), ECL (Emitter-Coupled Logic), CMOS (Complementary Metal-Oxide Semiconductor)* (Leblebici, 2003:534). Teknologi yang banyak

beredar dipasaran adalah tipe TTL dan CMOS. Setiap teknologi dikelompokkan menjadi suatu keluarga. Teknologi keluarga TTL memiliki notasi kode produk 74XX. Teknologi keluarga CMOS memiliki lebih banyak varian, mulai dari CMOS seri 4000, 40H00, dan masih banyak lagi. Setiap notasi pada kode memiliki makna berupa teknologi yang ada didalamnya, baik teknologi desain internalnya maupun performanya.

Teknologi TTL dan CMOS memiliki berbagai perbedaan dari banyak sisi teknis, fungsi, kinerja, desain internalnya, dan konsumsi listriknya. Namum teknologi TTL maupun CMOS memiliki satu kesamaan pada aspek desain kemasan luarnya. Teknologi TTL dan CMOS sama-sama berbentuk kemasan terpadu kotak hitam kecil, dengan beberapa pin atau kaki. Teknologi desain kemasan hingga saat ini belum ada perubahan dalam bentuk lain.



Gambar 11. Piranti *Integrated Circuit (IC)* jenis *TTL, ECL* & *CMOS*

Rangkaian *input/output (I/O)* yang dikemas sedemikian rupa dan dicetak/dicor padat silikon hitam merupakan sebuah perwujudan kualitas dan kehandalan terhadap berbagai gangguan eksternal berupa *noise* sinyal luar,

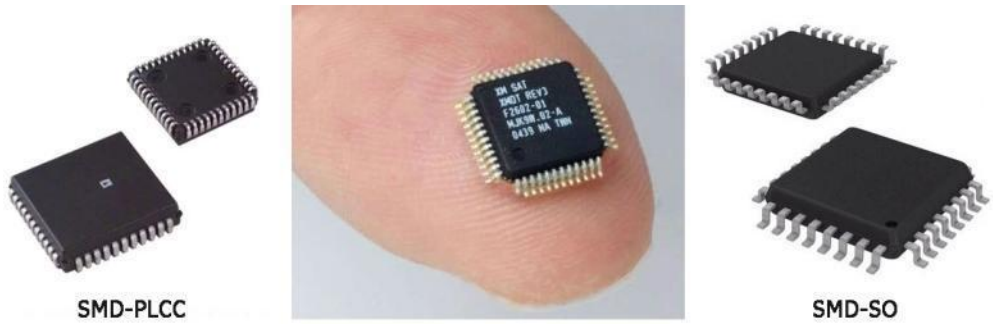
electrostatic discharge (ESD), dan berbagai bahaya lain (Leblebici, 2003:534). Piranti *IC* dari keluarga *TTL* yang paling standar mempunyai nomor awal 74XX, sementara *CMOS* mempunyai nomor awal 40XX. Jenis *TTL* 7400 memiliki 4 gerbang NAND dan dua *input* **(Gambar 9)**. Jenis *CMOS* 4001 memiliki 4 gerbang NOR dan dua *input* .

Kode notasi pada nomor *IC* melambangkan spesifikasi teknis yang ada didalamnya, sebagai contoh notasi *"S"* dan *"L"*. Kode notasi *"S"* menjelaskan bahwa rangkaian gate didalam *IC* jenis rangkaian *"Schottky"*. Kode notasi *"L"* menjelaskan bahwa *IC* jenis *"Low"* atau daya rendah (hemat energi). Piranti *IC* jenis *ECL* digunakan jika kecepatan merupakan pertimbangan utama, dengan mengorbankan faktor konsumsi listrik. Jenis *ECL* pada kenyataannya saat ini jarang yang menggunakan karena sifatnya yang demikian. Berikut adalah karakteristik dari spesifikasi *TTL, ECL dan CMOS* ;

Tabel 5. Perbandingan Piranti *Logic Gate*
(sumber: *"Digital Techniques"* K.L. Ibrahim)

	TTL	TTL (Low)	TTL (*Schottky*)	ECL	CMOS
No. Seri	7400	74L	74S	10	4000
Fan out	8-10	8-10	8-10	000	20
Fan in	8	8	8	15	8
Konsumsi Daya (mW)	20	2	20	5	0,001
Kecepatan (*nanosecond*)	10	30	3	1-2	15-20

Perkembangan teknologi piranti sistem digital berupa *IC* terus mengalami inovasi oleh para ahli, baik para ahli desain, material, produksi, hingga analis uji sistem. Dinegara maju penemuan dan pengembangan teknologi piranti digital berkembang dalam hitungan jam. Penelitian akan produk *IC* yang memiliki kepadatan tinggi dan mampu mengakomodasi lebih banyak fungsi, melahirkan sebuah teknologi baru bernama *SMT (Surface-Mount Technology)*. Piranti dan berbagai komponen *SMT* disebut *SMD (Surface-Mount Device)*. Piranti *SMD* berupa *IC* yang secara teknis memiliki kepadatan lebih tinggi pada komponen didalamnya, namun dengan desain kemasan secara keseluruhan lebih kecil (Kaeslin, 2008:547-548).



Gambar 12. Piranti *Integrated Circuit (IC)* teknologi *SMD*

Proses produksi SMT terdiri dari tiga langkah utama; 1) pencetakan pasta solder, di mana pasta solder yang terdiri dari *flux* dan *solderable metal particles* dicetak pada permukaan papan, 2) penempatan, di mana komponen elektronik diposisikan pada *pasted pads* pada *board*, 3) *reflow soldering*, di mana sambungan solder dibentuk melalui peleburan antara pin komponen dan

board pads dengan memanaskan dan mendinginkan pasta solder yang dicetak. Beberapa cacat dapat terjadi selama setiap langkah proses produksi, misal variasi parameter atau material dapat menyebabkan sambungan solder rusak yang menyebabkan kegagalan komponen akhir (Acciani, 2011:115).

Anatomi *SMD* yang sedemikian rupa membuat perakitannya mudah dalam produk masal industri mikro komputer. Proses perakitan tidak memerlukan lagi aktivitas pelubangan *PCB* untuk menanamkan *IC* seperti halnya pada jenis *IC* teknologi sebelum *SMD*. Piranti *SMD* memiliki dua jenis, yaitu *SO (Small Outline)* dan *PLCC (Plastic Leaded Chip Carrier)* (Ayers, 2005:637). Perbedaan mendasar pada *IC-SMD* jenis *PLCC* dan *SO* terletak pada desain pin *input/output (I/O)* beserta desain konfigurasi catu daya positif-negatifnya. Jenis *SO* memiliki desain pin *I/O* yang menjulur keluar hingga melebihi ukuran bodi kemasannya. Jenis *PLCC* memiliki desain pin *I/O* yang nyaris simetris dengan ukuran luas bodi kemasannya.

Desain produk *PLCC* yang demikian, menjadikan jenis *PLCC* favorit digunakan dalam berbagai perangkat *PC, Tablet PC, Smartphone, Flash memory*, yang berorientasi pada desain minimalis dan penggunaan tata letak komponen yang seefisien mungkin. Pada tahun 1988 sebuah perusahaan Jepang "SONY", membuat sebuah piranti *IC* Pengolah Sinyal Digital untuk mengolah sinyal suara dari piringan optik laser yang sangat canggih, menggunakan teknologi *SMD* jenis *PLCC*. Perangkat tersebut bernama *IC Digital Signal Processor* CXD 1125QZ/1130QZ/1135QZ. Ukuran *IC* CXD tersebut; panjang 20mm dan lebar 14mm. Produk tersebut digunakan sebagai

pengendali dan pengolah sinyal audio *Hi-Fi Compact Disc Player* untuk produk buatan SONY sendiri, dan dipasarkan tahun 1988 (Widjanarka, 2006:172).

Gambar 13. Penggunaan *IC-SMD* jenis *PLCC* pada *Smartphone*

Pada ilmu teknik digital, sebuah sistem digital yang telah dirancang melalui *circuit diagram* dapat digunakan dalam berbagai keperluan. Pada kehidupan sehari-hari biasa kita jumpai pada sistem penyimpanan file atau memori *(flip-flop)*, pencacah *(counter)*, pewaktu *(timer)*, dan *register*. Rangkaian *Flip-Flop* ada berbagai jenis, diantaranya *SR Flip-Flop, JK Flip-Flop*, *D Flip-Flop*, *T Flip-Flop* dan *Master Slave Flip-Flop* (Depari, 2013:121). Piranti penyimpanan pada sistem digital dapat kita jumpai dalam bentuk *magnetic disk*, *memory chip*. Sebuah *memory chip* terdiri dari sejumlah sel memori ke dalam mana berbagi *bit* data dapat disimpan (ditulis).

Data yang tersimpan dapat dicari atau dibaca kembali *(retrieve)* dari piranti tersebut. Sel-sel memori dikelompokkan membentuk suatu lokasi memori berupa 1*bit*, 2*bit*, 4*bit*, atau 8*bit* (Ibrahim, 1996:109). Data yang tersimpan dalam lokasi demikian disebut kata *(word)*. Sebuah kata tersusun atas beberapa *bit* dan merupakan unit dasar dari informasi pada sistem tersebut. Setiap lokasi mempunyai suatu kode biner yang unik yang digunakan untuk identifikasi yang disebut alamat *(address)*. Susunan kata yang terdiri dari 4*bit* disebut *nibble*, sedangkan untuk yang terdiri dari 8*bit* disebut *byte*.

Perkembangan saat ini piranti penyimpanan data pada sistem digital sangat pesat sejak awal pertama ditemukan piranti penyimpanan/memori. Seiring dengan kebutuhan manusia akan piranti penyimpanan data dalam kapasitas yang lebih besar. Produk dan teknologi baru memori yang bermunculan merupakan jawaban atas kebutuhan manusia akan piranti tersebut. Saat ini piranti memori seperti pita kaset, *floppy, disk*, telah tergantikan oleh piranti seperti *SD card, micro SD card, flash disk, HDD (Hard Disk Drive), SSD (Solid State Drive),* dengan kapasitas penyimpanan yang lebih besar hingga satuan *Terra Byte (TB)*. Pada kategori piranti *chip memory* jenis lain yang umum dikenal antara lain, *ROM (Read-Only Memory), PROM (Pogrammable Read-Only Memory), EPROM (Erasable Pogrammable Read-Only Memory), EEPROM (Electrical Erasable Pogrammable Read-Only Memory), RAM (Random Access Memory), SRAM (Static Random Access Memory), DRAM (Dynamic Random Access Memory).*

Gambar 14. Ragam Piranti Sistem Digital Penyimpan Data

Suatu rangkaian dalam sistem digital dapat dengan mudah dipahami sifat-sifat sistemnya, kinerjanya, hasilnya dan bahkan kecacatan sistemnya, jika pada keluarannya *(output)* dipasang sebuah beban. Beban yang digunakan adalah *LED*, yaitu dioda yang dapat memancarkan cahaya (Widjanarka, 2006:168). Pemberian *LED* pada *output* memudahkan pemahaman secara visual dari karakteristik dan kinerja sistem digital yang telah dirancang. Piranti *LED* ada berbagai jenis, salah satu jenis yang paling lazim digunakan untuk memahami kinerja sistem melalui konversi antar jenis bilangan adalah *LED seven segment*. Konversi antar bilangan yang umum adalah antara bilangan desimal, bilangan yang umum digunakan oleh manusia dan bilangan biner, bilangan yang menjadi ketetapan dari mesin/perangkat.

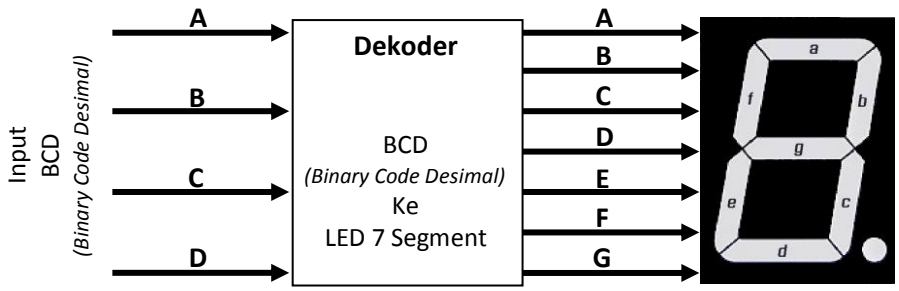
Pada proses alih bahasa antara manusia dengan mesin, atau berupa bilangan desimal dan bilangan biner memerlukan sebuah sistem yang bernama *"Decoder-Encoder"*. Dekoder adalah suatu rangkaian *logic gate*

yang mengubah suatu kode *input* biner dalam *bit* tertentu menjadi *output* sedemikian rupa sesuai dengan nilai gabungan *input* (Depari, 20013:111). Pada proses penerjemahan di sebuah sistem dekoder, memerlukan sistem konversi angka bernama *BCD (Binary-code Decimal)*. Sistem *BCD* memiliki banyak jenis, diantaranya ada *BCD 8421, BCD 7421, BCD 6311, BCD 5421, BCD 5311,* hingga *BCD 2421*. Berikut adalah salah satu contoh kode bilangan konversi *I/O (input-output)* dari *BCD 8421*:

Tabel 6. *BCD (Binary-code Decimal)*

Angka Desimal	Input				Output						
	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1

Pada uraian sebelumnya telah dijelaskan bahwa sebuah sistem digital yang telah dirancang melalui *circuit diagram* akan mudah dipahami sifat-sifat sistemnya, kinerjanya apabila pada *output* dipasang sebuah perangkat untuk memvisualkan. Karena pada dasarnya kinerja dari sebuah sistem digital tidak bisa kita amati dengan mata telanjang. Pemasangan perangkat *LED seven segment* pada *output* merupakan salah satu cara untuk mensimulasikan secara visual agar mudah dipahami kinerja sistemnya. Gambar berikut ini adalah rangkaian dekoder ketika pada *output* dipasang perangkat *LED seven segment* untuk mensimulasikan kode angka pada *BCD (Binary-code Decimal)*;



Gambar 15. Sistem Kerja *I/O* Dekoder pada 7 Segmen

Konversi antara *BCD input* hingga menghasilkan susunan biner *output* **(Tabel 6)**, ketika disimulasikan dalam piranti *LED seven segment* akan menayangkan kombinasi nyala lampu yang sesuai dengan format angka desimal yang diberikan pada *input*. Salah satu contoh ketika sebuah mesin/sistem akan memberikan informasi berupa angka 6, maka mesin akan mengirim input berupa; A=0, B=1, C=1, D=0 pada rangkaian *logic gate*, sehingga *output* akan menayangkan; a=0, b=0, c=1, d=1, e=1, f=1, g=1. Pada *output* yang berupa sebuah piranti *LED seven segment*, maka akan menayangkan kombinasi nyala lampu pada *output* (c,d,e,f,g). Hal tersebut akan terlihat *LED seven segment* menayangkan format angka 6 desimal pada masing-masing segmen lampu yang menyala. Pada sebuah ketetapan ilmu teknik digital, sebuah mesin hanya bisa memberikan komunikasi berupa menyala/*on* (1), dan mati/*off* (0).

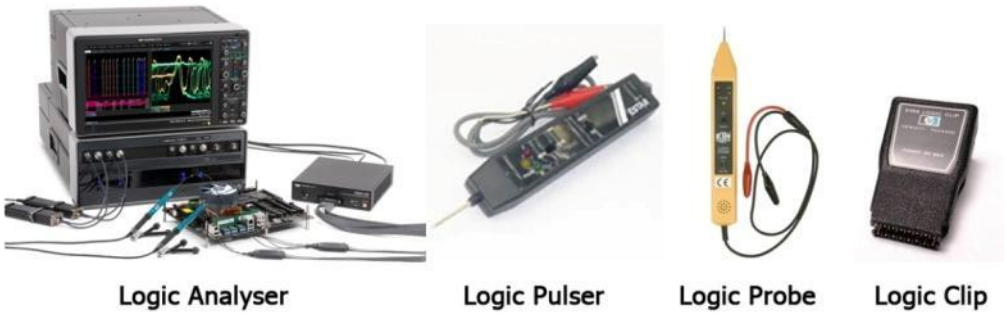
Piranti *LED seven segment display* sering digunakan untuk menayangkan data angka. Pada penggunaannya sering kali dalam bentuk berkelompok yang terdiri dari tiga, lima digit untuk membentuk sebuah formasi penayangan yang lengkap (Tooley, 2007:126).

Gambar 16. *LED 7 Segment Display*

Piranti *output LED seven segment* hanya digunakan sebagai alat bantu untuk memahami karakteristik sistem digital yang dioperasikan. Pada ilmu teknik digital, dalam menguji sebuah sistem digital terdapat berbagai jenis peralatan yang umum digunakan. Peralatan penguji sistem digital diantaranya adalah **(Gambar 16)**; *Logic Analyser, Logic Pulser, Logic Probe, Logic Clip* (Ibrahim, 1996:217).

Perangkat *logic analyser* digunakan untuk menguji sistem digital yang sangat kompleks seperti mikroprosesor yang memiliki transaksi data hingga 24bit, 32bit, 48bit. *Logic analyser* digunakan untuk mengamati *output* pada domain digital dan analog, seperti tertulis LAX_4K16 memiliki arti *memory* 4k dan *width* 16 bit. Sruktur dari ROM ke *logic analyser* tersusun dalam blok lain yang terbuat secara skematik (Panigrahy, 2014:4). *Logic pulser* digunakan untuk memicu satu per satu *I/O* sistem digital, sehingga akan terdeteksi apabila ada salah satu *logic gate* yang rusak. *Logic probe* sebagai alat yang digunakan untuk menguji *logic gate* dalam keadaan logika 1 atau logika 0, yang ditunjukkan melalui lampu indikator pada *logic probe*. Pada piranti *logic clip* berfungsi untuk mengetahui status logika pin *I/O* secara

serentak, dan status masing-masing pin akan ditunjukkan melalui lampu indikator.



Gambar 17. Peralatan Penguji *Logic Gate* Sistem Digital

4. *Virtual Learning (V-Learning)*

Teknologi *virtual learning* merupakan sebuah konsep pembelajaran yang merepresentasikan lingkungan kerja/belajar mendekati realitas suasana lingkungan nyata. Sarana dan berbagai peralatan belajar bersifat subtitutif dari peralatan dan suasana lingkungan yang sesungguhnya. Teknologi *virtual learning* dirancang untuk menciptakan kondisi belajar yang efektif dan efisien dalam segi ekonomis pengadaan peralatan dan lingkungan kerja/belajar. Karakteristik sebuah pembelajaran pada pendidikan kejuruan dan ketrampilan (vokasi), selalu dihadapkan pada pengadaan kondisi lingkungan dan peralatan belajar/praktikum yang mendekati suasana nyata pada lingkungan kerja (industri/laboratorium) yang sesungguhnya. Tuntutan akan realitas kondisi dan pengadaan peralatan yang benar-benar merepresentasikan kondisi dunia

kerja (industri) yang nyata tentu akan membutuhkan pendanaan yang besar bagi lembaga pendidikan sehingga akses belajar peserta didik terbatas.

Kasus yang demikianlah yang membuat banyak negara berkembang dan negara miskin lainnya selalu tertinggal dari aspek penguasaan ketrampilan seperti pada konsep pendidikan teknologi dan kejuruan (vokasi). Kita mempunyai pemikiran yang sama bahwa sebuah ketrampilan akan penguasaan teknologi pada SDM negara kaya dan negara miskin adalah sama, dan sama-sama mampu berdiri sejajar. Perbedaan dari kasus tersebut adalah, negara kaya mempunyai kecenderungan untuk mampu memenuhi fasilitas belajar mereka sehingga kita dapat mengatakan bahwa hal itu wajar ketika mereka terlihat lebih "mendahului" dan "maju" secara peradaban teknologinya. Hal ini tentu sama halnya ketika melihat seorang siswa pengendara mobil Ferrari terlihat lebih cepat dan lebih juara dibandingkan dengan seorang siswa pengendara sepeda kayuh Polygon ketika diuji kecepatan mereka keduanya pada lintasan sirkuit balap, atau seseorang akan lebih sukses menjalankan aksinya ketika menggunakan senjata RPG dan mobil tank baja, dibanding dengan seorang yang hanya menggunakan bambu runcing atau batu kerikil.

Sebuah konsep pemikiran untuk mengatasi permasalahan belajar pada pendidikan berbasis ketrampilan teknologi tersebut, maka lahirlah sebuah teknologi berkonsep *virtual learning*. Produk dari sebuah konsep *virtual learning* adalah pembuatan sebuah lingkungan pembelajaran virtual *(virtual learning environment)*. Pembuatan sebuah lingkungan belajar virtual

mengacu pada lingkungan kongkrit yang akan diimajinasikan (Weiss, 2006:1). Pengembangan sebuah lingkungan virtual yang relatif menyerupai kondisi lingkungan yang sesungguhnya pada era digital ini biasa disebut dengan *"virtual reality"* dan penggambaran realitas tersebut pada sebuah lingkungan untuk tujuan pembelajaran maka lazim disebut dengan *"virtual learning environments (VLE)"*.

> *The relatively recent development of the digital age has spawned interest in what has come to be called 'virtual reality' and in delineating what this means for learning and the creation of 'virtual learning environments'* (Weiss, 2006:1).

Teknologi *virtual learning environments (VLEs)* pada dasarnya adalah pembuatan lingkungan belajar secara virtual yang berbasis pada peralatan media belajar berupa komputer (Weiss, 2006:2).

Konsep *virtual learning* merupakan teknologi dari sebuah keilmuan induk bernama *Human-Machine Interface (HMI).* Teknologi *Human-Machine Interface* pada keperluan pembelajaran memiliki pengertian bahwa sebuah proses pembelajaran berbasis pada perangkat komputer, dengan semua aktivitas belajar dan instruksional pembelajaran dikemas dalam bentuk *software* (Ito, 2006:221). Merancang sebuah *software* interaksi antara manusia dengan mesin perlu dipertimbangkan aspek keselarasan antara yang diinginkan manusia dengan instruksi yang mampu dipahami mesin *(clearly intelligible),* sehingga tidak terjadi sebuah "*asymmetric relations*" antara manusia dengan mesin (Ito, 2006:236).

Acuan dari sebuah teknologi *virtual learning* adalah inovasi rekayasa dari lingkungan/suasana yang aslinya, menjadi bentuk manipulasi atau salinan

yang dapat dikendalikan. Realitas tiruan, selanjutnya dihidupkan pada sebuah *virtual reality*. Inovasi dari sebuah realitas virtual merupakan respon dari sebuah "*problematic of iconoclasm*", atau permasalahan atas pedoman dan kebiasaan yang telah dilakukan (Žižek, 2006:1550). Teknologi realitas virtual ditandai dengan pengalaman penggunaan perangkat sensor dan *electrical signal*. Perangkat mesin berbasis digital menghidupkan tampilan simulasi dalam pengalaman nyata berdasarkan kenyataannya. perangkat komputer *mobile* dengan konekti internet.

> *...virtual reality marks the radical reduction of the wealth of our sensory experience to-not even letters, but-the minimal digital series of 0 and 1, of passing and non-passing of the electrical signal. On the other hand, this very digital machine generates the "simulated" experience of reality which tends to become indiscernable from the "real" reality, with the consequence of undermining the very notion of "real" reality - virtual reality is thus at the same time the most radical assertion of the seductive power of images* (Žižek, 2006:1550).

Teknologi virtual merupakan representasi dari fungsi sebuah kondisi yang asli yang mengacu pada konsep *"screening the real"*. Makna *"screen"* mengacu pada penyajian tampilan dari salinan kondisi yang nyata, sehingga melahirkan sebuah *"desert of the real"* yang dapat dinikmati (Žižek, 2006:1556). Sejak ketika pembelajaran berbasis lingkungan virtual - *"virtual learning environments (VLEs)"* dianggap penting dan terus terjadi banyak penelitian yang mengangkat tentang kemungkinan sebuah pembelajaran virtual. Hal tersebut menuntun populernya sebuah istilah yang disebut dengan *"blended learning"*, dimana pembelajaran virtual dan pembelajaran *"face-to-face"* dikombinasikan (Gillespie, 2007:3). Untuk peserta didik dalam lingkungan

belajar, sebuah pengalaman pembelajaran virtual digunakan sebagai pelengkap pada strategi pengajaran *"face-to-face".*

> *... Since then VLEs have become increasingly important and more and more research has been undertaken into the educational possibilities of virtual learning. This has also led to the rise of the term 'blended learning' where virtual and face-to-face learning are combined* (Gillespie, 2007:3).

Pembuatan lingkungan virtual dalam bentuk simulasi yang berdasarkan pada kondisi yang nyata *(real world),* karena kondisi yang nyata mahal dalam segi pendanaan, waktu yang terbatas, serta tingkat resiko bahaya ketika dilakukan dalam kondisi nyata. Desain dari semua lingkungan virtual benar-benar merepresentasikan kondisi yang nyata serta bersifat *"actual system"* dalam kinerjanya. Pembuatan simulasi dalam lingkungan virtual dalam bentuk 2D cenderung kurang mewakili informasi sudut pandang secara keseluruhan. Bentuk lingkungan virtual dalam 3D lebih kompleks dan lebih bersifat *"understandability"* ketika simulasi dijalankan (Louloudi, 2013:106-107).

Pembelajaran virtual merupakan pembuatan sebuah kondisi lingkungan yang mempertemukan peserta belajar dengan obyek belajar secara interaktif. Penyajian tampilan dengan mengangkat bentuk ekologi dari lingkungan nyata yang akan dibuat dalam bentuk lingkungan virtual (Siddiqi, 2010:110). Sifat dari dunia virtual tiga dimensi (3D) memungkinkan terjadinya aktivitas dan interaksi, dan memungkinkan dapat disediakan dalam sebuah *platform* untuk aktivitas yang sulit dapat disediakan oleh lingkungan kelas pada umumnya. Lingkungan virtual didesain fokus untuk aspek

pengajaran yang bersifat sulit diajarkan dalam bentuk fisik yang sesungguhnya dan berbahaya (Thackray, 2010:139).

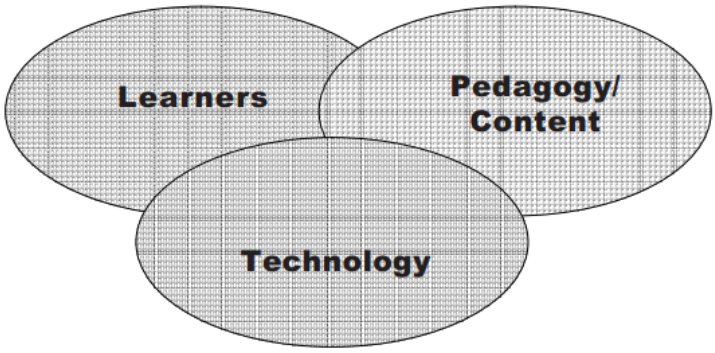
Teknologi *virtual learning* pada keperluan proses pendidikan akan mampu memberikan potensi manfaat serta peluang pada peserta didik untuk aktif, fleksibel, dan menambah keluasan pengalaman dalam belajar. Produk dari belajar virtual dapat digunakan sebagai sarana untuk menguji pemahaman dan sudut pandang peserta didik terhadap pelajaran/materi yang dikemas dalam bentuk *virtual learning environments*, mulai dari format yang sederhana seperti *Computer-Aided Instruction (CAI)* hingga pada format *complex virtual reality environments* atau realitas lingkungan virtual yang kompleks (Ainley, 2006:365).

Lingkungan virtual merupakan dunia kedua atau *"second life"* yang dapat dimanfaatkan untuk merancang sebuah *platform*, yang menyediakan desain kolaborasi sinkron bersifat efektif dan aktivitas belajar yang memberikan keleluasaan untuk kreatif dalam desain (Gu, 2009:159). Proses pengembangan desain lingkungan virtual 3D untuk pendidikan perlu menjaga adanya penekanan saat penggunaan dunia virtual 3D dengan desain lingkungan belajar, menyediakan kerangka edukasi, tutorial, serta masukkan pada perancang dan interaksi secara aktivitas berkelompok. Desain pembelajaran dalam dunia virtual mencakup penggunaan berbagai parameter desain interaksi, desain grafis, desain produk, serta dunia virtual 3D untuk membangun lingkungan belajar (Gu, 2009:162). Muatan edukatif merupakan *basic skills* (keterampilan dasar) dalam fitur lingkungan virtual,

scripting/programming pemodelan 3D dunia virtual dan komunikasi digital. Peserta didik menguasai "*basic skills*" maka selanjutnya adalah mengenali dan merasa *comfortable* (nyaman) dengan dunia virtual 3D, kerangka muatan edukatif dapat fleksibel dan *"responsive"* untuk mengakomodasi desain aktivitas level tinggi, konsultasi dan saran (Gu, 2009:176).

Pendekatan virtual *(virtual approach)* untuk *"Role-Play"* (pembelajaran berbasis simulasi), dapat ditambah nilai-nilai untuk penilaian pelatihan unjuk kerja dan keinginan untuk menyelidiki potensi ketrampilan tersebut (Morse, 2009:197). Perancangan kondisi belajar virtual *(virtual learning environment - VLE)* dalam pengajaran dan pembelajaran harus serupa dengan kondisi pembelajaran yang sesungguhnya *(real learning environment)*, serta perlu memperhatikan aspek warna, logo, tata ruang/tampilan, gaya huruf, ukuran huruf, dan gambar (Gillespie, 2007:75). Perancangan lingkungan belajar virtual akan bisa efektif dengan menganggap penting memahami keterampilan dan sikap peserta didik. Penilaian dan *"tracking"* pada pengembangan keterampilan dalam menggunakan perangkat pembelajaran virtual dianggap berguna (Gillespie, 2007:82). Teknologi *"virtual learning environments (VLEs)"* jenis apa saja pada dasarnya membutuhkan ketersediaan infrastruktur. Infrastruktur baik berupa *"hardware"* ataupun *"software"* yang secara utuh disediakan sekolah atau yang dimiliki secara personal saat pendidik memeberikan pelatihan (Gillespie, 2007:85).

Perangkat virtual untuk pembelajaran *(virtual machine)* dapat mensimulasikan semua komponen perangkat keras *(hardware)* seperti *memory*, *disk*, *CPU* dan sebagainya. Setiap simulasi mesin virtual dapat dengan bebas terpasang *(install)* pada sistem operasi *(operating system)*, aplikasi *(software)* berjalan secara simultan (Liao, 2008:27). Proses perancangan lingkungan pembelajaran virtual tidak lepas dari keterlibatan sebuah teknologi *"Machine Learning Technology"* , yang merupakan sebuah teknologi/sistem yang memiliki kecerdasan untuk mengolah data. Teknologi "*Machine Learning*" umumnya tersaji dalam bentuk perangkat lunak *(software)* yang mampu mengidentifikasi jika *"virtual machine"* yang dirancang/dibuat mengalami kondisi normal ataupun *"fault state"* (kegagalan sistem). Teknologi "*Machine Learning*" memiliki kemampuan mengolah data secara akurat pada kondisi *"virtual machine"* saat *"live"* , selain itu teknologi "*Machine Learning*" pembelajaran memiliki keakuratan tinggi dalam mengolah dan menguji data (Chang, 2010:91).



Gambar 18. Inter-relasi Elemen Keberhasilan Pembelajaran Virtual

Keberhasilan dalam proses perancangan sebuah lingkungan pembelajaran virtual dipengaruhi oleh tiga elemen; (Boehm, 2006:10):

1) *Learners*: Konsep ilmu pengetahuan peserta didik dan sifat menghargai atau menganggap penting terciptanya pembuatan pembelajaran virtual, adanya peningkatan pengalaman bersama perangkat belajar elektronik, dan faktor pentingnya membudayakan keterampilan pembeda antar individu yang lain; 2) *Pedagogy/Content*: Pengertian tentang bagaimana cara melakukan penyesuaian perihal muatan/materi pembelajaran yang dibutuhkan para peserta didik, termasuk penetapan tujuan pembelajaran dan hasil dari pembuatan lingkungan pembelajaran virtual (peningkatan kecakapan penggunaan media, analisa permasalahan, dan penerapan keterampilan) dan menentukan metode instruksional efektif untuk mencapai tujuan pembelajaran; 3) *Technology*: Penerapan solusi *"user-friendly"* untuk penggunaan teknologi tinggi pembelajaran, menciptakan dimensi interaksi sosial yang efektif, pembelajaran yang berkesinambungan *(sequencing)* meski peserta didik tanpa memiliki keahlian secara teknis.

Produk dari sebuah konsep *virtual learning* adalah apa yang disebut dengan *virtual laboratory* (laboratorium virtual). Teknologi *virtual laboratory* dapat memungkinkan pengguna untuk membuat dan melakukan berbagai percobaan menggunakan konsep intuisi dan teknik dari dunia yang sebenarnya. Pada *operating system* (sistem operasi) menuntun pengguna merasa dalam sebuah lingkungan komputasi, sebuah *virtual laboratory* memastikan persepsi pengguna dalam lingkungan dimana tempat berbagai

simulasi percobaan tersebut dilakukan. Sebuah *virtual laboratory* merupakan kesejajaran dari dunia yang sesungguhnya, merupakan sebuah *"playground"* dan tempat melakukan berbagai eksperimen, mengenali dan mengoperasikan berbagai peralatan, mendapatkan "*reference book*". (Lindenmayer, 2004:193).

Secara teknis *virtual laboratory* merupakan sebuah *"microworld"* dimana tempat melakukan berbagai eksplorasi yang dipandu oleh sistem. Istilah *"microworld"* diberikan pada lingkungan interaktif untuk membuat dan melakukan berbagai simulasi ujicoba. Panduan dapat berupa dalam bentuk *"tradisional book"*, tetapi dokumen elektronis lebih pantas untuk dimasukkan dan dijadikan satu dengan *"microworld"*. Sebuah teknologi *virtual laboratory* dapat dibagi menjadi dua komponen: berupa program aplikasi *(software)*, *data file*, atau berupa *support system* yang disajikan dalam *framework* yang *domain-dependent* (sebuah bentuk *operating system/OS* yang mandiri) (Lindenmayer, 2004:194).

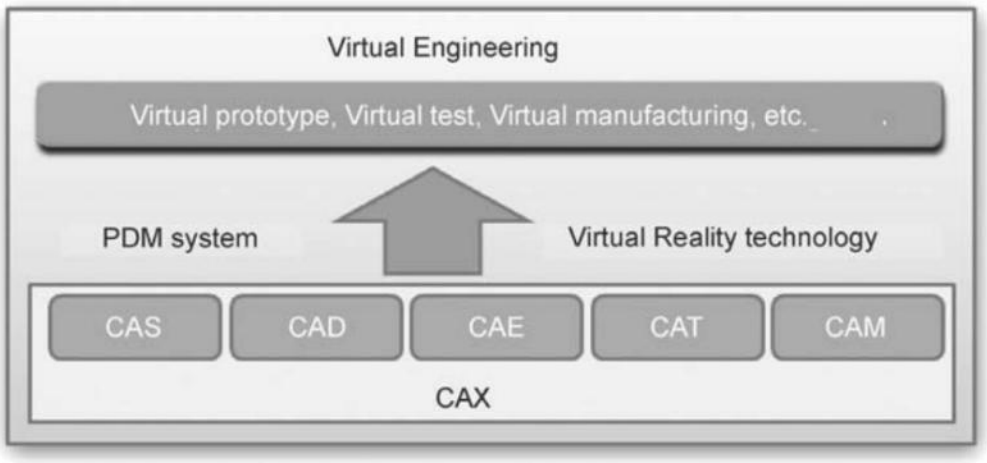
Pembelajaran virtual menyediakan konten dengan tujuan ilmu pengetahuan yang asli *(authentic science)*, menyediakan peralatan ilmiah virtual, dan memberikan akses pada lingkup materi-materi referensi. Sebuah tantangan bagi peserta didik adalah untuk mengoperasikan pada dunia simulasi tersebut kedalam pedoman untuk pencapaian rangkaian tujuan pembelajaran, dan selanjutnya melakukan pembelajaran untuk memikirkan dan merealisasikan selayaknya penyelesaian permasalahan ilmiah (Slator, 2002:1). Lingkungan pembelajaran virtual dalam berbagai disiplin ilmu,

bersifat membangun kesepakatan untuk kumpulan prinsip-prinsip keseragaman yang memberikan desain pedoman serta berbagi dan memperkenalkan sebuah teknologi maupun sebuah pendekatan. Penelitian dan pengembangan teknologi lingkungan pembelajaran virtual merupakan sebuah upaya pengembangan inovasi baru dalam media instruksional, dan upaya mengeksplorasi strategi yang terbatas (Slator, 2002:2).

Penelitian dan pengembangan sebuah lingkungan pembelajaran virtual ada yang terkait dengan penguraian sebuah perancangan tentang *"Virtual Tools"* yang mana sebuah desain rancangan dibuat dalam bentuk *software* yang berisi fitur tentang keahlian pembuatan dunia virtual untuk sebuah perintah/penugasan yang sedikit campur tangan dan kekeliruan profesional komputasi. Sebuah perangkat lunak yang dibuat dapat memuat berupa penyusunan hirarki abstraksi, kerangka konsep, *object interface* (antarmuka obyek), pemetaan virtual, petunjuk sikap, dan lainnya (Slator, 2002:34).

Teknologi *Virtual Learning Environments (VLEs)* dapat digunakan pula sebagai alat penilaian dalam pembelajaran. Alat penilaian dalam *virtual learning environments* seringkali digunakan mengenai terbatas pada penilaian ketrampilan dasar dan bidang kognitif level bawah. Penilaian dalam *virtual learning environments* dapat diterapkan sebagai *"resource"* untuk lingkup pelajaran yang lebih luas maupun lingkup penggunaan khusus untuk penilaian. Penilaian belajar peserta didik menggunakan alat *virtual learning environments*, hal ini sama dengan membuat *cost-effective* (biaya efektif) yang digunakan pihak terkait (Fuller, 2004:107).

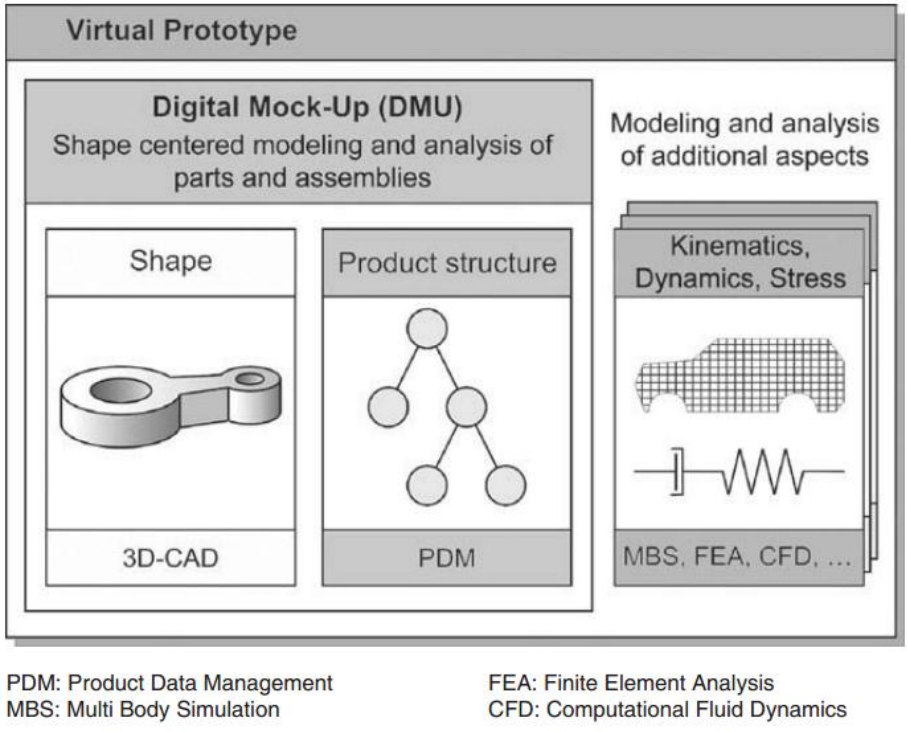
Konsep pembelajaran virtual salah satunya dapat disajikan dalam bentuk *teknologi VE (Virtual Engineering)*. Sebuah teknologi *teknologi VE (Virtual Engineering)* memiliki perbedaan dengan teknologi *VR (Virtual Reality)*. Istilah perangkat lunak *VE (Virtual Engineering)* di industri otomotif cina, tren tersebut muncul untuk meramu teknologi VR dan teknologi *computer-aided* yang lain seperti *CAE Analysis* atau *Simulation Analysis* (Jiang, 2011:171). Teknologi *VE (Virtual Engineering)* terintegrasi bersama dengan teknologi CAx, untuk menyelesaikan sebuah *virtual prototype, virtual test, virtual manufacture* dan lainnya (Jiang, 2011:172).



Gambar 19. Konsep Teknologi *VE (Virtual Engineering)*

Proses perancangan sebuah lingkungan pembelajaran virtual *(virtual learning environments)* dilakukan dengan pembuatan prototipe atau produk yang akan dibuat kedalam bentuk tiga dimensi (3D). Sistem 3D-CAD merupakan basis pembuatan model produk bangun/bentuk. Perincian produk bangun/bentuk dan proses pemasangan diwujudkan oleh "*product structure"* (struktur produk). Oleh karena itu proses ini diakomodasi oleh sebuah

"Product Data Management (PDM)". Produk bangun/bentuk yang sendirian dan dalam penggabungan bersama struktur produk untuk pengembangan *shape-based design* (desain bangun dasar) produk tersebut, apa disebut dengan *Digital Mock Up (DMU)*. Tampilan dari DMU merupakan gambaran lingkup komposisi semua komponen dan pemasangan produk tersebut. Tampilan DMU dapat digunakan pula sebagai sarana percobaan seperti mendeteksi ketidakseimbangan/ketidaksesuaian, pengecekan proses perakitan dan urutan pembongkaran. Untuk proses analisa tingkah laku pada sistem, harus dipikirkan penambahan *"additional aspects"* (aspek tambahan) lain yang ditawarkan oleh sebuah *"virtual prototype"* (Gausemeier, 2011:4).



Gambar 20. Konsep Pemodelan Benda untuk Prototipe Virtual
(sumber: *"Design and VR/AR-based Testing of Advanced Mechatronic Systems"* Jürgen Gausemeier)

Sebuah *"virtual prototype"* merupakan integrasi antara DMU dengan *"additional aspects"*. Prototipe virtual tidak hanya berupa sebuah bangun/bentuk namun memiliki fitur fungsional dan tingkah laku atau mekanisme kinerja. Meskipun prototipe virtual tidak dapat menggantikan secara utuh sifat-sifat prototipe nyata, ini merupakan kontribusi ilmiah secara waktu yang efektif dan biaya pengembangan yang lebih minimal untuk proses perwujudan produk secara komplek dan keseluruhan (Gausemeier, 2011:5).

Keuntungan menggunakan VR (*virtual reality)* dalam pendidikan dan pelatihan terkait dengan kemampuannya untuk memungkinkan siswa berinteraksi satu sama lain dalam lingkungan virtual tiga dimensi. Pengertian intuitif tentang subyek pembelajaran juga dapat dikembangkan dengan berinteraksi dengan objek, pesan terhubung dan isyarat di lingkungan virtual. Berbeda dari pendekatan pendidikan dan pelatihan konvensional, seperti pemanfaatan gambar statis atau gambar dua dimensi, representasi visual VR memungkinkan lebih banyak *degrees of freedom* (DoF)/derajat kebebasan untuk diintegrasikan (Wang, 2018:2). Meskipun berbagai metode penilaian pelatihan didasarkan pada *virtual reality* dapat ditemukan dalam literatur, penting untuk menyoroti bahwa pilihan metode penilaian tergantung pada jenis pelatihan, terutama pada variabel yang dapat diukur selama pelaksanaan pelatihan (Moraes, 2012:11). Teknologi komputer berkembang dengan cepat. CIM, CSCW, berkembang menjadi CVE *(Collaborative Virtual Environments)* dengan evolusi teknologi 3D/4D CAD dan VR, yang memiliki format *environments* berupa teknologi *multiple users* (Wang, 2011:107).

5. *Mobile Learning (M-Learning)*

Definisi dari *mobile learning* adalah sebuah pembelajaran yang menggunakan perangkat teknologi *mobile/handheld* (ponsel, PDA, *smartphone, pocket-PC, tablet-PC,* dsb), tentunya menggunakan berbagai fitur dan konten perangkat *mobile* itu sendiri yang bersifat edukatif. Hal ini sependapat dengan definisi berikut ini, yang mengatakan bahwa;

> *Mobile learning, using a mobile device to access and study learning materials and for communicating with the institution, tutors and fellow students,* (Ally, 2009:287).

Pada sumber lain mengatakan bahwa *mobile learning* merupakan pembelajaran yang tidak terbatas pada ruang kelas dan dapat berlangsung dimana saja dengan dukungan media belajar berupa perangkat elektronik yang selalu tersedia di saku maupun dalam genggaman tangan, yaitu berupa *mobile phones (cellphones/handphones), smartphones, palmtops, handheld computers (Personal Digital Assistants / PDA), Tablet PC.* (Kukulska, 2005:1). Pembahasan teori dan praktik teknologi pembelajaran dalam bentuk bahan ajar berupa *mobile learning* dibutuhkan beberapa kemampuan yang mendukung. Kemampuan tersebut berhubungan dengan kemampuan untuk analisis kurikulum mulai dari kompetensi dasar, bahan ajar, analisis konten atau materi, topik bahan materi yang akan dibuat hingga SAP dan silabus atau RPP.

Kemampuan lain yang harus dimiliki adalah kemampuan menganalisis bahan ajar buku, serta kemampuan untuk menganalisis ketersediaan dan daya dukung dari media pembelajaran modern, khususnya

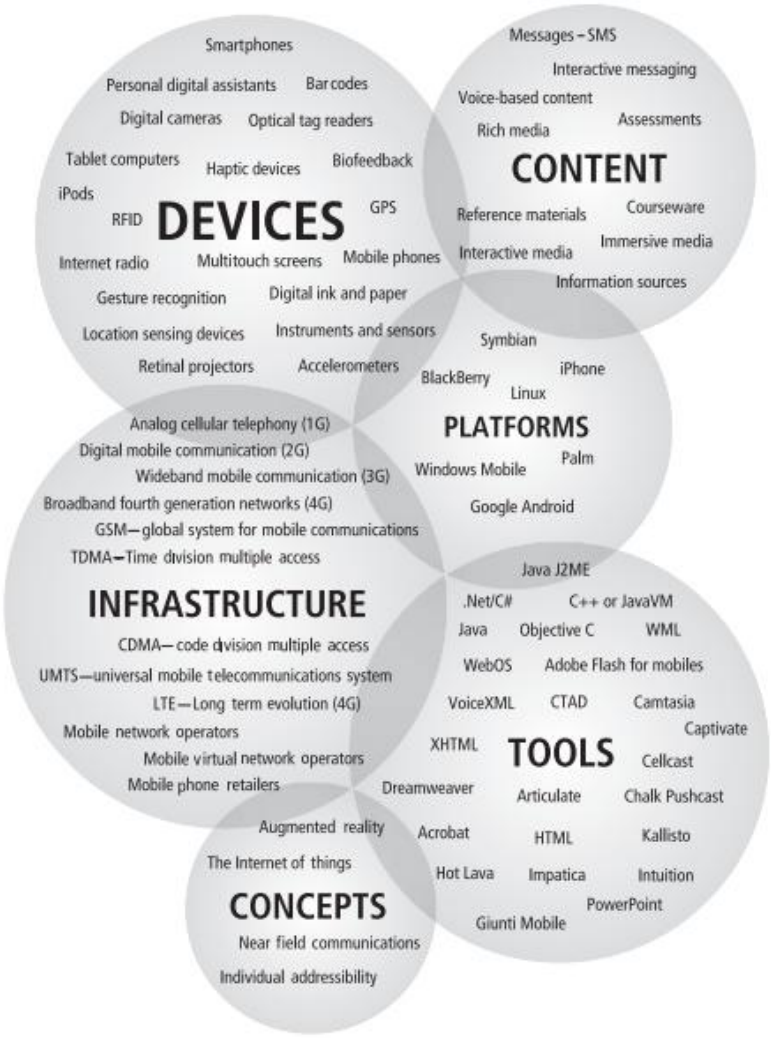
media berbasis ICT. Di sisi lain, kebutuhan awal yang harus dimiliki adalah kemampuan untuk melakukan instalasi *software* yang mendukung pembuatan bahan ajar *mobile learning* baik secara *online* maupun *offline* (Darmawan, 2012:91). Menurut sumber lain (Madjarov, 2011) mengartikan bahwa *m-learning* adalah representasi dari *e-learning* yang disajikan dalam perangkat komputer *mobile* dengan konekti internet.

> *The authors believe that m-Learning can be presented as a mobile extension of e-Learning through mobile computa tional devices with Internet connectivity*, (Madjarov-Boucelma, 2011:259).

Definisi lain menurut Woodill (2011), menjelaskan bahwa *mobile learning* merupakan aktivitas belajar yang melalui media/perantara dari perangkat-perangkat *mobile (mobile device)*, serta tentang semua teknologi yang ada pada perangkat tersebut, (Woodill, 2011:14). Kelebihan lain dari cara belajar berbasis pada perangkat *mobile* ini adalah "*just in time, just enough, and just for me*." Demikian pula sifatnya yang menurut Woodill (2011:24) diantaranya adalah;

- *Portability*
- *Any time, any place connectivity*
- *Flexible and timely access to e-learning resources*
- *Immediacy of communication*
- *Empowerment and engagement of learners, particularly thosein dispersed communities*
- *Active learning experiences*

Secara garis besar woodill mengklasifikasikan berbagai keanekaragaman perangkat *mobile (mobile device)* menjadi sebagai berikut;



Gambar 21. Klasifikasi Perangkat *Mobile*
(sumber: *"The Mobile Learning Edge: Tools and Technologies for Developing"* Gary Woodill, 2011)

Penjelasan dari klasifikasi perangkat *mobile* sebagai media belajar **(Gambar 21)**, dapat dijelaskan sebagai berikut. Berdasarkan pada bentuk dan anatomi *hardware (device)* perangkat *mobile* terdiri dari banyak varian diantaranya berbentuk *PDA*, *Tablet computer*, *Mobile phone*, *Smartphone*, *iPods*, dll. Berdasarkan konten atau isi dari fitur layanan yang disediakan pihak vendor, perangkat *mobile* memiliki berbagai bentuk varian, diantaranya perangkat yang hanya miliki fasilitas *massage (sort massage/SMS)*,

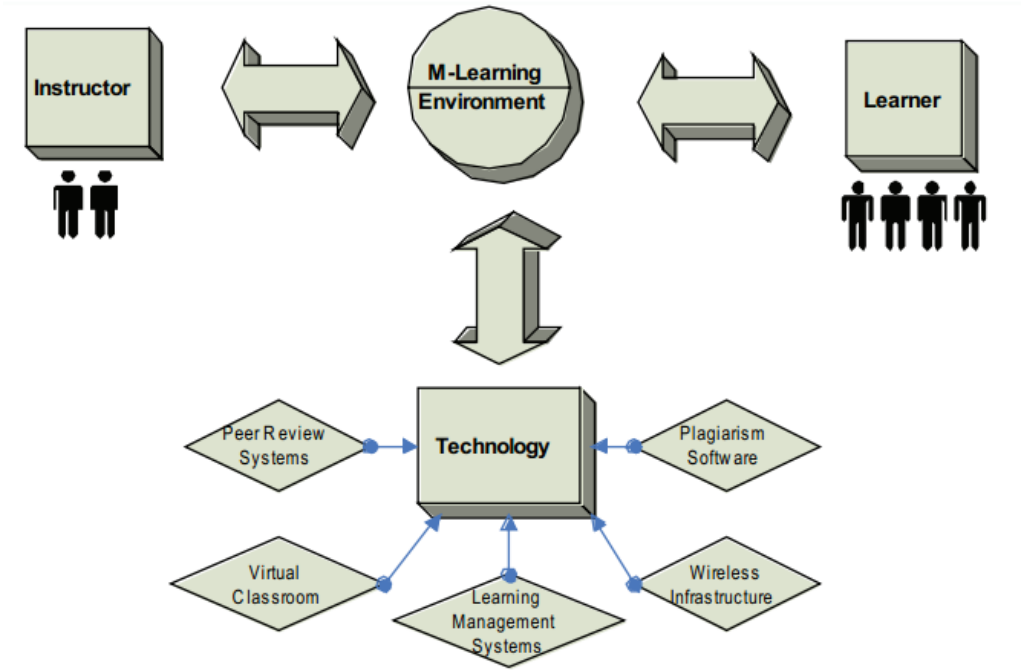
interactive massage (chat), voice-based content (phone call), course ware atau fitur aplikasi untuk *office* (kegiatan perkantoran).

Berdasarkan jenis *platform*/*operating system (OS)*, ada berbagai bentuk varian untuk saat ini diantaranya; *Symbian, Palm, Linux, BlackBerry, Windows Mobile, iPhone, dan Android*. Berdasarkan infrastruktur atau sistem lalulintas data antar perangkat pada jaringan terdapat berbagai bentuk untuk saat ini, antara lain; *1G, 2G, 3G, 4G (LTE), 5G, GSM, CDMA*, dll. Berdasakan tools atau bahasa instruksi program pada *software* ada berbagai jenis diantaranya; *C++, Java, XHTML, WebOS, Adobe Flash mobile, WML, CTAD*. Berdasarkan konsep penggunaan pada sisi *user* terdiri dari berbagai tujuan, untuk keperluan pengalamatan individu *(phone/SMS)*, untuk terkoneksi pada jaringan internet *(browser)*, untuk berbagai keperluan *augmented reality* dan *intertainment* (video, foto, musik, game, film, dll).

Pembelajaran model *M-Learning* (*mobile learning*) memang diyakini dapat memberikan karakter pembelajaran yang efektif dalam memfasilitasi pendidik dan peserta didik pada proses dan aktivitas pembelajaran.

> *The authors believe that ICT tools facilitate such learning activities and provide support for the learner and the instructor to engage in such activities in ways that facilitate effective mobile learning and teaching environments. The key components of those environments are the learner, the instructor, the technology and the processes involved in the creation of such environments* (Raj, 2009:25).

Komponen kunci pada model lingkungan pembelajaran *M-Learning* adalah peserta didik, pendidik, teknologi yang dilibatkan dalam proses pembelajaran dalam rangka untuk menciptakan lingkungan belajar yang sesungguhnya.



Gambar 22. Berbagai Komponen dalam Lingkungan *M-Learning*

Pembelajaran berbasis *handheld device* atau *mobile learning* digolongkan dalam 6 kategori berdasarkan fungsinya (Song, 2009:302): *educational communication*, *managing*, *multimedia access*, *game & simulations*, *data collection*, dan *context-aware applications*. Penggunaan perangkat *handheld* (komputer genggam) berupa pembelajaran simulasi dalam kelas sebagian besar merasa praktis. Metode simulasi pada perangkat *handheld* dapat menyediakan peluang untuk peserta didik kepada pengalaman pembelajaran cara baru yang relatif, inovatif sebab cukup efektif menggunakan peralatan *handheld* yang *small-screen* pada ruang kelas. Penyajian simulasi dalam bentuk *microworld* perangkat *handheld* cukup lebih representatif dan praktis dibandingkan dengan penggunaan desktop atau laptop yang memiliki layar besar dan konsumsi energi yang besar pula. Penggunaan simulasi pada proses

pendidikan dirasa lebih berguna dan cukup membantu aktivitas pembelajaran daripada "*physical settings"* (Song, 2009:312).

Pembelajaran berbasis *mobile learning (m-learning)* berkembang tanpa teori dan model, yang mana hal tersebut dikembangkan *bottom-up* oleh para praktisi, dan beberapa ditingkatkan *top-down*, sama seperti material pada *e-learning* dan sengaja diterapkan untuk penggunaan *mobile*. Akses yang bersifat interaktif dan multimedia pada perangkat *mobile* menawarkan peluang yang baik, sebuah potensi, untuk sebuah cara belajar baru (Roland, 2012:221). Pembelajaran menggunakan perangkat *mobile* dapat dilakukan pula dalam rangka aktivitas pembelajaran *work-base learning*, keperluan merekam gagasan dan ide sama seperti saat anda melakukannya pada tempat kerja (Janet, 2010:34). Desain pembelajaran pada *m-learning*, tidak sama seperti sistem pembelajaran statis berbasis komputer, sebab *m-learning* dirancang sedemikian rupa untuk memudahkan pengguna menghentikan dan memulai kembali pembelajaran atau mode episodik (Ryu, 2009:11).

Pengembangan dari revolusi dalam sistem dan teknologi *mobile* membuat peningkatan pada banyak hal. Informasi yang berkembang mengenai persoalan dari individu, industri, dan akademisi dan beberapa pihak membatasi dan menghimbau untuk lebih sadar dalam menggunakan perangkat *mobile* mengenai perihal lokasi, akses dan kepentingan umum. Himbauan yang ditawarkan untuk potensi mengurangi *worst effects* (berbagai dampak buruk) dari informasi yang telah berkembang begitu pesat (Hu, 2010:26-27).

6. *Android Application*

Definisi dari kata *android* disini adalah mengarah pada salah satu dari sekian banyak jenis sistem operasi (*operating system / OS*) yang "berjalan" pada perangkat *handheld* (komputer genggam) dan perangkat *mobile* seperti; ponsel/*handphone*, *smartphone, tablet-PC* dan sejenisnya. Android merupakan sebuah desain dari sistem operasi *(platform)* terbuka *(open source)* yang komprehensif untuk perangkat *mobile* (Nakamura, 2014:1). Sejarah sistem operasi android berawal dari tahun 2005 sebuah perusahaan bernama Google membeli perusahaan Android, Inc. Tahun 2007 sistem operasi tersebut dikembangkan Open Handset Alliance menjadi sebuah sistem operasi yang terbuka *(open source).* Tahun 2008 Android 1.0 memasuki pasar ponsel dunia, dengan penggunaan perangkat keras yang diproduksi oleh HTC Corporation. HTC merupakan produsen perangkat telekomunikasi asal kota Taoyuan, Taiwan. Sistem operasi android hingga saat ini telah diproduksi dan ditanam *(embed)* pada berbagai perangkat *mobile phone*, *tablet PC*, *TV*, *home automation systems*, kontrol *dashboards* mobil, dan *navigation systems* (Nakamura, 2014:3).

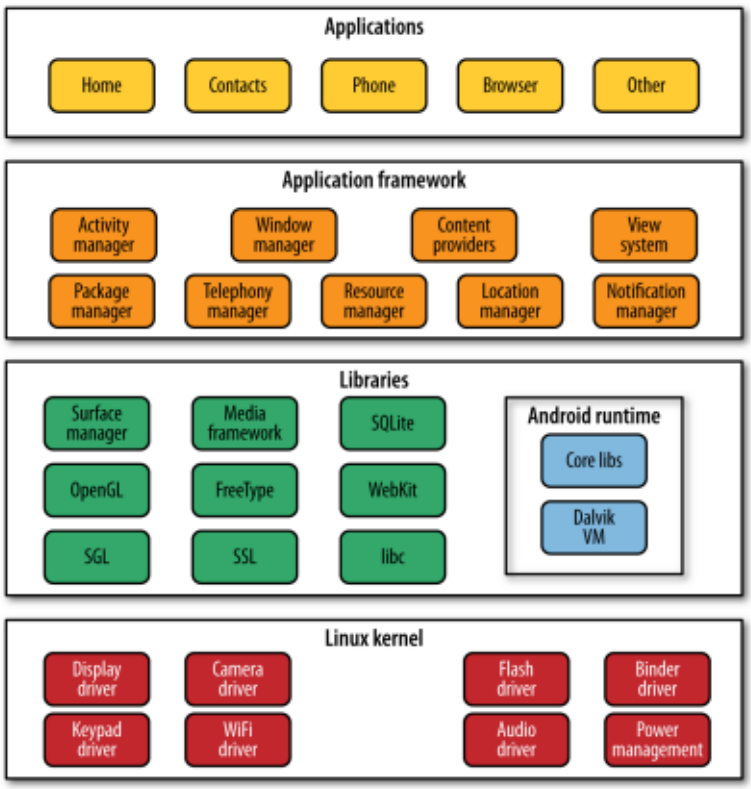
Sistem operasi android seperti halnya kue yang terdiri dari berbagai macam lapisan. Setiap lapisan memiliki karakteristik dan fungsi tertentu. Android dibangun diatas sistem operasi Linux. Sistem operasi Linux dipilih sebagai basis dasar tumpukan *(stack)* Android, karena Linux memiliki sisi kelebihan dalam hal *portability*, *security*, dan *features* (Nakamura, 2014:31). Faktor *portability* memiliki pengertian bahwa sistem operasi Linux memiliki

tingkat fleksibilitas untuk dapat ditanam pada berbagai bentuk perangkat keras *(hardware architectures).*

Komponen perangkat lunak pada Linux sebagian besar ditulis dalam *C code* (bahasa C) yang sederhana, sehingga memiliki kemudahan pula dalam terhubung ke berbagai jenis perangkat *(device)*. Faktor *security* pada Linux merupakan menjelaskan bahwa sistem operasi Linux memiliki tingkat keamanan sistem yang tinggi. Android bertumpu pada sistem operasi Linux secara langsung memiliki tingkat keamanan berupa ijin akses sistem pada Android *(Android permission)*, hal ini berbeda dengan sistem operasi berbasis Java *(Java-based mobile platform)*. Faktor *features* pada Linux memiliki arti bahwa sistem operasi Linux sangat memberikan berbagai fasilitas yang bermanfaat. Sistem operasi Android sehingga memiliki berbagai fitur seperti kesesuaian pada piranti memori, manajemen daya listrik, terhubung pada jaringan internet dan radio frekuensi (Nakamura, 2014:32).

Sebagian besar vendor *smartphone* memproduksi sistem operasi berbasis Android. Sistem operasi Android menyediakan *platform* terbuka *(open source)* bagi para pengembang sehingga menjadikan sistem operasi ini banyak digunakan oleh para pengembang perangkat lunak *(software-builder)*. Sistem operasi Android memberikan pengalaman *smartphone* inti, tetapi sebagian besar pengalaman pengguna bergantung pada *third-party applications.* Android memiliki banyak pasar tempat pengguna dapat mengunduh aplikasi pihak ketiga yang memungkinkan akses mudah ke jejaring sosial, *game*, dan banyak lagi. Seperti halnya aplikasi desktop

tradisional, ada kebutuhan untuk melindungi pengguna dari aplikasi jahat dan melindungi pengembang dari *plagiarists* yang ingin mendapat manfaat dari kerja keras pengembang yang sah (Crussell, 2012:37).



Gambar 23. Struktur Sistem Operasi Android
(sumber: *"Learning Android – 2nd edition"* Masumi Nakamura)

Android *application* merupakan sebuah perangkat lunak *(software)* yang beroperasi hanya pada *operating system (OS)* Android, dan berfungsi untuk menjalankan berbagai fungsi sepeti membuat catatan *(editing a note)*, memutar musik *(playing a music file)*, membunyikan alarm *(ringing an alarm)*, atau membuka kontak telepon *(opening a phone contact)* (Steele, 2011:23). Buku Wei-Meng Lee, mendefinisikan bahwa;

> *Android is a mobile operating system that is based on a modified version of Linux. It was originally developed by a startup of the same name, Android, Inc. In 2005, as part of its strategy to enter the mobile space, Google purchased Android and took over its development work (as well as its develop-ment team).* (Lee, 2012:2)

Android merupakan sebuah *mobile operating system* yang telah dimodifikasi dari dasar sistem operasi Linux. Aslinya android dikembangkan oleh nama yang sama yaitu Android Inc. Pada 2005 merupakan bagian dari strategi untuk memasukkannya pada *mobile space* adalah Google membeli Android dan mengambil alih untuk pengembangannya. Android telah mengalami berbagai perkembangan semenjak pertamakali di *release*, berikut adalah beberapa generasi sistem operasi android hingga saat ini;

Tabel 7. Generasi Sistem Operasi Android

Android Version	Release Date	Code Name
1.1	9 Februari 2009	
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0/ 2.1	26 Oktober 2009	Eclair
2.2	20 Mei 2010	Froyo
2.3	6 Desember 2010	Gingerbread
3.0/3.1/3.2	22 Februari 2011	Honeycomb
4.0	19 Oktober 2011	Ice Cream Sandwich
4.1	Juli 2011	Jelly Bean
4.2	November 2012	Jelly Bean
4.3	Juli 2013	Jelly Bean
4.4	Oktober 2013	Kit Kat
5.0	November 2014	Lolipop
6.0	5 Oktober 2015	Marshmallow
7.0	22 Agustus 2016	Nougat
8.0	Agustus 2017	Oreo

Android adalah sistem operasi *open source,* yang bisa *costumize* dengan mengkonfigurasikan *hardware* dan *software*. Menurut Lee (2012:3), Android memiliki beberapa fitur berikut ini; (1) *storage*, menggunakan SQLite, relational database; (2) *connectivity*, supports GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluethooth, WiFi, LTE, dan WiMax; (3) *messaging*, supports SMS dan MMS; (4) *web browser*, didasarkan pada open-source WebKit bersama dengan Chrome's V8 JavaScript engine; (5) *media support*, termasuk H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, dan BMP; (6) *hardware support*, akselarasi sensor, kamera, digital kompas, *proximity sensor*, dan GPS; (7) *multi-touch*, supports multi-touch screens; (8) *multi-tasking, supports multi-tasking applications*; (9) *Flash support*, Android 2.3 supports Flash 10.1.; (10) *Tathering, support* koneksi internet.

Struktur aplikasi android terdiri dari berbagai lapisan (Lee, 2012:4), diantaranya adalah; (1) *Linux kernel* – merupakan lingkup inti android untuk memuat berbagai *driver* perangkat yang ada pada *hardware* piranti *mobile*; (2) *Libraries* – lapisan untuk memuat semua kode pada berbagai fitur yang ada pada android, sebagai contoh *SQLite library* untuk penyimpanan data dan *WebKit library* untuk *web browsing*; (3) *Android runtime* – sebagai pencatat keluar masuk data dengan menggunakan bahasa pemrograman Java; (4) *Application framework* – sebagai pembuka sistem operasi android dalam menjalankan berbagai jenis aplikasi yang ada didalam sistem; (5) *Application*

– biasa ditemukan pada lapisan atas perangkat *mobile*, seperti berupa *Phone*, *Contact*, *Browser*, dll.

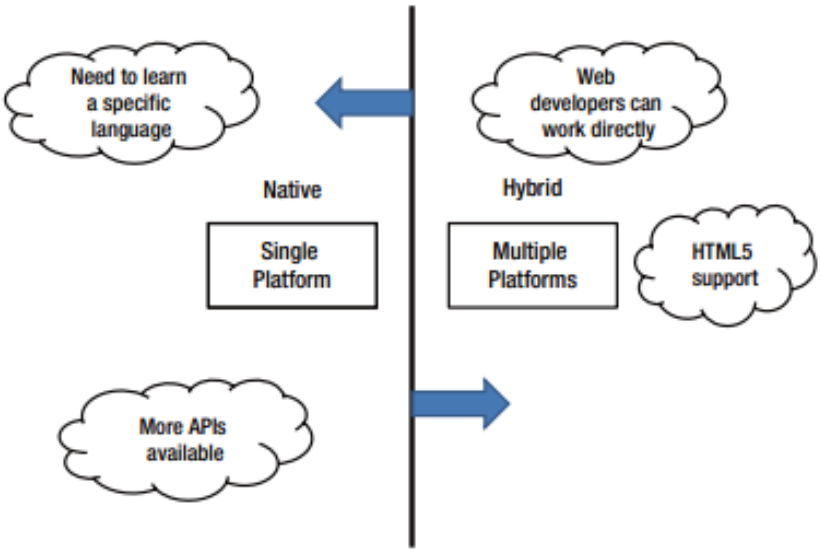
Pengembangan perangkat lunak *(mobile applications)* terdiri dari dua jenis (Panhale, 2016:15), diantaranya adalah;

- *Native Mobile Application Development (NMAD)*

Native mobile applications development, merupakan jenis aplikasi yang yang dikembangkan sesuai dengan *Operating System (OS)* yang ada pada perangkat *mobile* yang bersangkutan. Aplikasi jenis NMAD dibuat sesuai dengan teknologi perangkat keras yang telah dibuat oleh vendor, seperti halnya; Android (Google), iOS (Apple), Windows Phone (Microsoft). Aplikasi jenis NMAD telah dikembangkan sejak tahun 1973 oleh vendor Motorola. Saat itu aplikasi *mobile* hanya bisa digunakan untuk keperluan panggilan suara *(phone call)*. Tahun 1993 lahir sebuah pengembangan aplikasi berbentuk *Sort Message Service* (SMS), sebuah aplikasi untuk layanan pesan tulisan. Tahun 1996 muncul sebuah sistem operasi Palm OS yang ditanamkan pada perangkat *Personal Digital Assistant* (PDA). Aplikasi yang dioperasikan dalam Palm OS beragam fitur, diantaranya seperti kalkulator, kalender, buku catatan. Pada tahun yang sama lahir pula sistem operasi buatan Microsoft, yaitu *Windows Embedded Compact* (Windows CE). Tahun selanjutnya berkembang sebuah sistem operasi bernama Symbian, yang merupakan lisensi dari Nokia. Perkembangan aplikasi *mobile* berlanjut hingga tahun 2002 muncul sistem operasi BlackBerry, dan 2008 lahir Android yang ditanam pada perangkat keras merek HTC.

- *Hybrid Mobile Application Development (HMAD)*

Hybrid mobile applications development, merupakan aplikasi yang dikembangkan untuk mampu beroperasi pada berbagai perangkat *mobile* dan lintas *platform*, sebagai contoh HTML, CSS, JavaScript. Perkembangan HMAD berawal pada tahun 2008 oleh perusahaan BDI dan saat itu aplikasi ini diberi nama UIWebView. Aplikasi UIWebView sebagai sarana untuk menjembatani antara komunikasi bahasa Objective-C dan JavaScript. Tahun 2009 perangkat iPhone 2 memperkenalkan WebView dan dukungan basis data SQLite. Perkembangan selanjutnya muncul Adobe dengan memperkenalkan PhoneGap yang diperuntukkan pada sistem operasi iOS dan Android. Perkembangan HMAD hingga saat ini muncul dalam berbagai bentuk dan inovasi, seperti Adobe Flash, Microsoft Silverlight, Xamarin.



Gambar 24. Dasar Perbedaan NMAD dan HMAD
(sumber: *"Beginning Hybrid Mobile Application Development"*, Mahesh Panhale)

Aplikasi *mobile web* adalah aplikasi dengan *web-based* yang dioptimasi agar dapat berjalan dengan baik pada perangkat *mobile* (Pocatilu, 2006:103). Aplikasi kategori ini didesain dari aspek *user interface* maupun aspek fungsionalitasnya supaya dapat dijalankan secara maksimal pada perangkat *mobile*. *Mobile web* dikembangkan dengan teknologi web seperti HTML, JQuery dan JavaScript.

Sistem operasi Android memiliki perbedaan dibandingkan dengan sistem operasi seperti Symbian. Android memiliki banyak kelebihan seperti salah satunya, Android dapat dilakukan penulisan bahasa pemrograman melalui komputer desktop berbasis sistem operasi Windows, Linux, atau Mac OS X. Symbian hanya dapat dilakukan pengembangan bahasa penulisan program melalui komputer desktop berbasis sistem operasi Windows. Android dan Symbian namun memiliki beberapa kesamaan yaitu, keduanya memiliki kemampuan mengemas berkas *(file compilers)* dalam bentuk *binary*, dan memiliki format *fast-loading* dalam mengeksekusi data (Wilcox, 2009:244). Android hingga saat ini menguasai mayoritas pengguna yang semula didominasi oleh para pengguna Symbian. Android sebagai *platform* perangkat *mobile* mampu mendominasi karena memiliki berbagai kelebihan dibandingkan dengan kompetitor yang lain seperti Symbian, BlackBerry, iPhone, Windows Mobile, LiMo.

Tabel 8. Perbandingan Berbagai *Mobile Platform*

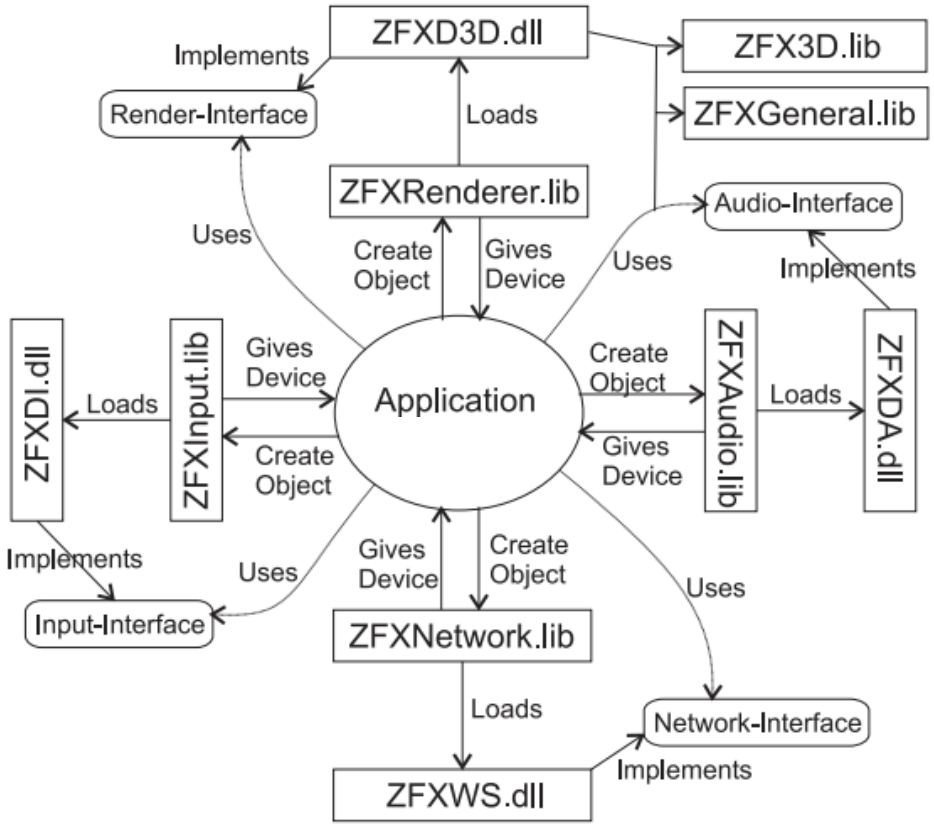
	Symbian Foundation	RIM BlackBerry	Apple iPhone	Windows Mobile	LiMo	Android
Market share	High	Medium	Medium	Medium	Low	Low
Source code licensing	Open and free	Closed, not externally licensed	Closed, not externally licensed	Shared,[a] per-unit royalties	Shared,[b] moderate entry cost	Open (gated) and free
Maturity	High	High	Medium	High	Medium	Low
Range of hardware	High	Medium	Very low	High	Medium	Low
Cross-platform APIs	Very wide support	Java only	Limited support	Wide support	Moderate support	Java only

7. *Graphics Engine*

Teknologi *graphics engine* merupakan teknologi untuk proses rekayasa grafis *(graphics engineering)* yang tentunya grafis pada ilmu komputer. Sebuah *graphics engine* berwujud sebuah perangkat lunak *(software)* yang mengkoordinir keperluan pembuatan grafis serta proses rekayasanya, umumnya menggunakan bahasa komunikasi komputer *(programming language)*. Bahasa perintah dalam ilmu komunikasi komputer ada banyak ragamnya, seperti: Java, C, C++, C#, PHP, SQL, Pascal, Python, Ruby dan sebagainya. Perangkat untuk proses merekayasa/memainkan grafis inilah yang umumnya lazim dikenal dengan istilah *"game engine"* . Pada kenyataannya aktivitas rekayasa grafis tidak hanya digunakan untuk pembuatan permainan grafis *(video game)*, namun setiap perusahaan pembuatan perangkat lunak *(software)* perekayasa grafis sering menamai produk perangkat lunaknya dengan istilah *"game engine"*.

Sebuah *engine* pada rekayasa grafis sudah termasuk didalamnya adalah fitur *3D engines, sound engines, input engines, network engines, physics engines, artificial intelligence (AI) engines,* dan sebagainya (Zerbst, 2004:5). Sebuah *graphics engine* memiliki cakupan yang cukup luas dan kompleks, sebab merupakan subyek dalam pembuatan aturan/tatanan pada sebuah desain *library* (kumpulan data) yang luas pada penggunaan *object-oriented programming* (pemrograman berorientasi obyek). Sebuah *graphics engine* memiliki cukup fitur untuk mengatur dan merekayasa obyek sebab ini merupakan esensi dari kumpulan inti dari pelayanan otomatis *(automatic services)* untuk para pembuat skrip perintah perangkat lunak *(application writers)* yakin (Eberly, 2005:105). Rekayasa grafis menyangkut proses pembuatan efek pada obyek, dan pembuatan sistem manajemen data *"front-end"* yang disajikan pada tampilan untuk diolah *"back-end" rendering system* (sistem eksekusi) (Eberly, 2005:1).

Struktur *engine* terdiri dari berbagai *compiler* dan *file builder* seperti DLLs, dan *static libraries (lib)*. Sebagai contoh dibawah ini merupakan struktur dari *engine* bernama ZFXEngine 2.0. (Zerbst, 2004:36). Sebuah *engine* tidak hanya berupa *"single compiled file"* untuk membangun sebuah aplikasi *(software),* namun terdiri dari empat *interface* dan dua *static libraries.* Sebuah aplikasi yang dibuat dapat menentukan modul mana saja yang akan digunakan secara bersama, atau bagian tertentu saja yang digunakan dengan melompati bagian lain yang tidak ingin dibutuhkan.



Gambar 25. Struktur *Engine*

Ada empat modul utama sebagai *interface* (antar muka) yang memiliki ekstensi DLL *(dynamic link libraries)*. Modul utama tersebut sebagai pemroses *render device, the network device, audio device,* dan *input device.* Struktur *engine* ditambahkan pula *static library* pada masing-masing keempat modul utama. Komponen *libraries* memiliki kegunaan sebagai *loading* dari DLL. Komponen *static libraries* sebagai penunjukan alamat obyek yang telah dibuat melalui tampilan *interface* (antar muka), kelas obyek yang telah dibuat berada pada DLL untuk dieksekusi.

Berikut ini penjelasan dari berbagai komponen lain yang menjadi bagian dari struktur *engine* (Zerbst, 2004:37):

- *ZFXRenderDevice Interface*

Merupakan pemroses data berkaitan dengan kontrol grafis dan *rendering* grafis secara sederhana, hal ini secara teknis bisa kita lihat pada bagaimana sebuah bahasa instruksi *(programming language)* bekerja untuk obyek yang dibuat. Proses berikutnya *ZFXRenderDevice Interface* akan bekerja pada perangkat lunak pihak kedua, seperti Direct3D, OpenGL, atau API *(Application Programming Interface)*.

- *ZFXInputDevice Interface*

Merupakan pemroses data berkaitan dengan kontrol *interface* (antar muka) perangkat *input* seperti *keyboard*, *mouse*, *mouse-write pen*, *joystick*, *touchscreen* yang digunakan oleh penguna.

- *ZFXAudioDevice Interface*

Merupakan pemroses data berkaitan dengan kontrol modul perangkat *audio.* Tanpa media *audio*, pengalaman multimedia pada aplikasi kurang begitu tepat. Karena modul ini akan memuat pula fitur efek suara.

- *ZFXNetworkDevice Interface*

Merupakan pemroses data berkaitan dengan modul yang memfasilitasi fitur *IP addresses, protocols,* dan koneksi LAN *(local area network).* Termasuk transaksi data pada koneksi antara *client* dan *server*.

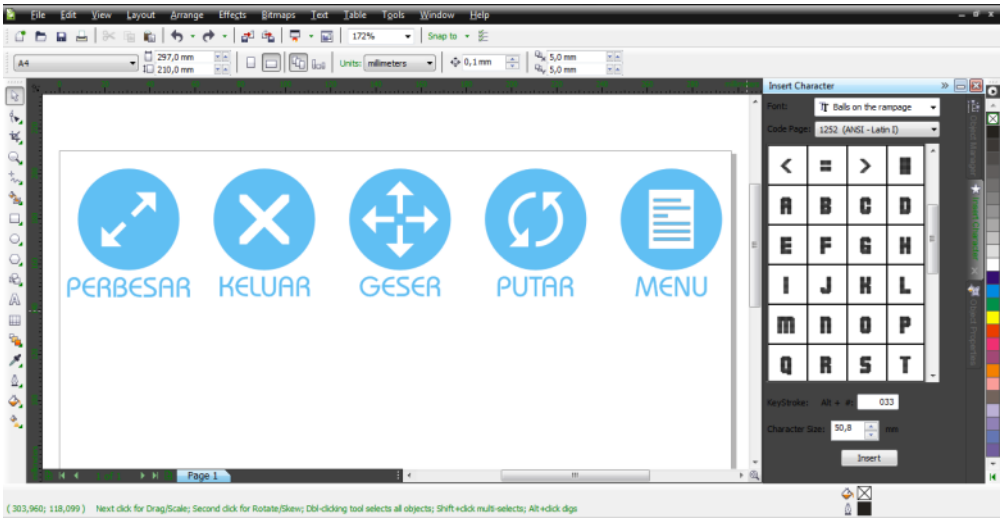
- *ZFX3D Library*

Merupakan pemroses data yang berkaitan dengan *calculus* (kalkulus) yang digunakan untuk pengembangan 3D pada aplikasi *(software)* yang dibuat karena bekerja pada perhitungan koordinat vektor, *rays*, *plane*.

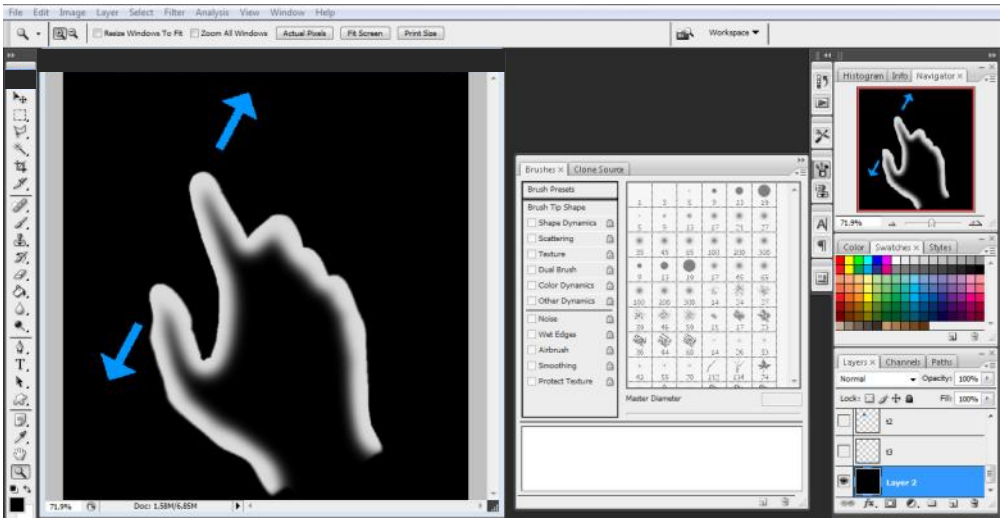
- *ZFXGeneral Library*

Merupakan pemroses data yang berkaitan dengan fitur data pergerakan kamera, sehingga tidak terjadi kesalahan data disekitar rotasi metrik. Data yang diberikan tanpa mengalami kesalahan sehingga dapat diintegrasikan dengan persamaan matematis pada modul.

Sebuah *graphics engine* hanyalah perangkat lunak yang digunakan untuk mengolah berbagai konten yang ada didalam perangkat lunak tersebut. Berbagai konten tersebut terdiri dari obyek dua dimensi (2D), tiga dimensi (3D), audio, *file* skrip bahasa *coding* dan berbagai fitur efek lainnya. Pembuatan obyek 2D dapat menggunakan perangkat lunak desain grafis dua dimensi yang umum kita kenal, dan dapat pula menggunakan perangkat lunak *editor* foto apabila obyek 2D yang dibuat memerlukan sentuhan aliran realis.



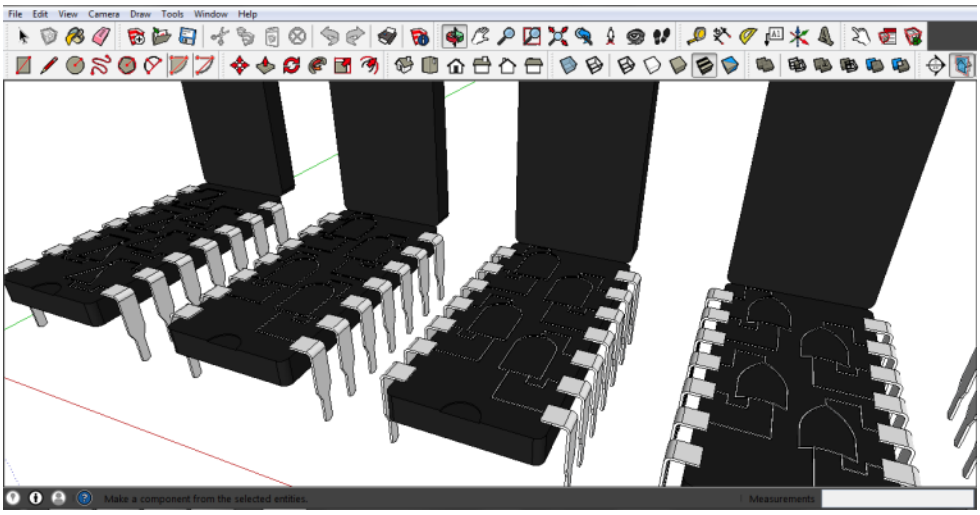
Gambar 26. *Software* Perancang Obyek 2D Berbasis *Drawing*



Gambar 27. *Software* Perancang Obyek 2D Berbasis *Photo-Editor*

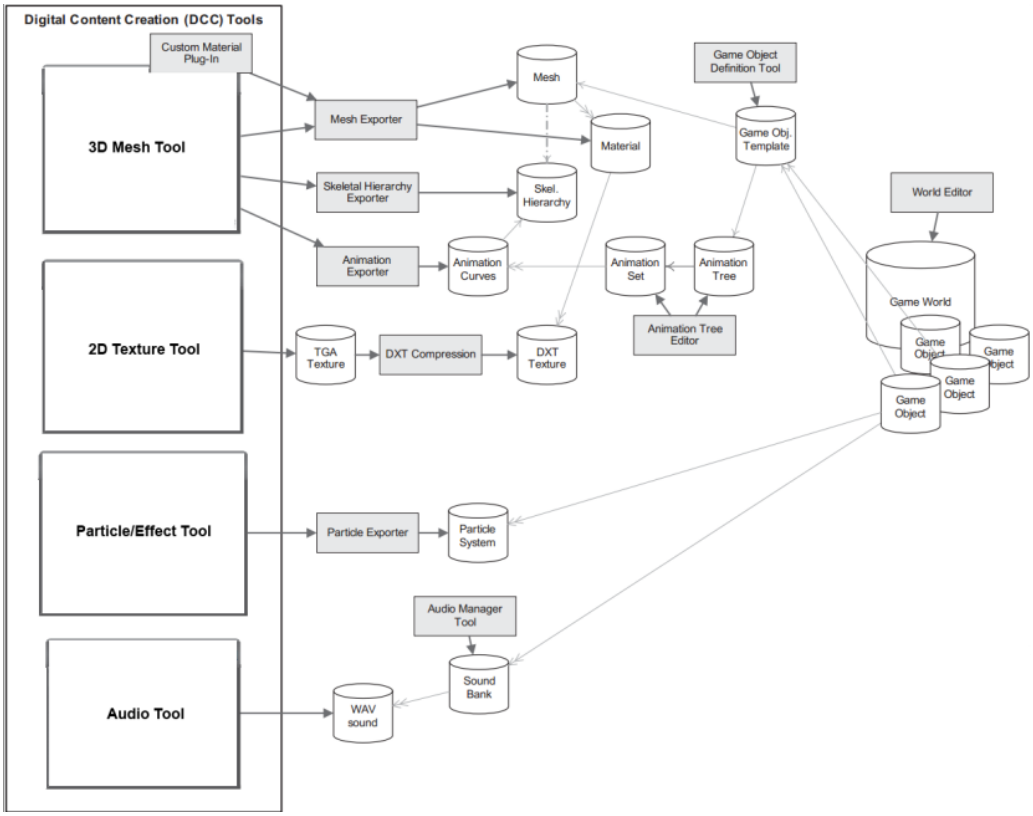
Proses perancangan obyek 3D memerlukan perangkat lunak CAD*(Computer-Aided Design)* yang secara profesional dirancang untuk membuat obyek 3D (Zerbst, 2004:668). Setiap perangkat lunak CAD tentu memiliki tingkatkan dan kemampuan yang beraneka ragam dalam merancang sebuah obyek tiga dimensi (3D). Sebagai seorang perancang grafis 3D pemilihan perangkat lunak CAD tentu disesuaikan dengan kebutuhan dan kemampuan. Sebuah *graphics engine* melakukan proses *input* data dalam berbagai bentuk format, mulai dari bentuk data *3D mesh*, *texture bitmaps*, *animation data*, *audio file*. Semua bahan baku tersebut akan diolah dengan sebuah perangkat lunak, jenis perangkat lunak tersebut lazim disebut dengan perangkat lunak *digital content creation (DCC)*. Perangkat lunak *(software)* DCC memiliki kinerja secara umum untuk memproduksi satu jenis tipe data, meskipun beberapa perangkat lunak DCC yang lain dapat memproduksi tipe *multiple data* (Gregory, 2009:49). Sebuah perangkat lunak DCC selain dapat

memproduksi data obyek 3D (*3D mesh*), juga dapat memproduksi data animasi obyek bersangkutan untuk *engine* dapat memainkan obyek 3D.



Gambar 28. *Software* Perancang Obyek 3D

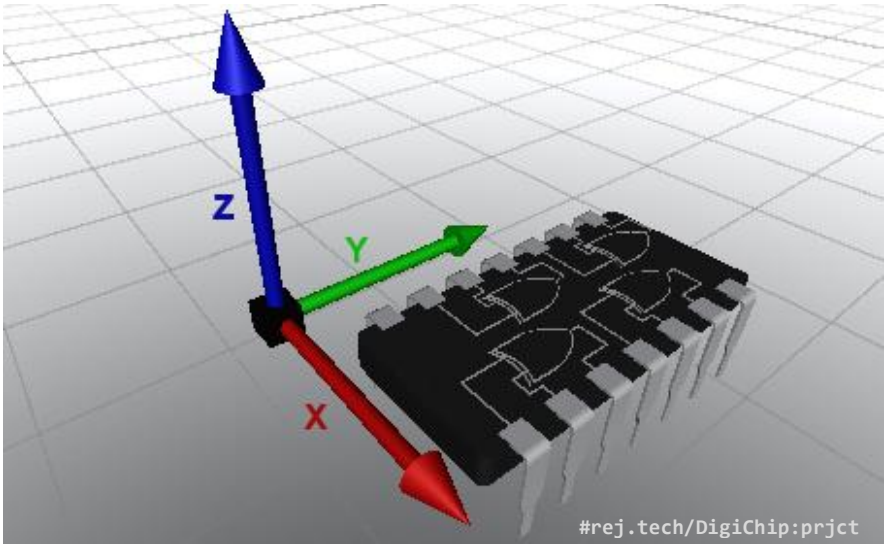
Perangkat lunak *graphics engine* jenis dan merk apapun dirancang memiliki kemampuan untuk mengolah data dalam bentuk yang kompleks, baik dalam format *asset, configuration files, scripts,* dan sebagainya (Gregory, 2009:49). Format *file* yang diproduksi oleh perangkat lunak DCC selanjutnya akan di *export* menuju perangkat lunak *graphics engine*. Gambar berikut merupakan mekanisme kerja pada sebagian besar perangkat lunak *graphics engine* modern. Gambar arah panah abu-abu menunjukkan bagaimana aliran data dari perangkat lunak yang digunakan dalam membuat *original source assets* untuk siap digunakan *software engine* bersangkutan.



Gambar 29. Mekanisme Kerja *Software Graphics-Engine*

Seorang perancang grafis dalam membuat lingkungan virtual menggunakan *graphics engine* dihadapkan pada cabang keilmuan matematika berupa trigonomeri, aljabar, statistik, kalkulus, vektor dan metrik (Gregory, 2009:137). Obyek tiga dimensi (3D) memang bekerja berada pada dimensi vektor (x,y,z). Koordinat (x,y,z) berpadu membentuk sebuah ruang tempat dimana semua obyek 3D dan 2D dalam dunia virtual berada. Sebuah *graphics engine* membutuhkan ruang untuk bekerja pada lintasan untuk menentukan posisi, arah putaran, skala obyek dan menganimasikan obyek. Ruang yang terdiri dari koodinat (x,y,z) tersebutlah yang dalam ilmu matematika disebut vektor. Vektor merupakan jumlah dari sebuah besaran dan sebuah arah dalam

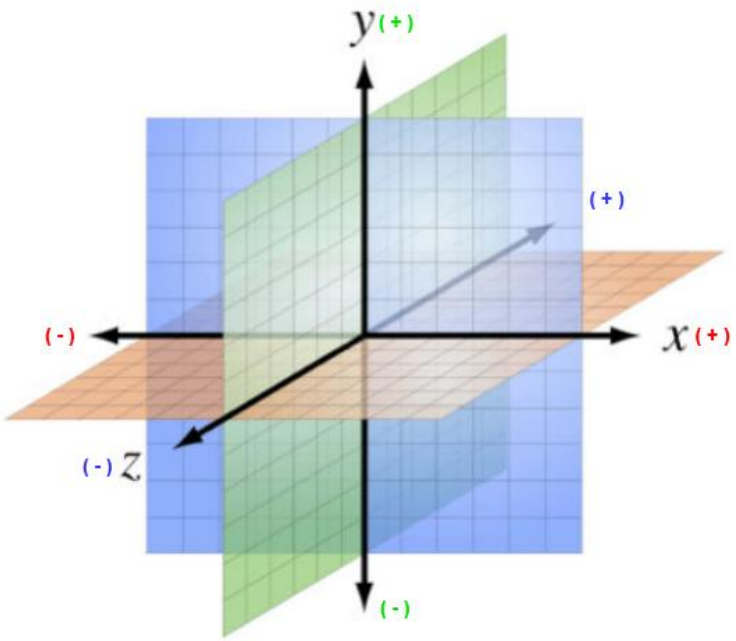
ruang *n*-dimensional (Gregory, 2009:140). Vektor dapat memperlihatkan sebuah garis yang dibangun dari dua titik dalam ruang vektor, sehingga dapat memperlihatkan titik mana yang disebut titik mula dan yang disebut titik akhir. Pada aktivitas merancang obyek grafis pada komputer, istilah vektor digunakan sebagai referensi membuat titik koordinat *(position vectors)* dan untuk garis dalam aljabar *(purely directional vectors).*



Gambar 30. Dimensi Vektor Kooordinat (x,y,z)

Perancangan obyek grafis baik dalam bentuk dua dimensi (2D) maupun dalam bentuk tiga dimensi (3D), semuanya bekerja pada sistem koordinat tiga dimensi (x,y,z). Obyek dua dimensi (2D) hanya saja bekerja pada koordinat *cartesian* x dan y. Sistem koordinat bekerja pada, sumbu x berupa gerakan horisontal dari kiri hingga kanan, sumbu y berupa lintasan vertikal dari atas ke bawah, dan sumbu z merupakan arah menuju masuk dan

keluar pada layar (Parisi, 2015:34). Sumbu putar pada sistem koordinat dapat diubah sesuai keinginan, misalnya sumbu z sebagai sumbu vertikal, sedangkan sumbu y sebagai arah kedalam dan keluar dari *screen* (Parisi, 2015:35). Proses perancangan dunia virtual tiga dimensi (3D) berupa *software* berisi interaksi manusia dengan "mesin", lazimnya koordinat dua dimensi (2D) digunakan manusia untuk pengendalian obyek dari koordinat 2D layar monitor. Layar monitor (jenis *touchscreen*) karena berbasis koordinat x dan y (2D).



Gambar 31. Sistem Koordinat

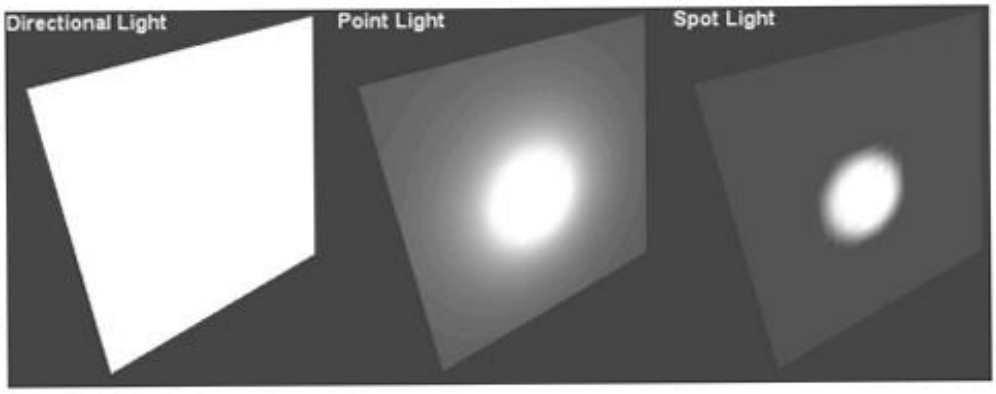
Perangkat lunak *graphics engine* tidak dapat bekerja sendirian tanpa adanya sebuah unit *graphics hardware* atau lazim dikenal dengan GPU (*Graphics Processing Unit*), demikian pula saat melakukan proses *rendering* dan *compelling* (Harbour, 2011:4). Kualitas dan performa tampilan visual juga sangat tergantung dengan perangkat GPU. Fitur pada *graphics engine*

ada berbagai ragam, antara lain berupa *3D shader-based rendering, 2D sprite animation, static meshes, hierarchical meshes, mesh rendering, shader-based dynamic lighting, entity management, picking object,* dan *collision detection* (Harbour, 2011:103).

Unit *graphics engine* memiliki kemampuan dalam perhitungan obyek untuk melakukan berbagai efek fisik atau *game physics*. Fitur fisika pada obyek digunakan agar rekayasa perancangan grafis lebih terlihat mengagumkan (Millington, 2007:2). Efek fisika pada obyek 3D diantaranya berupa gravitasi, inersia, memantul/bergetar dan mengapung. Selain itu efek fisika pada rekayasa grafis digunakan juga sebagai pembuatan efek *spark* (kilatan cahaya), *firework* (bunga api), asap, dan ledakan. Fitur berupa perhitungan efek fisika pada pembuatan obyek grafis dan animasi sangat berguna seperti halnya ketika membuat sebuah obyek kendaraan, sehingga akan terlihat begitu realis efek pergerakan suspensi, roda, seusai dengan kontur jalanan dan begitu pula ketika bodi kendaraan menghantam benda padat yang ada di sekitarnya. Hal inilah yang dimaknai sebagai pembuatan *virtual environments* (lingkungan virtual) benar-benar representasi dari *real-world* (dunia nyata) yang dijadikan model atau acuan, seperti yang telah dijelaskan pada pembahasan sebelum. Efek fisika pada obyek dirancang untuk membuat karakter pada obyek dan mensimulasikan obyek yang dibuat (Millington, 2007:3).

Fitur lain pada sebuah perangkat lunak *graphics engine* adalah *material, texture, lighting*. Berbagai fitur tersebut disediakan sebagai upaya

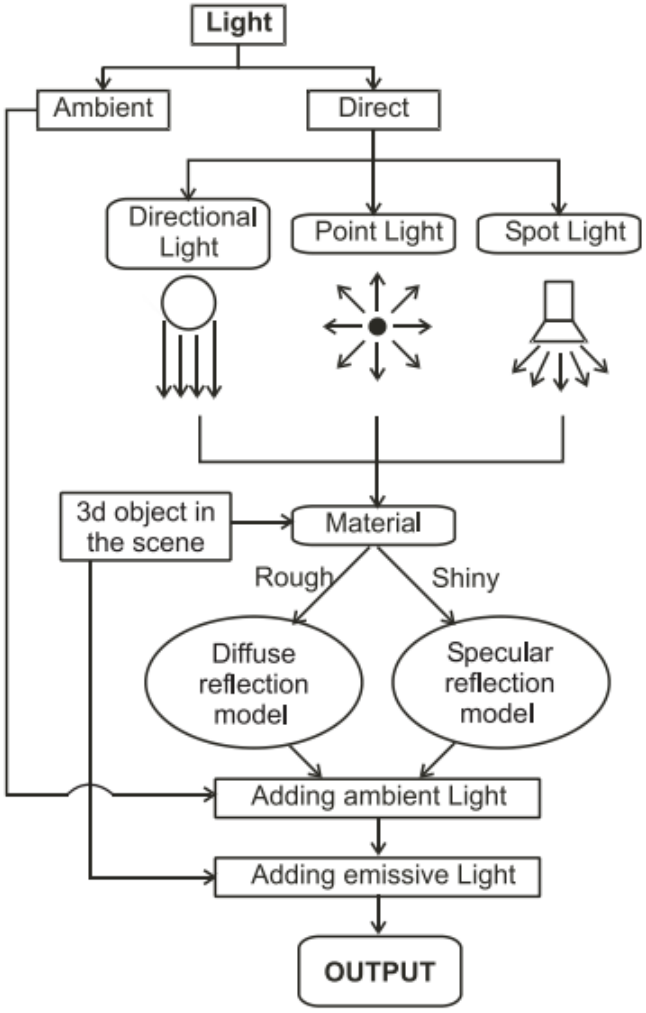
untuk perancangan lingkungan virtual tiga dimensi (3D) yang bersifat *realism* (Zerbst, 2004:194). Fitur pencahayaan merupakan bagian dari fasilitas penting dalam pembuatan obyek dan lingkungan virtual guna mewujudkan kondisi dan suasana yang benar-benar menyerupai keadaan yang sesungguhnya. Perangkat lunak *graphics engine* umunya menyediakan 3 jenis fitur karakter sistem pencahayaan dasar, diantaranya *directional light, point light* dan *spot light* (Zerbst, 2004:195).



Gambar 32. Jenis Pencahayaan pada *Software Graphics-Engine*

- *Directional Light:* Merupakan representasi dari cahaya matahari sesungguhnya, yang memiliki karakter cahayanya menyebar luas dan dan memiliki jangkauan kemanapun. Intensitas cahaya dapat disetel redup-terang layaknya matahari sedang redup dan terang. Warna dapat disetel untuk menyesuaikan layaknya *sun-rise* dan *sun-set*.
- *Point Light:* Merupakan jenis pencahayaan layaknya sebuah lampu pijar, yakni menerangi ke segala arah namun dengan jangkauan yang terbatas tidak seperti cahaya matahari. Fitur pencahayaan ini dapat diatur warna dan intensitas cahayanya.

- *Spot Light:* Merupakan jenis pencahayaan yang merepresentasikan karakteristik lampu sorot. Fitur pencahayaan *spot light* biasa digunakan untuk pembuatan obyek 3D yang memiliki karakteristik layaknya sebuah lampu senter ataupun lampu sorot pada kendaraan. Fitur pencahayaan ini tersedia pada berbagai jenis warna, kombinasi warna dan berbagai intensitas cahaya.

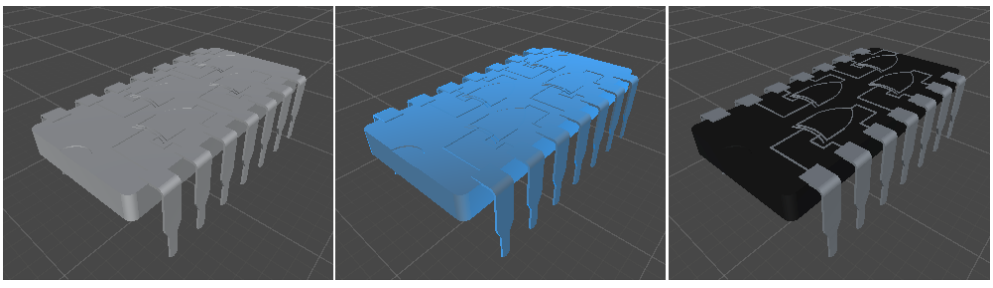


Gambar 33. Jenis Pencahayaan dan Sumber Cahaya pada Grafis Komputer

Fitur pencahayaan pada perangkat lunak *(software) graphics engine* merupakan perpaduan dari kinerja *software* dan *hardware*. Unit *hardware* disini tentunya akan memerintahkan perangkat LCD (*liquid-crystal display*)

pada layar monitor maupun pada LCD *screen* perangkat lain (*smartphone*, *PDA*, *tablet-PC*) yang digunakan untuk menjalankan *software* hasil rancangan. Perangkat *hardware* akan mengeksekusi perintah dari *software* untuk melakukan berbagai aktivitas pengaturan warna dan pengaturan intensitas cahaya berdasarkan besar kecilnya energi listrik yang masuk ke LCD. Sistem demikianlah yang membuat sebuah rancangan lingkungan virtual memiliki realitas layaknya referensi dunia nyata yang sesungguhnya.

Fitur lain yang terdapat pada sebuah perangkat lunak *graphics engine* adalah fasilitas untuk membuat "kulit" pada obyek 3D yang dibuat agar memiliki kualitas layaknya obyek referensi yang sesungguhnya, fitur ini lazim disebut dengan istilah *"skin"*. Seorang perancang obyek grafis apabila ingin memberi "kulit" pada obyek tiga dimensi (3D) yang dibuat, maka harus menyediakan terlebih dahulu dalam bentuk format *texture* (Zerbst, 2004:213). Format *texture* yang didesain sedemikian rupa hingga memiliki kualitas kemiripan yang sejajar dengan referensi obyek asli, maka realitas obyek 3D yang dibuat akan lebih terlihat nyata.



Gambar 34. Perubahan Obyek Ketika Menggunakan *Light* dan *Skin*

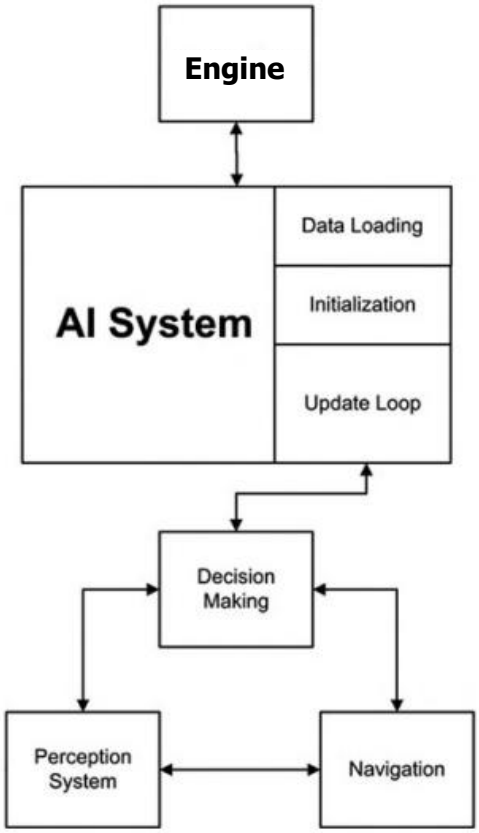
Sistematika pengaturan data dan pengaturan penamaan obyek pada perangkat lunak *graphics engine* dirancang sedemikian rupa untuk membuat sebuah tatanan data yang rapi dan sistematis. Sistem pengaturan data dan penamaan obyek tersebut biasa kita kenal dengan nama *Levenshtein Algoritm* (Algoritma Levenshtein). Levenshtein berasal dari nama seorang ilmuwan Rusia yang merancang algoritma tersebut tahun 1965 yang bernama lengkap Vladimir Levenshtein (Wihlidal, 2006:594). Sistem kerja algoritma tersebut yakni mendeteksi *string* (teknik penamaan pada ilmu *coding*) yang sama untuk kemudian diberi identitas khusus untuk membedakan dengan *string* lain yang sama pada basis data. Penerapan Algoritma Levenshtein juga biasa kita jumpai pada keperluan lain seperti untuk keperluan analisis DNA, deteksi plagiasi, sistem penerjemah dokumen, *speech recognition*, *spell checking* dan *search engine*.

Perangkat lunak *graphics engine* merupakan dominasi dari kumpulan *engine* bernuansa *Artificial Intelligence (AI)*. Hampir semua jenis AI ada pada sistem ini, seperti NN / *Neural Networks* (Jaringan Syaraf Tiruan), *Fuzzy control*, *Genetic Algoritms*. Kecerdasan buatan (AI) pada *graphics engine* sebagai fasilitas untuk merancang obyek grafis (2D dan 3D) agar memiliki karakter *reactive* (Schwab, 2009:2). Sehingga pada saat obyek diberi impuls maka akan melakukan sebuah aksi seperti yang telah diprogram pada bahasa *coding* komputer *(programming language)* yang telah ditanam pada sistem. AI *(Artificial Intelligence)* membuat program komputer menyamai perilaku dan berpikir seperti layaknya manusia, tentunya cara berperilaku dan berpikir

yang rasional. Sistem AI spesialis merancang sebuah komputer yang dikendalikan dengan *smart decisions* ketika dihadapkan pada berbagai pilihan dalam sebuah situasi, untuk menghasilkan sebuah tindakan yang relevan, efektif dan berguna (Schwab, 2009:2).

Sebuah lingkungan virtual yang memiliki sistem *artificial intelligence* dapat dimaknai sebagai area dalam lingkungan virtual tersebut, seperti: sistem *collision avoidance* (penghindar benturan) atau *pathfinding* (pencarian jalur), pengendalian obyek, *user interface*, sistem animasi. Pada beberapa kasus biasanya *artificial intelligence* digunakan untuk menciptakan sebuah sistem yang *poorly* (sulit dikendalikan, kerusakan, keanehan, dsb) yang membuat aktivitas pada dunia virtual terlihat *"stupider"*, namun itu bukan prioritas dalam sebuah sistem AI pada dunia virtual (Schwab, 2009:6).

Semua *engine* AI digunakan untuk mengikuti sistem dasar dalam beberapa bentuk seperti: *decision making/inference*, *perception*, dan *navigation*. Sebuah sistem *decision making* (pengambilan keputusan) memiliki definisi sama halnya dengan aktivitas mengeluarkan logika atau keputusan yang layak dari pengetahuan faktual atau asumsi dasar pemikiran yang benar. Penerapan pada perancangan sebuah dunia virtual, pengendalian *artificial intelligence* mendapatkan informasi tentang lingkungan sekitarnya dan membuat cerdas, keputusan yang layak tentang apa yang harus dilakukan untuk merespon (Schwab, 2009:31).



Gambar 35. Skema Dasar *Artificial Intelligence (AI) Engine*

Produk yang dihasilkan dari perangkat *graphics engine* adalah sebuah perangkat lunak (*software*) multimedia sebagaimana digunakan untuk aktivitas interaktif dan media pembelajaran. Perangkat lunak yang telah dirancang menggunakan *graphics engine* termasuk dalam media yang bersifat multimedia, karena *software* yang dihasilkan berisi berbagai komponen media seperti gambar, tulisan, audio. Secara bahasa istilah multimedia adalah kombinasi banyak atau beberapa media seperti teks, gambar, suara, video yang digunakan untuk menyampaikan pesan atau informasi (Surjono, 2017:2). Multimedia bermuatan pesan-pesan yang dirancang dengan jelas

bagaimana pikiran manusia bekerja cenderung mengarah pada pembelajaran yang berarti dari pada tidak. Sebuah teori kognitif pembelajaran multimedia mengasumsikan bahwa sistem pemrosesan informasi manusia termasuk saluran ganda untuk visual/bergambar dan pendengaran/pengolahan verbal, masing-masing saluran memiliki kapasitas terbatas untuk diproses, dan memerlukan pembelajaran yang aktif serta melakukan proses kognitif yang tepat selama belajar (Mayer, 2009:57).

a. *Bahasa C# (Xamarin C# Language)*

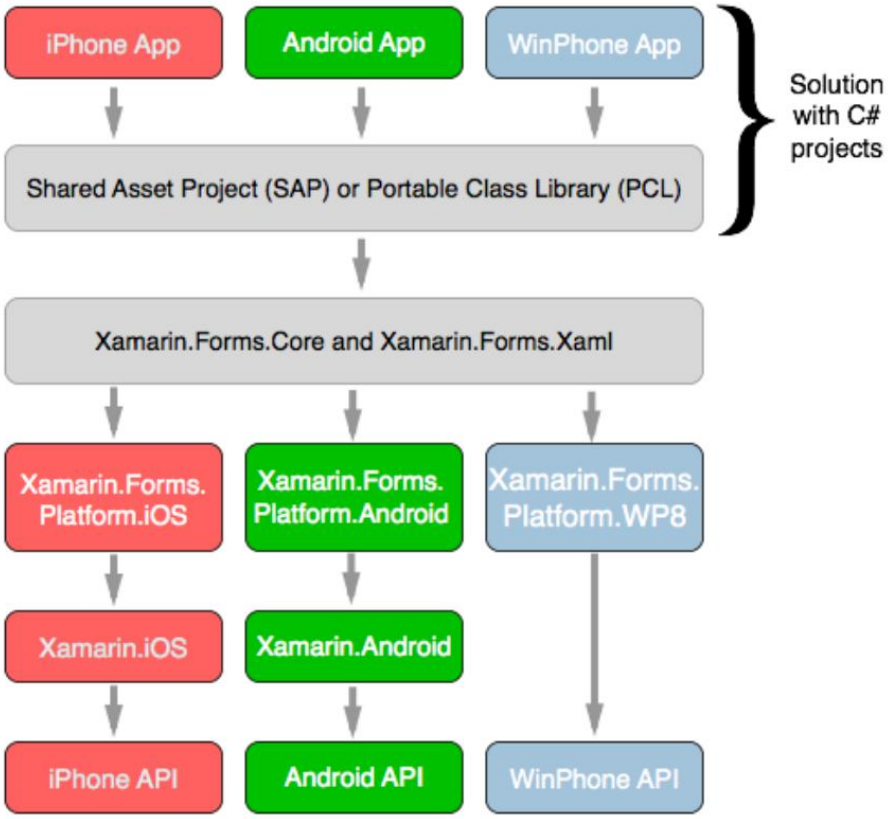
Perancangan produk media belajar praktikum "DigiChip" dalam penelitian ini menggunakan bahasa *coding (programming language)* C#. Skrip bahasa C# dalam pembuatan produk "DigiChip" disusun dan ditulis menggunakan perangkat lunak produk Xamarin dengan menggunakan format penulisan dari perangkat lunak Xamarin (*Xamarin.Forms*). Format penulisan skrip bahasa instruksi *(programming)* berupa Xamarin.Forms digunakan untuk penulisan aplikasi dalam proses perancangan *platform mobile* seperti *iOS*, *Android*, dan *Windows Phone*. Xamarin.Forms memiliki format yang aplikatif untuk seorang perancang program *(programmer)* dalam menulis skrip bahasa C#, *Xamarin.Android libraries* untuk menyasar *application programming interface* (API) yang *native* (alami) pada *platform* yang bersangkutan. Xamarin.Forms menyediakan berbagai fitur dalam proses pekerjaan *coding* (penulisan kode).

Permasalahan yang biasa dihadapi oleh para perancangan skrip kode era dahulu adalah sistem penyusunan bahasa kode yang berulang atau hanya berlaku untuk salah satu *platform*, *operating system*, dan *Application Programming Interface* (API). Pekerjaan *programming* (penyusunan skrip kode), para perancang skrip kode menginginkan adanya kemampuan untuk menulis *single program* yang mampu dijalankan pada berbagai jenis mesin. Alasan itulah yang mendasari penggunaan *high-level languages* digunakan, dan inilah mengapa konsep *"cross-platform development"* menjadi daya tarik para *programmer* (Petzold, 2015:1).

Bahasa C# merupakan jenis *programming language* (bahasa pemrograman) yang baru jika dibandingkan dengan bahasa *Objective-C* dan *Java*. Bahasa C# terlihat lebih berorientasi kedepan, skrip kode benar-benar diketik, bahasa *object-oriented* yang penting, memang terpengaruh dari C++ maupun Java, menyediakan lebih banyak pembersih *syntax* daripada C++ dan sama sekali tidak ada *historical baggage* . Versi pertama dari bahasa C# memiliki bantuan *language-level* untuk *properties* dan *events*, yang mana untuk menyingkirkan bagian lain dan menggunakan deretan yang utama untuk pemrograman *graphical user interfaces* (Petzold, 2015:4).

Penulisan skrip bahasa C# yang diakomodasi oleh Xamarin.Forms dapat diaplikasikan dalam tiga salinan proyek untuk tiga *mobile platform* yang umum kita tahu, seperti *iOS, Android* dan

Windows Phone. Tiga jenis proyek *mobile platform* dalam Xamarin.Forms memiliki ciri khas dalam lingkup kecil, seringkali terdiri dari berupa kode awalan pada *boilerplate* kecil (Petzold, 2015:8).

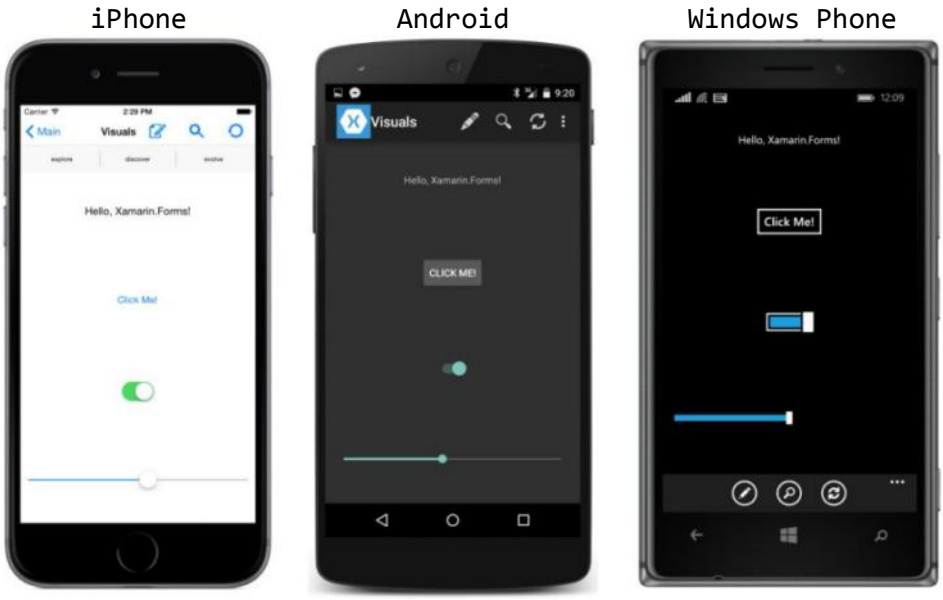


Gambar 36. Grafik Pembuatan *Mobile Software* Menggunakan Bahasa C#

Xamarin.Forms.Core dan Xamarin.Forms.Xaml *libraries* merupakan peralatan dari Xamarin.Forms API. Xamarin.Forms.Core akan digunakan dengan salah satu dari Xamarin.Forms.Platform, tergantung *platform* apa yang akan dirancang. Himpunan *libraries* sebagian besar merupakan kumpulan dari *classes called* "*renderers*" yang akan

mengubah Xamarin.Forms obyek *user-interface* kedalam sebuah *platform-specific user interface*. Sebagai contoh ketika akan merancang obyek *user-interface* yang didalamnya memuat sebuah *toggle* nilai aljabar boolean.

Penyusunan skrip bahasa kode *(programming)* pada Xamarin.Forms, yakni berupa pemanggilan sebuah **Switch**, dan sebuah *class* bernama **Switch** untuk diterapkan dalam Xamarin.Forms.Core *library*. Kode skrip pada ketiga *platform* adalah, **Switch** dinotasikan dengan sebuah **UISwitch** pada *iPhone*, sebuah **Switch** pada *Android*, dan sebuah **ToggleSwitchButton** pada *Windows Phone*. Xamarin.Forms.Core selanjutnya memuat sebuah *class* dengan nama **Slider** untuk menampilkan sebuah *horizontal bar* untuk pengguna dapat mengatur dan memilih sebuah nilai angka. Pada *renderers* dalam *platform-specific libraries*, dinotasikan dengan sebuah **UISlider** pada *iPhone*, sebuah **SeekBar** pada *Android*, dan sebuah **Slider** pada *Windows Phone*. Selanjutnya adalah perancangan sebuah **Label** dengan tulisan *"Hello, Xamarin.Forms!"*, sebuah **Button** bertuliskan *"Click Me!"*. Ketika program dijalankan maka tampilan *graphic user interface* (GUI) pada masing-masing *platform* dan perangkat keras *(hardware)* akan tampak seperti pada gambar berikut ini:



Gambar 37. Tampilan *Application Programming Interface (API)* pada *Platform*

Penyusunan skrip kode bahasa instruksi *(programming)* pada Xamarin, memberikan akses langsung ke *native Application Programming Interface* (API) pada tiap *platform* dan sebuah fleksibilitas ke bagian kode-kode pada C# diantara berbagai *platform*. Penggunaan Xamarin dan C#, membuat produktivitas perancangan skrip bahasa instruksi lebih baik jika dibandingkan bahasa Java atau Objective-C saat pemeliharaan performa keluaran dalam lingkup besar dibandingkan pada HTML atau solusi JavaScript (Peppers, 2014:1). Xamarin memiliki tiga peralatan dalam proses perancangan aplikasi *cross-platform* : Xamarin Studio (dulunya *MonoDevelop*), Xamarin.iOS (dulunya *MonoTouch*), dan Xamarin.Android (dulunya *Mono for Android*). Berbagai

perlengkapan Xamarin menyediakan untuk para perancang sebuah eksplorasi pada *native libraries* yang ada pada iOS dan Android dan membangunnya pada *Mono runtime*. Sebuah *Mono runtime* merupakan *open source* untuk mengimplementasikan C# yang sebenarnya digunakan pada sistem operasi Linux (Peppers, 2014:6).

Aktivitas perancangan skrip bahasa C# sama halnya dengan bahasa pemrograman yang lain, yaitu dikenal apa yang disebut dengan *methods*. Sebuah *methods* merupakan serangkaian *source-code* yang memiliki fungsi melakukan metode tertentu sesuai dengan karakteristik sebuah metode. Aktivitas perancangan *platform* Android kedalam bentuk *application* (perangkat lunak) memiliki perpustakaan *methods* yang tersusun pada *Android bundle*. Sistem operasi Android sangat identik pada sebuah aktivitas, yaitu sebuah penugasan pada unit (mesin) oleh manusia *(users)* untuk melakukan sebuah performa melalui *screen* dari unit tersebut. Sebagai contoh berbagai *methods* pada sebuah aktivitas *callback* (Nayrolles, 2015:33) :

• **OnCreate** : Merupakan *method* pertama yang dipanggil saat membuat sebuah aktivitas. Mengatur *views* dan *perform* pemuatan logika. Memiliki fungsi yang sama pentingnya dengan *method* **SetContentView** dalam mengatur *view* aktivitas.

• **OnResume** : Merupakan *method* yang biasa dipanggil untuk *view* aktivitas agar terlihat pada *screen*. Metode ini akan dipanggil jika sebuah

aktivitas ingin ditampilkan diawal waktu dan jika *user* kembali dari aktivitas yang lainnya.

• **OnPause** : Merupakan *method* yang biasa dipanggil untuk memberi tahu pengguna tentang aktivitas yang salah. Ini dapat terjadi untuk mengarahkan ke sebuah aktivitas baru didalam sebuah *app*, mengunci *screen*, atau eksekusi *home button*. Jika *user* mungkin tidak akan kembali lagi, maka perlu untuk melakukan *save* setiap perubahan yang dibuat oleh *user*.

• **OnStart** : Metode ini terjadi secara segera sebelum **OnResume** dimana sebuah *view* aktivitas sedang tampil pada *screen*. Ini terjadi dimana sebuah aktivitas sedang dimulai dan bilamana *user* kembali ke format ini dari aktivitas yang lain.

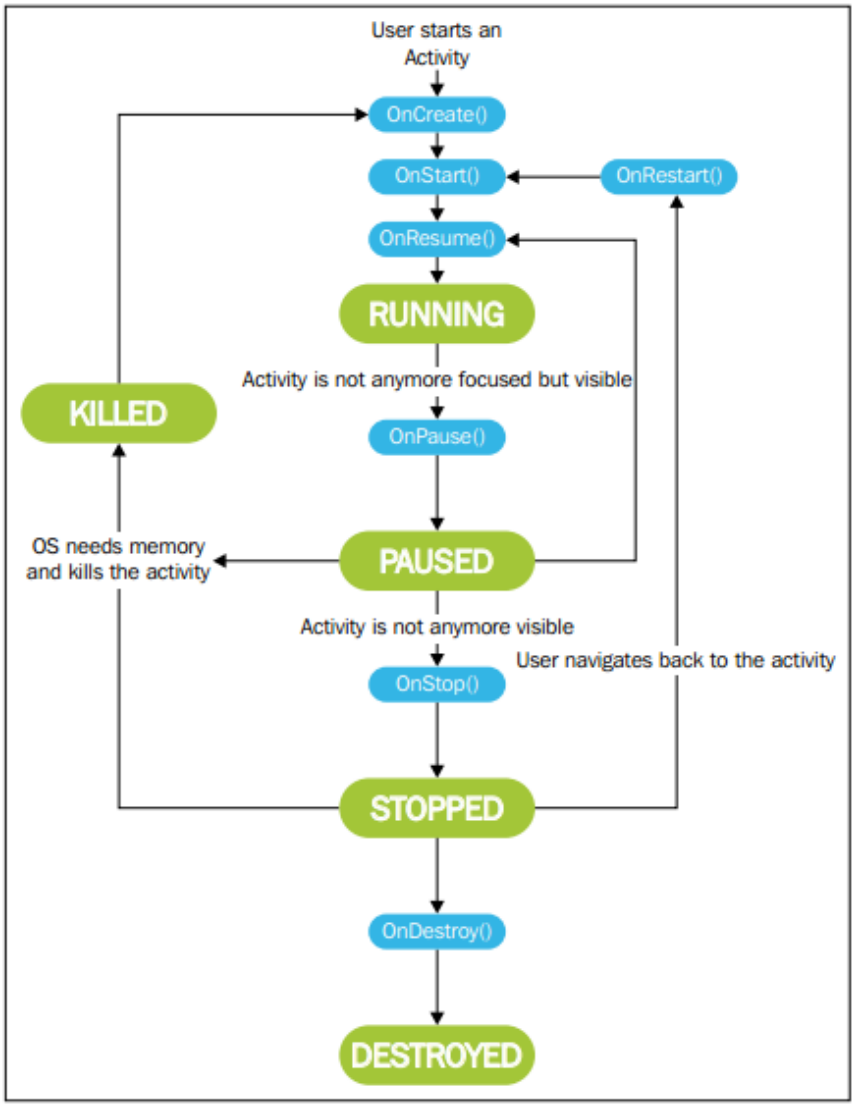
• **OnStop** : Metode ini terjadi secara segera sesudah **OnPause** bila sebuah *view* aktivitas lama tidak tampil pada *screen*.

• **OnRestart** : Metode ini terjadi bila *user* kembali pada aktivitas dari sebuah aktivitas sebelumnya.

• **OnActivityResult** : Metode ini digunakan untuk berkomunikasi dengan aktivitas lain dalam aplikasi lain pada Android. Metode ini digunakan bersambungan dengan **StartActivityForResult**; sebagai contoh, ketika ingin menggunakan metode ini untuk tersambung dengan aplikasi Facebook untuk *user* dapat melakukan aktivitas *login*.

• **OnDestroy** : Metode ini dipanggil untuk aktivitas agar terbebas dari *memory*. Performa apa saja yang ditambahkan akan mengalami *clean-up*

dengan meminta bantuan dari sistem operasi, demikian pula penempatan setiap obyek bermuatan berat lainnya saat aktivitas tengah digunakan.



Gambar 38. Kinerja *Methods* pada *Platform Android*

Aplikasi *(software)* Android dikirimkan untuk instalasi dalam sebuah format *Android package* , yang mana sebuah arsip tersebut dengan ekstensi ".apk". Sebuah *Android package* memuat *apps code*

dan semua *file* pendukung untuk jalannya sebuah aplikasi *(app)*, diantaranya adalah: *Dalvik executables (*.dex files)*, *Resources*, *Native libraries*, dan daftar muatan aplikasi (Reynolds, 2014:36). Perancangan sebuah *platform* Android dalam bentuk aplikasi (app) tidak lepas dari sebuah *tools* yang merupakan fitur dari Xamarin, yaitu Xamarin.Android. Berbagai kelebihan menggunakan Xamarin.Android adalah: (1) Efisiensi bagi perancang aplikasi tentang waktu dan energi dalam penguasaan berbagai fitur pada bahasa C#. Bahasa Java dan *OO language* (bahasa berorientasi obyek) lainnya memiliki banyak kesamaan, penguasaan dalam bahasa C# mampu memenuhi tuntutan yang sama seperti halnya pada bahasa Java; (2) Penggunaan bahasa C# dalam keperluan penyelesaian tugas berupa *cross-platform*. Memberikan porsi lebih luas tentang *code base* (bahan utama kode) yang dibutuhkan dalam banyak *platform*. Karena secara umum *user interface code* dan kode untuk transaksi antar perangkat tiap *platform* berpotensi untuk dapat digunakan lagi. Demikian pula saat aktivitas *service client logic*, *client side validation*, *data caching*, dan *client side data storage*, sehingga dapat menghemat banyak waktu secara signifikan (Reynolds, 2014:22).

Proses pengikatan bahasa C# kedalam *native* Android API untuk pengembangan pada piranti *mobile* dan *tablet*, dilakukan oleh Xamarin.Android. Hal demikian itu akan memberikan kekuatan pada *user interface* (UI) Android, *notifications*, grafik, animasi, dan fitur

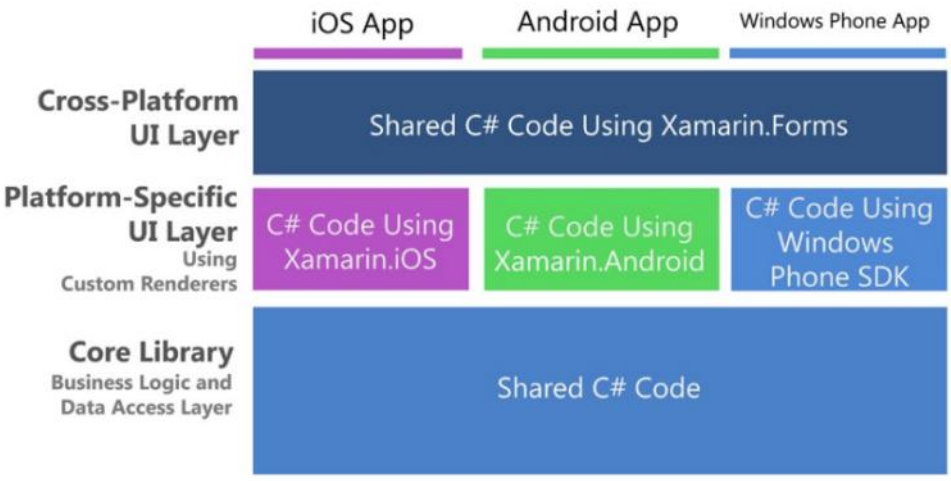
phone, demikian juga *location* dan kamera, yang semuanya menggunakan bahasa C# (Hermes, 2015:2).

Gambar 39. Xamarin C# *libraries* pada *native* OS SDK

Penyusunan skrip bahasa C# yang terdiri dari berbagai kode dengan menggunakan *libraries* tersebut, sebagai aktivitas *development environments* (pengembangan sebuah lingkungan) dan perancangan desain UI *(user interface)*. Desain UI dihasilkan oleh *file* berupa *Extensible Markup Language* (XML) pada masing-masing format *file* dari tiap *operating system* (OS).

Proses perancangan desain UI menghasilkan kemampuan kontrol pada *screen* (layar) untuk *image* (gambar), animasi, dan interaksi pengguna pada sebuah *handheld device* (perangkat komputer genggam). Sebuah *tools* Xamarin merupakan media untuk proses *hybridization*

antara Xamarin.Forms dan kode pada *platform-specific* yang tepat, berguna dan menunjang. Gambaran arsitektur secara kompleks dapat dijelaskan pada grafik berikut:

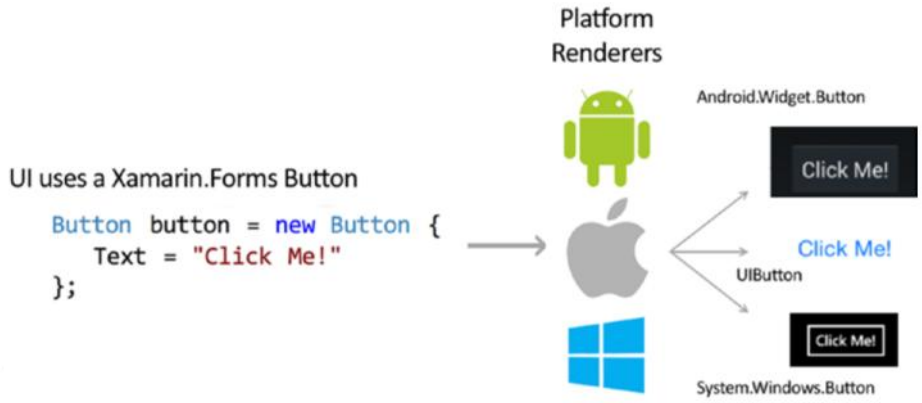


Gambar 40. Arsitektur Xamarin.Forms dengan *custom renderers*

Xamarin adalah berbasis pada Mono Project, yang merupakan *open source* yang menerapkan CLR *(Common Language Runtime)*, namun itu bukan untuk mengatur pembuatan Android *apps* dengan Xamarin. Komponen Mono bertugas mengelola bahasa C# yang menjadi dasar sebuah aplikasi pada banyak jenis platform tidak hanya pada perangkat Android (Nayrolles, 2015:8). Sebuah *compiler* Xamarin memberikan respon untuk perubahan dari bahasa C# kedalam kode Android yang dapat dipahami. Proses *compile* dan penghubungan semua kode bahasa C# menggunakan teknologi yang dimiliki *compiler* dan secara langsung kedalam sebuah file APK, yang merupakan format *file* untuk perangkat Android. Penyebaran APK kedalam target perangkat

Android menggunakan keuntungan dari sistem penghimpunan JIT *(Just In Time)*. Strategi JIT secara konsisten akan menerjemahkan kode secara terus menerus, menginterpretasikan, sehingga akan menghemat kode yang telah diterjemahkan dengan sebuah *cache mechanism* untuk menghidari unjuk kerja yang buruk (Nayrolles, 2015:9).

Mekanisme pengolahan dari sebuah kode **Button** menjadi *native equivalent*, Xamarin.Forms membutuhkan sesuatu yang biasa disebut dengan *platform renderer* (Versluis, 2017:25). Pada *runtime*, kode diinterpretasikan, dan setiap elemen visual melakukan *renderer* secara berturut-turut. Kombinasi dari semua mekanisme akan menghasilkan sebuah keluaran pada *native layout*. Artinya bahwa untuk setiap perancangan *page*, *layout*, dan kontrol UI, membutuhkan adanya proses *renderer*. Kata *"button"* disini merupakan penyebutan dari *source-code* **Button** dan elemen visual pada UI *(user interface)* sebagai kontrol utama pengguna *(user)* dalam berinteraksi dengan mesin *(device*).



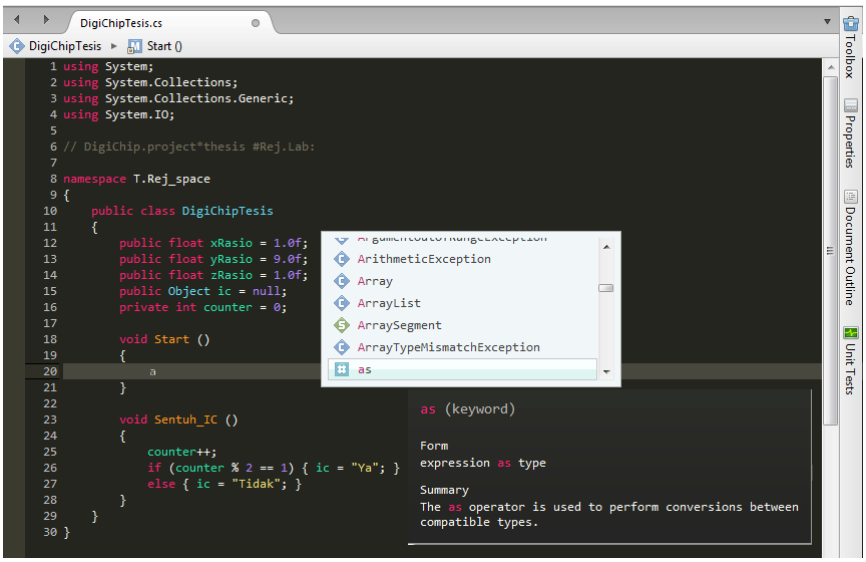
Gambar 41. Mekanisme Penerjemahan *Button* oleh *Platform Renderer*

Bahasa C# memiliki *library project* yang disebut dengan *portable class library* (PCL), yang mempunyai kemampuan untuk mendukung pada *multiple platforms* yaitu meliputi Android, iOS, Windows, Windows Store apps, Windows phone, Silverlight, dan Xbox 360. MVVMCross dan Json.NET juga merupakan *framework* populer yang bersifat *cross-platform* dan *open source libraries* yang memulai proses pengembangan menggunakan PCL (Peppers, 2016:55). Permasalahan yang sering muncul dalam operasi pengaktifan sebuah skrip bahasa C# adalah sebuah kebocoran *memory*. Permasalahan kebocoran *memory (memory leak)* terjadi hampir pada semua struktur bahasa komputer. Sebagian besar masalah *memory leak* pada C# disebabkan oleh *events* (Peppers, 2016:159). Mekanisme preventif dalam mencegah *memory leak* adalah dengan pengaturan bahasa C# dan bahasa *garbage-collected* (GC). GC berupa teknik pengaturan *memory* secara otomatis dan efektif pada perancangan *platform* aplikasi modern (Bilgin, 2016:31). Mekanisme GC adalah dengan mengalokasikan sumberdaya *memory* untuk digunakan obyek dari aplikasi dan mengambil alih kembali ketika tidak digunakan dalam waktu lama oleh aplikasi.

Penyusunan skrip bahasa perintah berupa bahasa C# sama halnya dengan karakteristik bahasa yang lain seperti C, C++, Java dan lainnya. Hampir semua bahasa *programming* memiliki karakter berupa pendeklarasian *class, method, function, string* dan sebagainya. Perangkat lunak Xamarin mengemas aktivitas penyusunan bahasa C# menjadi lebih

menyediakan berbagai fitur yang dibutuhkan oleh para perancang skrip program C#. Perangkat lunak Xamarin menyediakan berbagai fitur yang memfasilitasi sebuah pekerjaan perancangan bahasa C#, salah satunya adalah fitur yang bernama *IntelliSense code snippets*.

Fitur *IntelliSense* merupakan alat bantu untuk memudahkan penulisan kode dalam sebuah library yang selalu dilakukan secara *reusable* dan berulang-ulang sehingga penulisan kode akan lebih baik (Del Sole, 2017:319). Sebuah bahasa pemrograman secara umum memiliki perbendaharaan kode *(library)* yang sangat banyak bahkan mencapai ribuan kode. Seorang perancangan program mayoritas kurang begitu baik dalam menghafal kode program sebanyak itu. Penjelasan yang mudah dipahami yaitu, fitur *IntelliSense* merupakan fitur yang membantu dalam mencari kode program *(source-code)* ketika seorang perancangan skrip ingin mencarinya, dan tersajilah dalam bentuk tampilan *popup*.



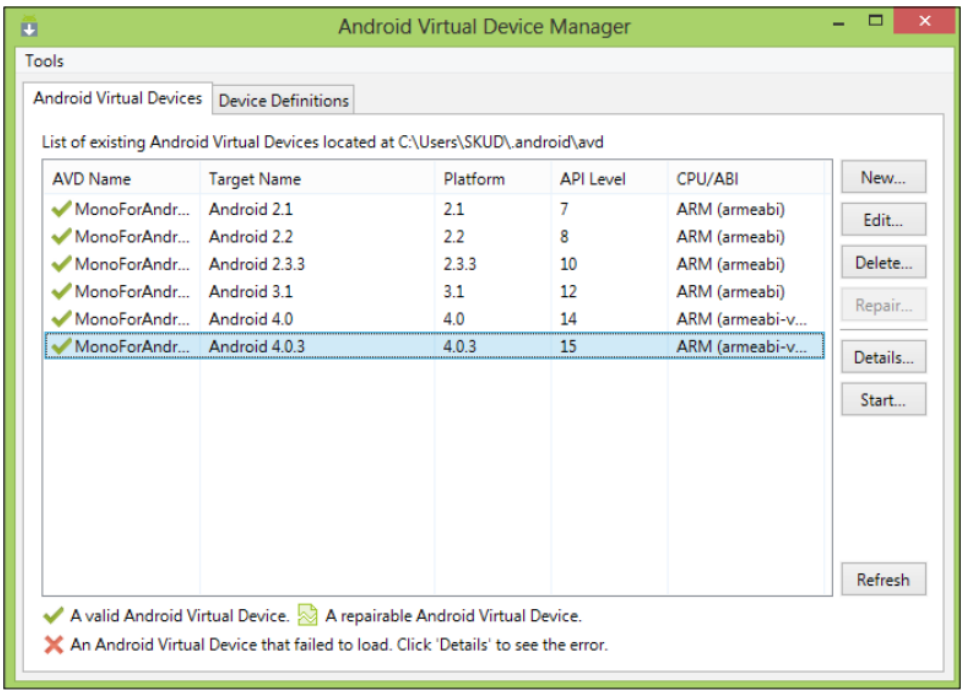
Gambar 42. Penyusunan *source-code* Skrip Bahasa C# dengan *IntelliSense*

b. *Application Programming Interface* (API)

Metode *application programming interface* (API) digunakan untuk aktivitas *built-in* sebuah *user interface* (UI). Pendekatan menggunakan API merupakan pendekatan *code-based* (berbasis kode). Pendekatan berbasis *application programming interface* digunakan untuk para perancangan skrip perintah *(programmer)* yang merasa nyaman dengan penggunaan kode (Smith, 2014:126). Pembuatan *file* skrip bahasa C# dengan ekstensi ".cs" merupakan peran dari sebuah *application programming interface* (Smith, 2014:141). Secara khas, sebuah aplikasi Android hanya dapat dijalankan pada perangkat yang menjalankan konfigurasi API yang ditargetkan atau yang lebih tinggi. Penggantian pada bidang versi minimum Android akan membuat sebuah aplikasi menjadi hanya dapat dijalankan pada perangkat yang lebih tua versinya. Penggantian versi target Android yang berbeda dengan *framework* target dilakukan jika aplikasi menggunakan *libraries* yang spesifik dengan versi API target (Smith, 2014:67).

Proses instalasi sebuah *application programming interface* (API) pada perancangan aplikasi Android dilakukan melalui perangkat yang bernama *Android SDK Manager* (Smith, 2014:93). Sebuah *application programming interface* (API) tidak termasuk didalam *portable subset* dan tidak dapat digunakan dari sebuah *shared code* karena itu seorang perancang skrip perintah *(programmer)* harus menuliskannya dalam sebuah kode *platform-specific* pada proyek iOS dan Android (Del Sole,

2017:186). Aktivitas perancangan aplikasi Android seringkali dihadapkan dengan penentuan sebuah level *application programming interface* (API). Level sebuah API merupakan representasi dari versi spesifik dalam sebuah operating system (OS) dan didalam sebuah API menyediakan pengenalan dalam bentuk angka (Del Sole, 2017:114). Sebagai contoh API level 23 pengenalan untuk Android 6.0 Marshmallow, API level 22 untuk Lollipop 5.1, API level 21 untuk Lollipop 5.0, dan begitu seterusnya. Perancang aplikasi Android dalam membangun *application programming interface* (API) untuk setiap jenis *hardware*, agar pengguna dapat menggunakan setiap jenis *hardware* pada cara yang sama dan dalam hal ini Xamarin menyediakan untuk penggunaan API pada skrip bahasa C# (Nayrolles, 2015:221).

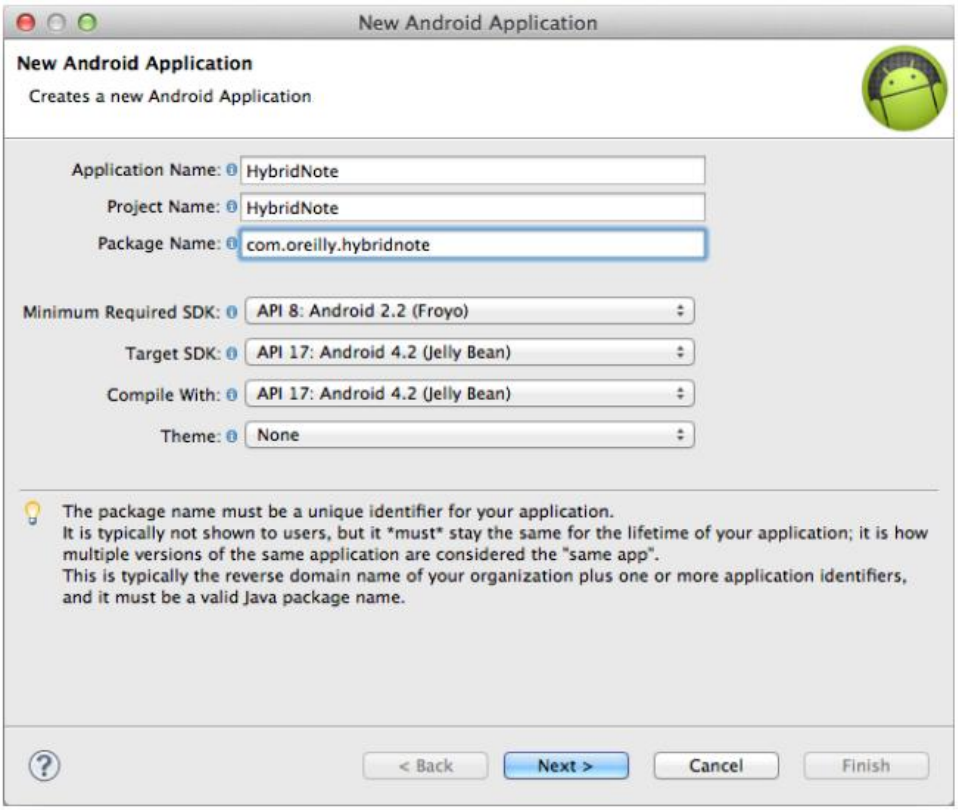


Gambar 43. Penggunaan API pada Setiap Jenis *Hardware/Device*

Komponen *application programming interface* (API) yang bersifat *native* disediakan untuk para perancang aplikasi *(software)* seperti komponen pada *platform* SDK. Aplikasi *native* umumnya dikembangkan dengan menggunakan bahasa pemrograman level tinggi, seperti Java untuk Android, Objective-C untuk iOS, atau C# untuk Windows Phone. Sebuah *platform* API umumnya didesain untuk menyediakan *native app* yang memiliki akses optimal kepada kemampuan perangkat keras *(hardware*). Sebuah *native app* yang bermanfaat adalah apabila optimisasi performa bersifat kritis – sebagai contoh, dalam berbagai simulasi dan *high-end* grafik interaktif. Membangun sebuah *native app* memerlukan *skill* tinggi pada *platform-specific* yang ditargetkan dan sebuah upaya belajar yang keras, sama halnya seperti seorang pengembang yang memiliki komitmen pada sebuah *nitty-gritty* pada sebuah *platform* (Nitin, 2013:2-3).

Sebuah *application programming interface* (API) *level* akan menyesuaikan dengan spesifikasi aplikasi. Seorang perancang aplikasi dapat menentukan *Minimum Required* SDK untuk ditargetkan pada API *level* yang lebih rendah. Tingkatan rendah sebuah API menyajikan perangkat lebih tetapi membatasi lebih sedikit fitur pada sebuah *apps* (aplikasi). API *level* 8 dan sebelumnya dapat mencakup lebih dari 95% perangkat pada pasar Android. Seorang perancang perangkat lunak dapat juga memilih API *level* yang lebih tinggi agar perangkat lunak dapat bekerja pada target SDK yang dipilih, menentukan minimum dukungan

SDK pada versi minimal untuk dapat saling *support* (bekerja bersamaan). Pemilihan API *level* pada tingkatan yang lebih kecil akan membuat pengguna tidak tersediakan tentang fitur pada perangkat *(device)* yang lebih tua (Nitin, 2013:13).



Gambar 44. Penggunaan API dengan Konfigurasi SDK

Setiap versi *platform* Android memberikan dukungan hanya untuk satu level API, dan level API yang lebih rendah memberikan dukungan secara implisit. Pada sebuah gerbang pengembangan Android, sebuah level API didokumentasikan untuk diperkenalkan agar membuat para pengembang mengetahui akan peruntukan tingkatkan pada sebuah

application programming interface (API). Sebuah *platform* Android sangat aktif dikembangkan hingga terwujud dalam jumlah yang banyak yang terperbaharui (Cinar, 2015:11).

Tabel 9. Dokumentasi Perkembangan Level API

Release Date	Platform Version	Platform Codename	API Level
September 2008	1.0	--	1
February 2009	1.1	--	2
April 2009	1.5	Cupcake	3
September 2009	1.6	Donut	4
October 2009	2.0	Éclair	5
December 2009	2.0.1	Éclair	6
January 2010	2.1	Éclair	7
May 2010	2.2–2.2.3	Froyo	8
December 2010	2.3–2.3.2	Gingerbread	9
February 2011	2.3.3–2.3.7	Gingerbread	10
February 2011	3.0	Honeycomb	11
May 2011	3.1	Honeycomb	12
July 2011	3.2–3.2.6	Honeycomb	13
October 2011	4.0–4.0.2	Ice Cream Sandwich	14
December 2011	4.0.3–4.0.4	Ice Cream Sandwich	15
July 2012	4.1–4.1.2	Jelly Bean	16
November 2012	4.2–4.2.2	Jelly Bean	17
July 2013	4.3–4.3.1	Jelly Bean	18
October 2013	4.4–4.4.4	KitKat	19
July 2014	4.4w	KitKat with Wearable Extensions	20
November 2014	5.0	Lollipop	21

Sistem operasi Android pada berbagai versi *platform* memiliki berbagai level *application programming interface* (API). Perkembangan yang telah didokumentasikan sejak rilis versi 1.0 hingga 5.0 tentu memiliki kuantitas berbeda-beda pada pasar penggunanya. Penyebaran

prosentase pada sisi pengguna, sampai pada desember 2014 level API dapat digolongkan sebagai berikut (Cinar, 2015:12) :

Tabel 10. Data Penyebaran Level API pada *Platform* Android

Platform Version	Platform Codename	API Level	Distribution
2.2–2.2.3	Froyo	8	0.5%
2.3.3–2.3.7	Gingerbread	9	9.1%
4.0.3–4.0.4	Ice Cream Sandwich	15	7.8%
4.1–4.1.2	Jelly Bean	16	21.3%
4.2–4.2.2	Jelly Bean	17	20.4%
4.3–4.3.1	Jelly Bean	18	7.0%
4.4–4.4.4	KitKat	19	33.9%

Sebuah *application programming interface* (API) disediakan oleh *framework* Android untuk keperluan pengembangan yang lebih luas pada *user interface* (UI) dalam *platform* Android. Ekosistem Android tergolongkan menjadi banyak varian tipe *layout*, ukuran *display* dan arsitektur *hardware* yang disediakan oleh *framework* Android. Sebuah *framework* Android juga menyediakan pengendalian *input* dan *output*, dan sebuah API memberikan akses pada sebuah *user interface* (UI) sehingga dapat *reusable* dan diatur pada berbagai komponen pada sebuah *application* (Cinar, 2015:138).

Akses pada sebuah *user interface* (UI) diakomodasi oleh *application programming interface* (API), sehingga pada *platform* Android dapat menyediakan UI berupa fitur tampilan *fundamental window* seperti; *action bar*, *toasts*, *dialogs*, dan *notifications*. Komponen *action bar* dan *notifications* merupakan komponen yang sering

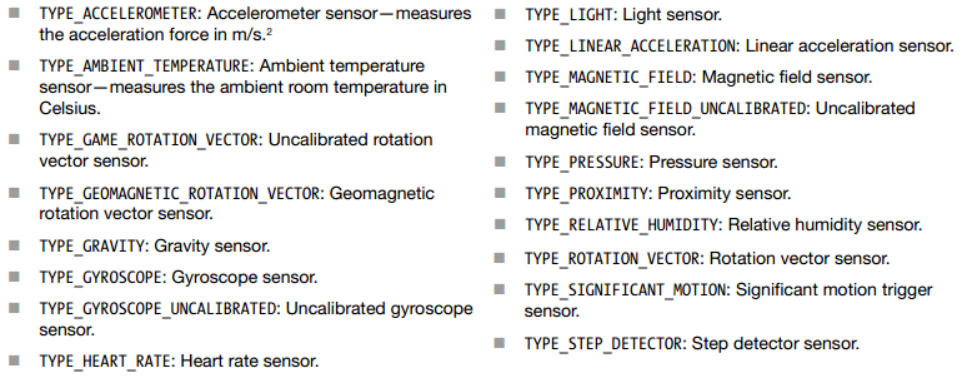
disediakan pada hampir setiap aplikasi Android. Sebuah *action bar* disediakan pada *platform* Android sebagai fitur untuk membantu pengguna dalam memudahkan identifikasi jalannya aplikasi, lokasi sebuah aplikasi, tindakan untuk aplikasi yang penting dan pilihan navigasi (Cinar, 2015:139).

Gambar 45. Tampilan *Action bar* pada API Android

Fitur *application programming interface* (API) pada *action bar* pertama kali diperkenalkan pada level API 11 , tetapi juga tersedia untuk level API sebelumnya yang disiapkan *Android Support Library*. Mengawali dengan level API 11, kode **ActionBar** dimasukkan dalam aktivitas menggunakan *default theme*. Aplikasi dapat mengakses perwujudan dari **ActionBar** kapanpun yang disiapkan oleh metode **getActionBar** dalam kelas **Activity** (Cinar, 2015:140).

Komponen *application programming interface* (API) menyediakan kemudahan dalam aktivitas cara mudah dalam memperlakukan data pada sebuah *platform* Android yang bernama I/O API *(input/output application programming interfaces).* Berkas I/O API memberikan cara mudah dalam menyimpan data dalam bentuk *file* dalam

sebuah perangkat penyimpanan. Sebuah platform Android menyediakan dua tipe penyimpanan, *internal storage (nonremovable)* dan *external storage (removable SD card)* (Cinar, 2015:171). Peranan API dalam aktivitas I/O *(input/output)* dapat dipahami dalam berbagai pengelolaan perangkat *hardware*, seperti Sensor, GPS, *Bluetooth*, *Networking*, Audio *(microphone, speakerphone)*, kamera dan sebagainya. Proses interaksi antara pengguna dengan perangkat *hardware* misalnya sensor, diakomodasi oleh API pada sebuah *framework* Android. Sistem operasi Android memiliki dukungan terhadap banyak jenis sensor. Beberapa sensor ada yang aktual dengan sensor *hardware* dan beberapa yang lain berupa sensor *software-based* yang dapat pula berkombinasi dengan sensor yang lain (Cinar, 2015:200).



Gambar 46. Penggunaan API pada Perangkat Sensor

Sebuah level API merupakan representasi sebuah kumpulan dari kemampuan. Setiap pertambahan level API, sebuah kemampuan baru

disajikan kepada para pengembang (Reynolds, 2014:10). Perancangan sebuah aplikasi Android membawa lebih dari kemudahan dalam menulis kode *(programming).* Sebuah kekayaan *mobile app* memerlukan sesuatu seperti *images*, *audio files*, *animations*, *menus*, dan *style*. Sebuah *framework* aplikasi menyediakan API agar dapat digunakan untuk memuat dan mengerahkan varian tipe sebuah sumberdaya bersama aplikasi *(apps)* Android yang telah dibuat (Reynolds, 2014:18).

Pekerjaan pembuatan skrip bahasa C# dalam tesis ini menggunakan perangkat MonoDevelop untuk mengikat C# bersama *application programming interface* (API). Pokok dari Xamarin.Android adalah Mono untuk Android dan pada iOS adalah MonoTouch. Perangkat Mono dan MonoTouch digunakan untuk *bindings* kepada *native* Android/iOS API untuk pengembangan pada perangkat *mobile* dan *tablet*. Proses ini akan memberikan kekuatan pada Android dan iOS *user interface* (UI), *notifications*, *graphics*, *animation*, dan *phone features* begitu pula *location* dan *camera* yang semuanya menggunakan C#. Setiap rilis sistem operasi (OS) Android dan iOS yang baru akan disesuaikan dengan rilis Xamarin yang baru yang didalamnya mengikat untuk API yang baru (Hermes, 2015:2).

Perangkat *MonoDevelop* menyediakan berbagai kebutuhan dalam pekerjaan menyusun skrip bahasa C# yang akan diikat dengan API. Fitur *MonoDevelop* yang paling mudah diingat adalah perpustakaan *source-code* atau biasa dikenal dengan istilah "*library*", yang biasa dibutuhkan

dalam menyusun sebuah *syntax*. Fitur *library* mengakomodasi berbagai format berupa *class*, *method*, *function*, *string* dan sebagainya. Aktivitas pengaksesan pada *library MonoDevelop* cukup dengan pendeklarasian **MonoBehaviour** pada *syntax* bahasa C# yang ingin dibuat. Pendeklarasian sebuah *library* maupun fungsi pada skrip C# sama halnya dengan pendeklarasian pada skrip bahasa komputer yang lainnya, yang selalu dihimpunkan dengan tanpa baca "{}" untuk eksekusi perintah yang ingin dijalankan.

moveCamera.cs

```
using UnityEngine;
using System.Collections;
using TeamUtility.IO;

public class moveCamera : MonoBehaviour
{
    public int scrollBoundary = 5;
    public int scrollSpeed = 10;

    public int minZoom = -5;
    public int maxZoom = 30;
    public int zoomSpeed = 50;


    void Update ()
    {
        //Debug.Log (scrollAmount);
        float mousePosX = Input.mousePosition.x;
        float mousePosY = Input.mousePosition.y;

        if(mousePosX < scrollBoundary)
        {
            transform.Translate(Vector3.right *- scrollSpeed * Time.deltaTime);
        }

        if(mousePosX >= Screen.width - scrollBoundary)
        {
            transform.Translate (Vector3.right * scrollSpeed * Time.deltaTime);
        }

        if(mousePosY < scrollBoundary)
        {
            transform.Translate (Vector3.forward *- scrollSpeed * Time.deltaTime);
        }
```

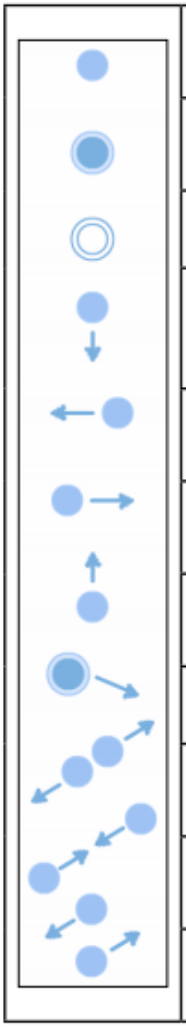
Gambar 47. Penggunaan *library* MonoDevelop pada API C#

Penggunaan *application programming interface* (API) dalam proses perancangan sebuah *user interface* (UI), secara teoritis termasuk dalam jenis pendekatan pragmatis. Proses perancangan sebuah UI

berbagi menjadi dua jenis pendekatan, yaitu *Programmatic approach* dan *Declarative approach* (Reynolds, 2014:160). Jenis pendekatan *programmatic* merupakan pendekatan yang dilakukan seorang perancang *programmer* dan *developer* ketika dalam proses perancangan UI dengan cara melakukan aktivitas pemanggilan perangkat API untuk ditanamkan *(embed).* Jenis pendekatan *declarative* merupakan jenis pendekatan dengan cara pembuatan *file* XAML *(Extensible Application Markup Language)* untuk menetapkan konten dan *layout* pada perancangan *user interface* (UI). Sebuah *application programming interface* (API) memiliki peran penting dalam proses perancangan sebuah *user interface*.

Komponen UI digunakan sebagai sarana interaksi pengguna *(user)* dengan perangkat. Interaksi pengguna *(user interaction)* terdiri dari *interactive control* dan *gesture*. Bagian dari *interactive control* adalah terdiri dari *text input*, *dropdown selection* dan *open selection* (Bilgin, 2016:234). Fasilitas *text input* merupakan salah satu dari banyak jenis media *input*, yang bisa saja terdiri dari bentuk *single line* maupun *multiline*. Fasilitas *text input* sebagai pilihan dalam memberikan masukan berupa teks dengan melakukan sentuhan *(touchscreen)* pada perangkat, serta tampilan *virtual keyboard* pada layar. Fasilitas *dropdown selection* dapat digunakan dalam setiap *platform* digunakan untuk kontrol spesial, yang berupa pilihan dialog atau pernyataan dengan tampilan menurut kebawah. Fasilitas *open selection* merupakan menu interaksi pengguna yang berupa bentuk *Checkbox* atau *Toggle switch* dalam jenis nilai

Boolean. Fasilitas *gesture* disediakan guna mengakomodasi interaksi pengguna dengan perangkat melalui *touchscreen*. Berbagai macam mekanisme interaksi *gesture* disediakan untuk membantu para pengembang untuk menciptakan sebuah *interface* yang dapat berinteraksi dengan pengguna dalam banyak cara kebiasaan (Bilgin, 2016:238).



Tap	In most scenarios, the tap gesture is analogous to a single click with a pointer device. It is primarily used to select a control.
Long Press	Long press or tap and hold is used to access a context menu on Windows Phone. It is used for item selection on Android.
Double-Tap	Double tap is generally used for scaling up / zooming-in on a control.
Swipe Down	Swipe down or pan down is used on vertical scroll scenarios. Also, list controls support swipe down for selection on Windows. It is also common to be used with "Pull to Refresh" implementations.
Swipe Right	Similar to swipe down, swipe right is used on vertical scroll scenarios and sibling navigation scenarios. It is called "flick" if the gesture is fast.
Swipe Left	This is same as other pan gestures. It can also be used to delete a list item on iOS and Windows Phone 10.
Swipe Up	This is another panning gesture. It can additionally be used to reveal a bottom sheet on Android applications.
Tap & Drag	This is generally used as an active gesture to interact with draggable components.
Pinch Out	This is used in active canvas application patterns. It is used to zoom in on a view. On Windows, semantic zoom control makes use of this.
Pinch In	This is similar to the Pinch-Out gesture and is used to zoom out of an active content area of application screen (for example, zoom in on a photo).
Rotate	This is another gesture used on active canvas applications (for example, a map client). It is used to rotate the current view-port.

Gambar 48. Mekanisme UI *Gesture*

Komponen *application programming interface* (API) memiliki perpustakaan *(library)* yang akan berbeda-beda pada setiap jenis *platform* dan sistem operasi *(operating system/OS).* Tiga jenis *platform* dalam tipe sebuah obyek *user interface* (UI) yang serupa, memiliki perbedaan penamaan. Sebagai contoh dalam membuat *user toggle* dengan nilai Boolean, pada *platform* iOS berupa obyek yang disebut "view" dengan memanggil **UIswitch**, pada *platform* Android dengan istilah "widget" dengan memanggil **Switch**, sedangkan pada *platform* Windows Phone dengan penyebutan istilah "control" dengan memanggil **ToggleSwitchButton** (Petzold, 2015:3).

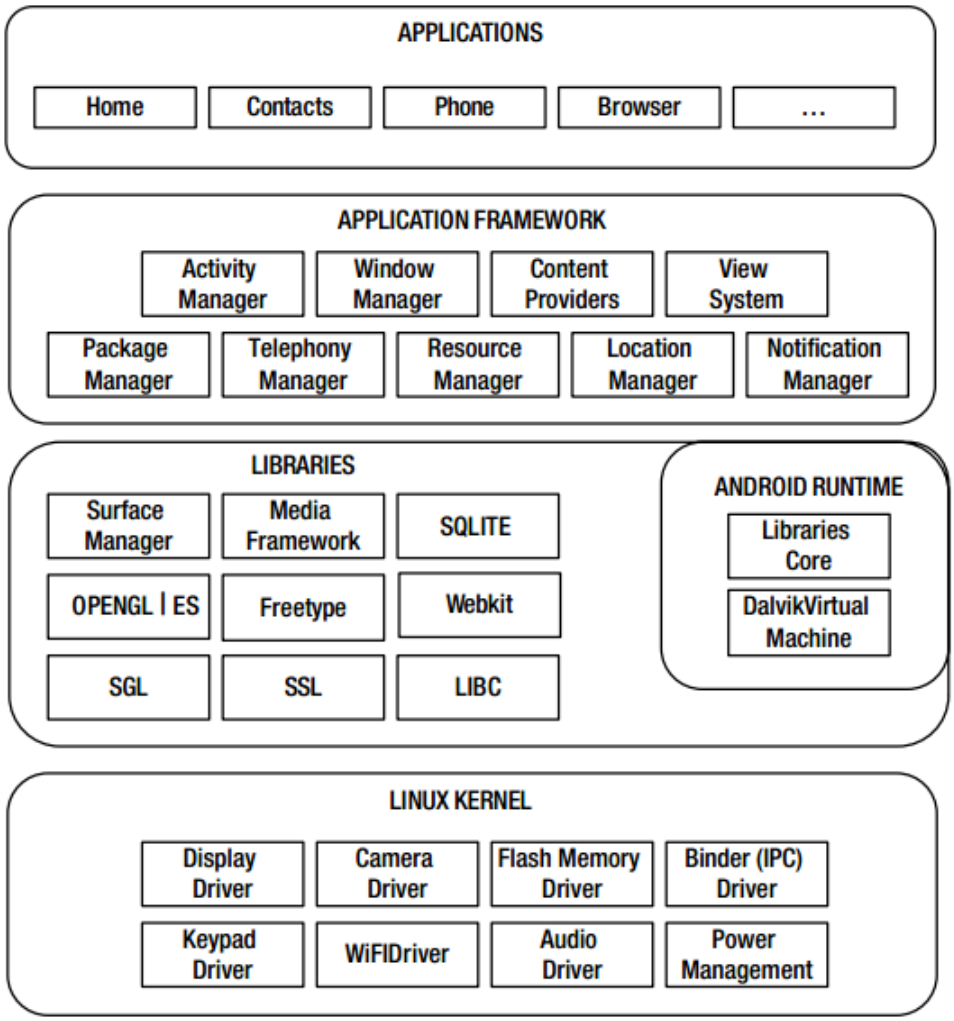
Karakteristik dari sebuah *user interface* (UI) yang modern adalah dibangun dari berbagai obyek visual untuk berbagai macam keragaman. Berbagai obyek visual merupakan kekuatan yang tersusun dari berbagai nama seperti *controls, elements, views, widgets,* yang semuanya akrab digunakan dalam pekerjaan presentasi ataupun interaksi. Pada Xamarin.Forms tampilan obyek pada *screen* secara kolektif disebut dengan *visual elements,* yang terbagi dalam tiga kategori utama yaitu; *page, layout, view.* Semua elemen tersebut dikemas dalam kelas *application programming interface* (API) yang telah ditetapkan diantaranya dengan nama **VisualElement**, **Page**, **Layout**, dan **View**. (Petzold, 2015:17). Penyebutan istilah *"view"* dalam Xamarin.Forms menunjukkan tipe yang lazim digunakan pada berbagai obyek presentasi dan interaksi, seperti; *text, bitmaps, buttons, text-entry fields, sliders,*

switches, progress bars, date, time pickers, dan sebagainya. Komponen yang telah disebutkan tersebut seringkali dipanggil dan dikontrol atau ditampilkan dalam lingkungan pemrograman yang lain.

c. *Android Software Development Kit (SDK)*

Android *Software Development Kit (SDK)* merupakan paket *starter* yang berisi *tools, sample code,* dan dokumentasi penggunaan yang berguna untuk pengembangan aplikasi untuk *platform* Android. Android SDK sebagai alat bantu untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa program Java. Pengembangan aplikasi untuk *platform* Android sendiri biasanya dikembangkan dalam bahasa Java sebagai *native language* menggunakan *Software Development Kit* (SDK)/*tools* dari Android atau sering disebut dengan Android SDK. Android SDK sendiri seperti *software development kit* lainnya juga menyediakan berbagai kebutuhan yang dibutuhkan oleh pengembang untuk melakukan pengembangan aplikasi Android. Tanpa Android *Software Development Kit* (SDK) akan sulit sebuah aplikasi menjalankan fungsi tertentu seperti, *bluetooth pairing*, dan *sensor changes* (Steele, 2011:12). Android SDK menyediakan berbagai fitur dan fasilitas *tools* pengembang Android, *debugger, library, emulator,* dokumentasi, sampel kode serta tutorial secara singkat. Android SDK juga memiliki peranan dalam proses *compile* ekstensi *format file* dari *(.swf)* menjadi ekstensi *(.apk)*, sehingga siap untuk dinikmati pada perangkat *mobile* dengan *platform* android.

Setiap aplikasi yang dirancang untuk *native* Android akan memiliki apa yang disebut dengan sebuah lapisan *framework* aplikasi. Skrip kode berjalan bersama dengan Android *runtime* dari sebuah *DalvikVirtual Machine*. Sebuah *Software Development Kit* (SDK) memiliki fitur bernama *libraries* yang digunakan untuk proses penyusunan skrip kode *(coding)* pada aplikasi *native* Android (Panhale, 2016:22).



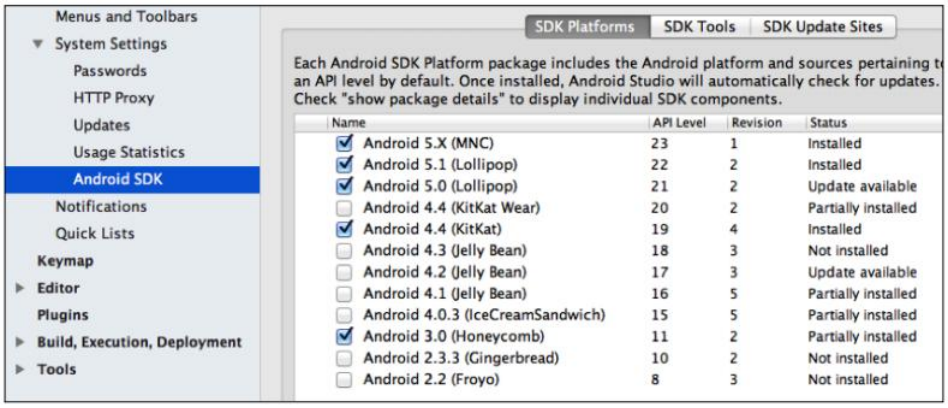
Gambar 49. Peran SDK pada Arsitektur Android *Layer*
(sumber: *"Beginning Hybrid Mobile Application Development"*, Mahesh Panhale)

Sebuah Android *Software Development Kit* (SDK) untuk digunakan oleh pengembang *software* aplikasi agar dapat melakukan *download* secara langsung dari *website* bersangkutan (Rogers, 2009:5). Android SDK mendukung beberapa *Integrated Development Environment* (IDE) yang berbeda-beda seperti, *Sun's Java Development Kit (JDK)*, *Android Developer Tool (ADT)* yang semua melalui proses instalasi pada sistem operasi (OS) yang digunakan (Rogers, 2009:13). Instalasi sebuah SDK pada hakekatnya sama pada berbagai sistem operasi (OS) apapun (Rogers, 2009:14). Android SDK dapat memberikan pilihan salah satu pada target *hardware* yang akan dikenalkan atau memulai sebuah *emulator* sebagai target. Proses tersebut akan secara otomatis memuat aplikasi yang sudah dibuat pada target dan mencoba untuk menjalankan aplikasi tersebut (Rogers, 2009:36).

Android *Software Development Kit* (SDK) tersusun dari *platform*, *tools*, *sample code*, dan kebutuhan akan pendokumentasian untuk pengembangan berbagai aplikasi Android. Android SDK dibangun bersama sebuah *add-on* pada *Java Development Kit* dan terpadu dengan *plugin* untuk sebuah *Eclipse Integrated Development Environment* (IDE) (Schwarz, 2013:12). Android SDK menyediakan pula sebuah peralatan yang sering digunakan *command line* dalam mengakses perangkat *(device)* Android. Peralatan tersebut bernama *Android Debug Bridge* (ADB), yang biasa beroperasi melalui perantara koneksi kabel USB (Schwarz, 2013:15). Android SDK menyediakan peralatan *multiple*

stand-alone untuk *debugging*, seperti *Android Debug Bridge*, *LogCat*, *Hierarchy Viewer*, and *TraceView tools* yang semuanya dapat ditemukan pada *directory* instalasi Android SDK (Schwarz, 2013:380).

Aktivitas perancangan perangkat lunak Android dalam keperluan akses *data service* dari milik aplikasi *(app)*, memerlukan instalasi sesuatu yang bernama *Parse SDK (System Development Kit)*. Sebuah *Parse* SDK memberikan kemudahan untuk panduan dalam memulai, yang mana memuat semua kode, termasuk API *key* untuk aplikasi, sehingga siap untuk menyalin dan menempelkan pada proyek yang dibuat (Ruiz, 2015:37). Proses instalasi sebuah SDK melalui sebuah perangkat lunak yang biasa dikenal dengan nama SDK *Manager.* Ketika membuka sebuah SDK *manager*, akan terlihat daftar yang disajikan seperti SDK *platforms* dan SDK *tools* (Ruiz, 2015:11).



Gambar 50. Operasional SDK *Manager*

SDK *manager* menyediakan semua yang dibutuhkan dalam pekerjaan perancangan aplikasi *(app)*. Proses percobaan menjalankan aplikasi yang dibuat, memerlukan suatu yang bernama Android *emulator*, sebagai

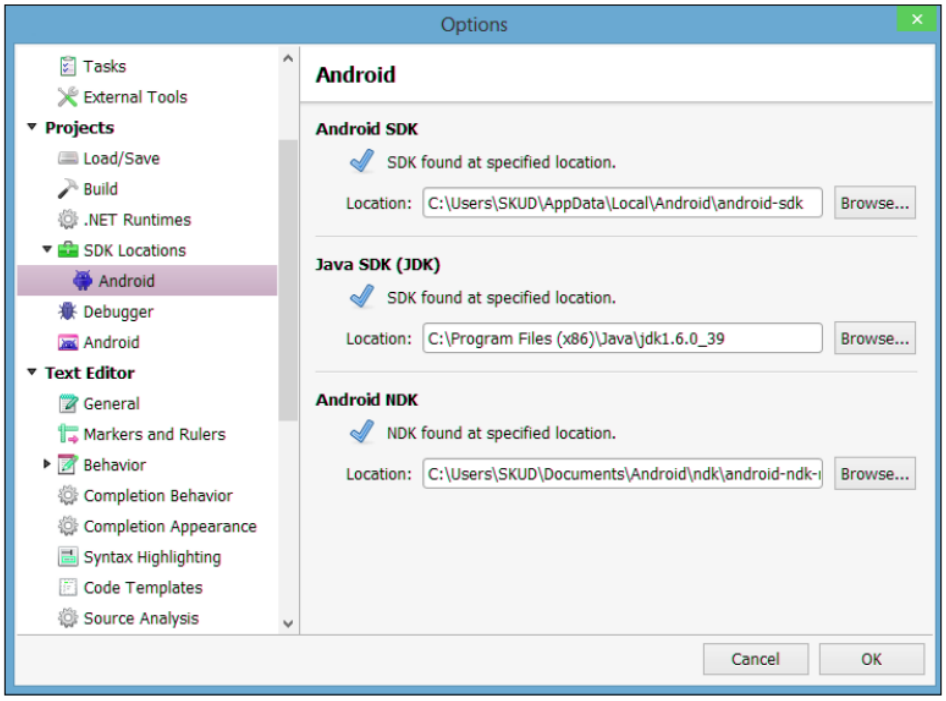
sarana untuk membantu menguji operasional aplikasi pada perangkat *(device)* yang berbeda-beda. Penggunaan *emulator* pada Android memiliki satu keutamaan masalah kecepatan. Selain itu penggunaan *emulator* memiliki keuntungan berupa banyak fitur lain, seperti *resizable windows, copying* dan *pasting* dari komputer lain, dan tentang permasalahan konsumsi waktu pada penggunaan *emulator*.

Android *Software Development Kit* (SDK) memiliki kinerja sebagai *emulator* pada *Android Virtual Device* (AVD) yang berupa *handy keyboard* (Steele, 2011:17), Berikut adalah daftar konversinya :

Tabel 11. Daftar Android OS *Emulator Controls*
(sumber: *"The Android developer's Cookbook - Building Applications with The Android SDK"*)

Key	Emulated Function
Escape	Back button
Home	Home button
F2, PageUp	Menu button
Shift-F2, PageDown	Start button
F3	Call/Dial button
F4	Hangup/EndCall button
F5	Search button
F7	Power button
Ctrl-F3, Ctrl-KEYPAD_5	Camera button
Ctrl-F5, KEYPAD_PLUS	Volume up button
Ctrl-F6, KEYPAD_MINUS	Volume down button
KEYPAD_5	DPAD center
KEYPAD_4, KEYPAD_6	DPAD left, DPAD right
KEYPAD_8, KEYPAD_2	DPAD up, DPAD down
F8	Toggle cell network on/off
F9	Toggle code profiling (when **-trace** set)
Alt-ENTER	Toggle fullscreen mode
Ctrl-T	Toggle trackball mode
Ctrl-F11, KEYPAD_7	Rotate screen orientation to previous or next layout
Ctrl-F12, KEYPAD_9	

Instalasi sebuah Android *Software Development Kit* (SDK) pada aktivitas pengembangan Android, dilakukan dengan melalui bantuan sebuah perangkat lunak. Jika *file* SDK sudah terpasang, dan ingin melakukan instalasi dengan memerlukan Android SDK yang berbeda, maka lakukan *browse* pada tool menu dan tekan tombol *Option*. Pada *window* baru cari Android masuk kedalam SDK *location* pada bagian proyek (Nayrolles, 2015:15).



Gambar 51. Pengait Berkas Android SDK

Android SDK memiliki fitur bernama SQLiteDatabase, yang merepresentasikan sebuah basis data menyediakan berbagai metode untuk operasional SQLite yang memiliki kemampuan melihat aktivitas yang lebih dulu. SQLite memiliki berbagai kemampuan operasional, seperti menerima sebuah *cursor* dalam respon. Sebuah *cursor* dalam

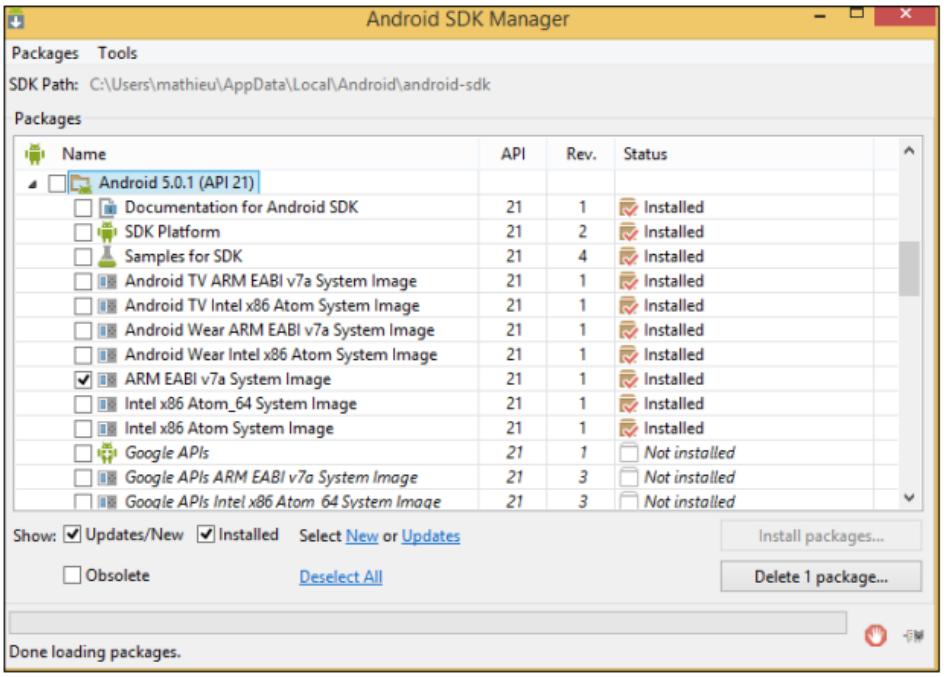
basis data bertugas mengatur penyusunan, yang menyediakan akses ke posisi proses perekaman. Fasilitas *cursor* lebih lanjut melakukan operasional dengan memunculkan kembali *records* (Nayrolles, 2015:99).

Android *Software Development Kit* (SDK) menyediakan berbagai perangkat *kit* untuk keperluan *debugging* pada perangkat *(device)* berupa USB *driver*, yang umum dikenal sebagai perangkat konektivitas guna untuk pengujian aplikasi pada perangkat secara langsung, demikian pula untuk keperluan operasional *emulator* (Nayrolles, 2015:246-247).

Gambar 52. Fasilitas USB *Driver* Android SDK

Android SDK memiliki peralatan lain yang bernama AdMob. Perangkat AdMob *(advertising on mobile)* merupakan perangkat *advertisements* yang diciptakan oleh Omar Hamoui pada tahun 2006. Perangkat *advertisements* seperti AdMob bukanlah perangkat bawaan dari Android SDK, apabila kita menginginkan kita harus menambahkan sendiri

komponen *(additional pieces)* tersebut. Proses melakukan penambahan komponen *advertisements* harus terlebih dahulu memiliki akses pada Android SDK Manager dan meminta izin untuk melakukan instalasi pada sistem yang kita kenakan (Nayrolles, 2015:257).



Gambar 53. Instalasi *Additional Pieces* pada Android SDK

Android *Software Development Kit* (SDK) satu kesatuan bersama aplikasi *Android Device Monitor* (ADM), dimana merupakan paket dengan fitur untuk membantu pengembangan dan *debug* aplikasi untuk dijalankan pada sebuah *emulator* pada perangkat *(device)*. Perangkat *Android Device Monitor* (ADM) menggantikan aplikasi *Dalvik Debug Monitor Service* (DDMS), dimana menyediakan kapabilitas yang sama (Reynolds, 2014:82). Pada Xamarin *Component Store* memilik

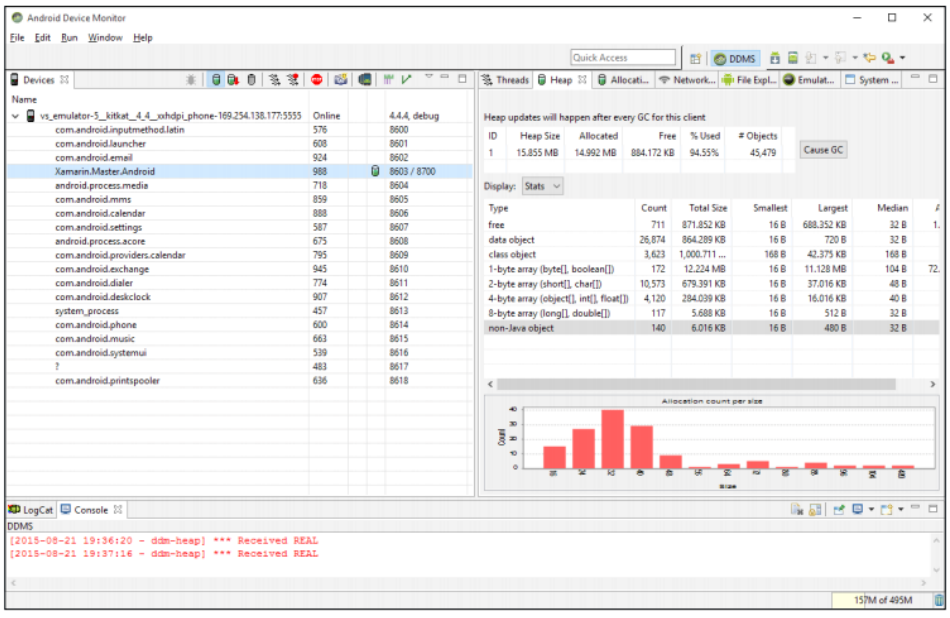
sebuah SDK bernama IBM *MobileFirst* SDK yang menyediakan jembatan penghubung kedalam IBM's *enterprise-grade* produk *platform* aplikasi *mobile* seperti bagian perlengkapan vendor *mobile solutions*. IBM *MobileFirst Platform Foundation* (dahulu IBM *Worklight*) menyediakan sederetan fitur pengembangan aplikasi *mobile* seperti *security*, *cloud data access*, *enterprise integration*, dan *application management* (Hermes, 2015:347). Android SDK memiliki fitur yang *built-in* mendukung untuk *auto-sizing* dan rancangan untuk tampilan, hal ini ada sejak Android memiliki *virtually* yang tak terbatas pada *screen* dan *densities*. Karakteristik Android SDK yang demikian tidak terdapat pada iOS yang memiliki sangat sedikit varian ukuran *screen* pada *device* (Peppers, 2016:124).

Semua Android *package* (apk *file*) melalui proses *signed* dari sebuah sertifikasi untuk dapat dipasang pada perangkat *(device)*. Saat proses *debugging* aplikasi, *package* akan secara otomatis *signed* dengan sebuah *development certificate* yang dihasilkan dari Android SDK (Peppers, 2016:258). Sebagian besar dari *diagnostic tools* seperti IDE digunakan dan didistribusikan seperti komponen pengembangan SDK untuk *platform* target. Untuk *testing* dan *diagnostics*, perangkat *mobile* sungguhan atau SDK-*provided* sebuah *emulator* dapat digunakan (Bilgin, 2016:xi). Para pengembang perangkat lunak memiliki lebih banyak fleksibilitas bersama dengan *emulator* untuk *platform* Android, mengingat pilihan untuk iOS dan Windows Store Apps terbatas untuk

emulator SDK-*provided* (Bilgin, 2016:13). Aplikasi Android dapat dijalankan dan diujicobakan dalam *emulator* pada sistem operasi (OS) Microsoft Windows dan Apple OS X. Android SDK hadir bersama sebuah *default emulator* yang terpasang *(install)* dalam mesin pengembangan. Pilihan dalam aktivitas emulasi dapat dilakukan pada *operating system* Windows maupun OS X. Android *emulator* menggunakan sebuah *Android Virtual Devices* (AVD) untuk menyamai *runtime* Linux *kernel* dan Android. Perangkat AVD tidak memerlukan *software* virtualisasi tambahan apapun untuk dijalankan. Tidak adanya dukungan virtualisasi akan membuat AVD lebih menjadi kurang responsif dan dan membuat waktu *startup* menjadi relatif lebih lama. Hal demikian akan menyediakan rentang pilihan emulasi yang lebih luas dalam aktivitas pengembangan perangkat lunak.

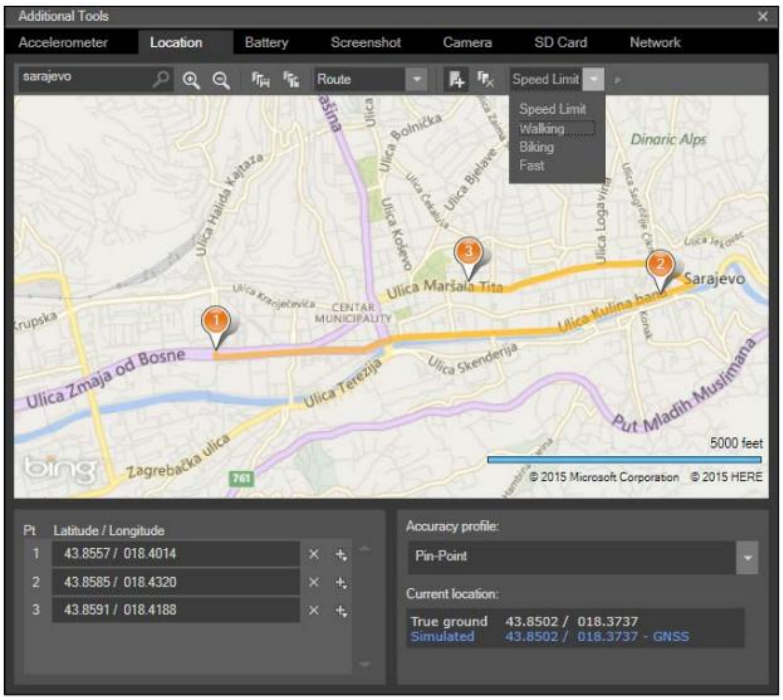
Gambar 54. Salah Satu Jenis AVD dan *Emulator* Android

Android *Software Development Kit* (SDK) juga memiliki perangkat pendukung dalam aktivitas pengembangan perangkat lunak, perangkat tersebut bernama *Android Device Monitor*. Perangkat *Android Device Monitor* hingga saat ini merupakan peralatan diagnostik yang utama dalam pengembangan perangkat lunak Android (Bilgin, 2016:35). Perangkat *Android Device Monitor* akan secara langsung dapat diakses dari item *tool box* pada Visual Studio apabila Android SDK sudah dilakukan instalasi. Halaman utama *Android Device Monitor* menyajikan tiga tampilan setiap perangkat *(device)* atau *simulator* dan dapat dihubungkan dengan *Android Device Monitor*. Penampilan informasi detail dan aktivitas sebuah perangkat dapat dilakukan dengan cara melakukan pilihan pada menu *device* dan memilih menu *allocation* atau *heap* dengan tampilan grafis.



Gambar 55. Perangkat *Android Device Monitor* pada Visual Studio *Emulator*

Komponen Android SDK lain yang sering kita gunakan dalam aktivitas menggunakan perangkat komputer genggam adalah SDK *location service*, sebagai contoh adalah produk Google Play Services. Google Play adalah jalan untuk merekam data akses lokasi dalam *platform* Android (Bilgin, 2016:179). Pengujian informasi lokasi dapat dikatakan sulit dalam sebuah aplikasi *mobile*. Android Emulator pada Android SDK dan Visual Studio Android Emulator dilengkapi dengan fungsionalitas *location emulation*, sistem yang memfasilitasi pengujian data GPS dan diagnostik. Visual Studio Android Emulator menyediakan fitur *emulate* yang dapat digunakan pada sebuah *automobile*, atau transportasi lain, dalam mengetahui perubahan dan pergerakan rute lokasi GPS dalam bentuk gambaran titik koordinat dengan gambaran jarak.



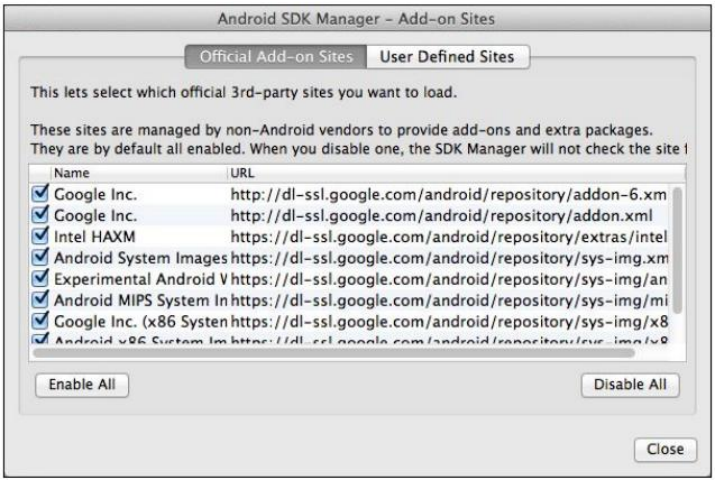
Gambar 56. Perangkat *Location Emulator* pada Android SDK

Secara teknis *emulators* dan *simulators* merupakan kedua teknologi yang berbeda (Smith, 2014:20). Pembahasan pada konteks pengujian aplikasi *mobile*, penting untuk diketahui bahwa pengujian pada Android dilakukan dalam sebuah *emulator*, sedangkan pengujian iOS dilakukan dalam sebuah *simulator*. Perangkat *emulator* pada Android memiliki karakteristik melakukan emulasi sesuai dengan lingkungan perangkat *(device)* yang sesungguhnya. Sebagai contoh jika perangkat yang diuji memiliki RAM 2GB, pada *emulator* akan persis beroperasi pada RAM 2GB. Sedangkan pada sebuah *simulator* iOS, jika komputer sistem operasi Mac yang digunakan memiliki RAM 32GB dengan prosesor 2.3GHz i7 quad-core, maka pada *simulator* iPhone akan sesuai dengan perangkat *(device)* iPhone, dan tentu iPhone tidak memiliki RAM 32GB dengan prosesor i7. Hal demikian pada pengujian iOS tentu dapat dikatakan bukan sebagai *real-world testing environment*, itulah kenapa lingkungan pengujian iOS memberikan istilah dengan *simulator*.

Aktivitas perancangan perangkat lunak Android, sebuah perlengkapan bernama Android SDK menyerahkan proses *emulator* pada perangkat bernama *Android Virtual Device* (AVD). Bilamana akan melakukan pengujian aplikasi Android dalam sebuah lingkungan virtual, maka membutuhkan sebuah *emulator* atau AVD (Smith, 2014:21). Proses pengaturan pada sebuah AVD untuk berbagai jenis perangkat Android idealnya semua sama. Selama tipe dan konfigurasi dari perangkat *(device)* dapat menjalankan sistem operasi (OS) Android,

maka tidak sulit sebuah AVD dapat dioperasikan pada setiap dari perangkat. Sebagian besar perangkat *(device)* Android beroperasi pada prosesor jenis ARM. Demikian pula dengan *Android Virtual Device* (AVD) yang berbasis pada *system image* milik ARM yang biasa dikenal dengan ARM EBI v7a *System Image* (Smith, 2014:26). Arsitektur pada prosesor tipe ARM memiliki tingkat efisiensi dan *suitable* untuk aplikasi perangkat *mobile*, hal ini sangat berbeda dengan arsitektur Intel. Performa *system image* ARM akan terlihat kurang bagus jika diemulasikan dalam *chipset* Intel. Karena itu Intel membuat *system image* khusus bernama Intel x86 Atom *System Image*, untuk menjalankan AVD pada mesin berbasis Intel.

Peralatan lain yang biasa dimiliki oleh sebuah Android *Software Development Kit* (SDK) adalah biasa dikenal dengan nama Add-on Site Manager. Merupakan sebuah perangkat SDK yang digunakan untuk memilih dan mengatur komponen tempat *user-defined third-party* ataupun rombongan alat tambahan Android SDK (Smith, 2014:93).



Gambar 57. Perangkat *Add-on Site Manager* pada Android SDK

Aktivitas *debugging* dan pengujian aplikasi lebih tepat menggunakan *emulator*. Melalui *emulator* perangkat *mobile* akan menjalankan lebih banyak ukuran perangkat dan semua yang dimiliki. Serta dapat memilih dan mengambil versi sistem operasi (OS) mana yang akan dijalankan (Miller, 2017:98). Keuntungan yang lain dalam menggunakan *emulator* adalah, seorang perancang perangkat lunak dapat dengan mudah mengganti bahasa dan wilayah untuk perangkat tanpa perlu rasa kawatir tentang kemampuan bahasa yang memadai, karena hal ini dapat dikembalikan ke menu pengaturan awal.

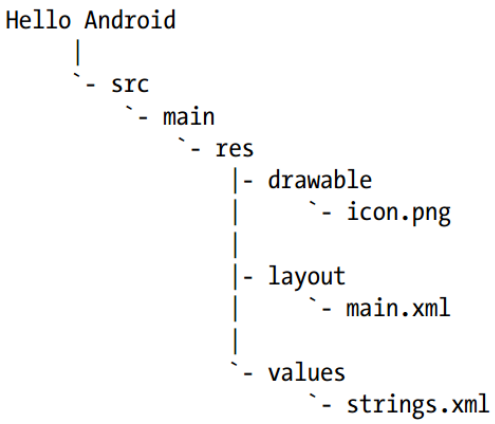
Android *Software Development Kit* (SDK) memiliki peralatan lain yang digunakan dalam mendukung aktivitas perancangan perangkat lunak dalam hal fitur *simulator* dan *emulator*, yaitu bernama *Android Debug Bridge* (ADB). Perangkat lunak ADB merupakan peralatan cekatan yang datang dari komponen pada Android SDK, yang memberikan interaksi dengan konektivitas perangkat Android atau target *emulator*. Perangkat Android menjadi dapat terkoneksi pada mesin utama untuk pengembangan menggunakan TCP ataupun USB (Nitin, 2013:17). Pada setiap versi SDK, dapat dilakukan pemilihan *emulator image* tertentu, seperti untuk perangkat tepat guna dan TV. Spesifikasi versi SDK dapat diatur, serta spesifikasi lokasi untuk tempat *folder* instalasi. Dialog pada instalasi dapat digunakan untuk memasang SDK pada target versi Android untuk mendukung penetapan target pada dialog *New*

Project atau pada *project properties*, seperti yang diinginkan perancang perangkat lunak (Del Sole, 2017:139).

Android *Software Development Kit* (SDK) juga menyediakan peralatan penting untuk pemantauan proses pencatatan dalam *real-time* dengan dukungan proses penyaringan yang lebih maju (Cinar, 2015:4). Android SDK hadir dengan peralatan yang dapat menerjemahkan Java *bytecode* standar kedalam DEX *bytecode* selama pengemasan dari aplikasi Android. DEX *bytecode* menyediakan peluang perbandingan lebih lanjut pada Java *bytecode* (Cinar, 2015:8). Android *Software Development Kit* (SDK) merupakan komponen kunci dalam jajaran peralatan Android. Android SDK menyediakan diantaranya adalah; *Platform API Java Libraries*, *Application Packager*, *Device Emulators*, *Bytecode Optimizer and Obfuscator, Android Debug Bridge, Sample Code and Tutorials* dan *Platform Documentation*. Android SDK adalah komponen yang hanya dibutuhkan untuk pengembangan berbagai aplikasi Android (Cinar, 2015:15-16).

Android SDK memiliki paket peralatan yang menjadi rekan dalam proses pengembangan perangkat lunak, peralatan tersebut bernama Android *Native Development Kit (NDK)*. Android NDK didesain untuk memungkinkan para pengembang dalam aktivitas *build* dan *embed* kode *native* yang tidak terlihat dengan aplikasi berbasis Java. Android NDK terdiri dari *cross compiler*, *debuggers*, *platform header files*, dan *extensive documentation* (Cinar, 2015:16). Android SDK memerlukan

aplikasi pembantu dalam penempatan pada sebuah *subdirectory* khusus dalam sebuah *directory* **src/main/res**. Sebagai contoh ketika aplikasi memiliki sebuah *image* (gambar) alat bantu akan memanggil **icon.png**, yang mana ditempatkan didalam *subdirectory* **drawable**, sebuah *screen layout* memanggil **main.xml** didalam *subdirectory* **layout**, dan beberapa alat bantu *string* didalam *subdirectory* **values** sebagai *file* **string.xml**.



Gambar 58. Hirarki Perangkat *Directory* pada Android SDK

Mekanisme penamaan dalam alat bantu *subdirectory* sangatlah penting. Android *framework* tidak dapat menemukan alat bantu tersebut jika tidak ditempatkan dengan yang seharusnya (Cinar, 2015:70).

Android *Software Development Kit* (SDK) merupakan kumpulan dari *Application Programming Interface* (API) dan berbagai peralatan pengembangan yang digunakan *developers* (para pengembang) untuk menciptakan aplikasi *native* Android (Wagner, 2011:10). Seorang pengembang dapat membuat dan menjalankan sebuah *emulator* perangkat Android pada *desktop*, jika Android SDK sudah terpasang.

Sebuah *emulator*, atau *Android Virtual Device* (AVD), dapat sangat berguna jika tidak memiliki akses ke *device* (perangkat) Android atau untuk keperluan pengujian *app* (aplikasi) dalam berbagai varian resolusi *screen* atau konfigurasi perangkat (Wagner, 2011:21). Seorang pengembang perangkat lunak yang memiliki *device* (perangkat) Android atau menjalankan sebuah *emulator* Android, dapat menggunakan sebuah perlengkapan yang disediakan Android SDK sendiri yang bernama *"adb"* (Gupta, 2014:10). Android SDK memiliki peralatan yang berguna untuk memeriksa *signature* dan mengetahui siapa yg *signed* pada *file* aplikasi Android (.apk), tool tersebut bernama *"jarsigner"* (Gupta, 2014:18).

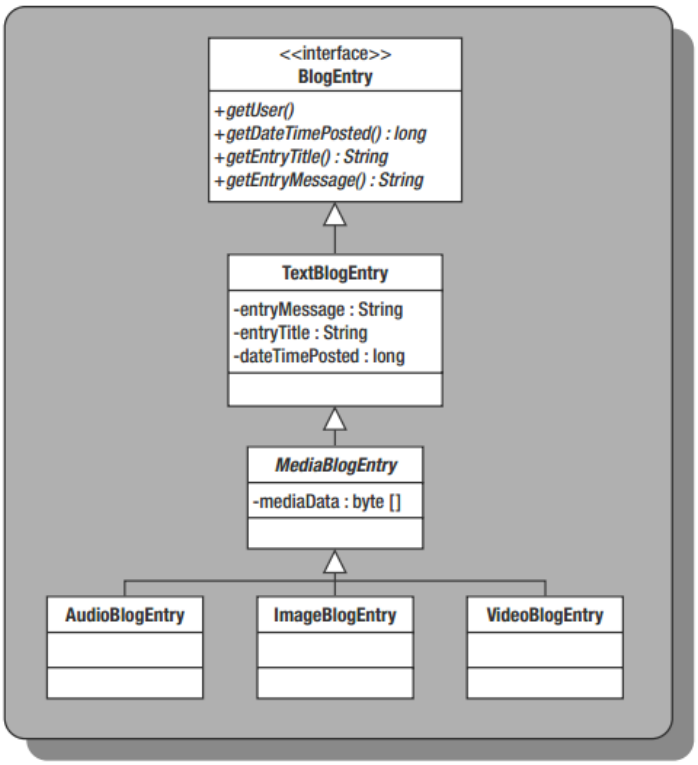
Setiap perangkat *mobile* hadir bersama sebuah SDK yang dapat digunakan untuk pengembangan. Sebuah SDK hadir dengan peralatan pengembangan, *bundling tools*, dan *emulators* yang dibutuhkan untuk menguji kode yang dibuat. Ketika memerlukan akses ke teknologi yang lebih lama atau teknologi yang lebih besar, membutuhkan untuk menggunakan sebuah SDK (David, 2011:4). Pengujian performa dalam Android SDK hanya dapat dilakukan pada Android 2.3 Gingerbread keatas (Milano, 2011:246). Android SDK termasuk diantara jajaran peralatan yang spesialis untuk analisis permasalahan performa dan menentukan kemampuan target untuk diaplikasikan secara optimis (Milano, 2011:251). Aktivitas pengujian perangkat lunak Android tidak lepas dari dukungan berbagai *tool* tambahan untuk meningkatkan kualitas

perancangan perangkat lunak. Peralatan *third-party* yang biasa digunakan antara lain seperti "*JaCoCo*" dan "*Mockito*". Unit pengujian JaCoCo digunakan untuk memperoleh ulasan kode untuk mencari komponen kode yang belum diuji (Nolan, 2015:27). Unit pengujian Mockito bekerja tentang fungsionalitas terhadap *database* dan akses *shared preferences* (Nolan, 2015:35).

8. *Unified Modeling Language* (UML)

Unified Modelling Language (UML) adalah sebuah bahasa visual dan grafis yang digunakan untuk memetakan sebuah konsep dan rancangan perangkat lunak yang akan diproduksi. UML disebut sebagai bahasa untuk pemodelan, bukan sebuah *method* . Secara umum UML *(Unified Modelling Language)* merupakan bahasa untuk visualisasi, spesifikasi, konstruksi, serta dokumentasi. UML ini digunakan oleh para pengembang sebagai sarana untuk mengkomunikasikan idenya kepada para pemrogram serta calon pengguna suatu sistem atau perangkat lunak (Novianti, 2009:A-42). UML *(Unified Modeling Language)* semakin dipandang sebagai standar *de-facto* untuk pemodelan dan desain perangkat lunak. Pada intinya, UML adalah serangkaian notasi berorientasi objek dan sangat efektif untuk menangkap model desain terperinci dari sistem perangkat lunak dalam bentuk yang sesuai untuk diterjemahkan ke dalam beberapa bentuk teknologi pemberlakuan (biasanya kode program) baik oleh pengembang yang memenuhi syarat atau dalam semi-otomatis cara melalui *CASE technology* (Russell, 2006:1).

Berikut salah contoh pembuatan model diagram *Device Blog* dengan menggunakan *Unified Modeling Language* (UML) (Goyal, 2006:193):



Gambar 59. Model UML *Device Blog* aplikasi MIDlet

UML menyediakan 9 jenis diagram yang dapat dikelompokkan berdasarkan sifatnya statis atau dinamis (Prastuti, 2009:24). Ke 9 diagram dalam UML itu adalah;

1. Diagram Kelas

   Diagram kelas bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi serta relasi.

2. Diagram Objek

Diagram objek bersifat statis. Diagram ini memperlihatkan objek-objek serta relasi antar objek. Diagram objek memperlihatkan instansiasi statis dari segala sesuatu yang dijumpai pada diagram kelas.

3. *Use case Diagram*

Diagram ini bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.

4. *Sequence Diagram* (Diagram urutan)

Diagram ini bersifat dinamis. Diagram sequence merupakan diagram interaksi yang menekankan pada pengiriman pesan *(message)* dalam suatu waktu tertentu.

5. *Collaboration Diagram*

Diagram ini bersifat dinamis. Diagram kolaborasi adalah diagram interaksi yang menekankan organisasi struktural dari objek - objek yang menerima serta mengirim pesan *(message)*.

6. *Statechart Diagram*

Diagram ini bersifat dinamis. Diagram ini memperlihatkan *state– state* pada sistem, memuat *state*, transisi, *event*, serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka, kelas, kolaborasi dan terutama penting pada pemodelan sistem - sistem yang reaktif.

7. *Activity Diagram*

Diagram ini bersifat dinamis. Diagram ini adalah tipe khusus dari diagram *state* yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dari suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi - fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antar objek.

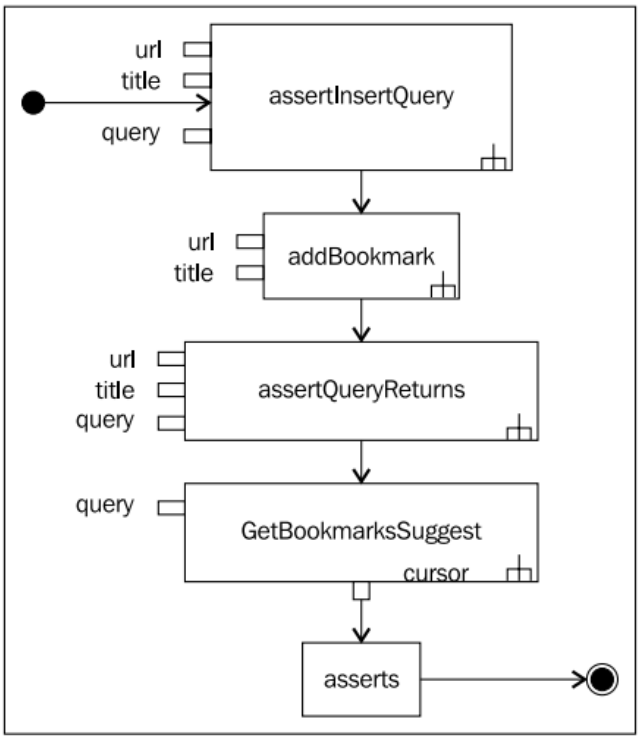
8. *Component Diagram*

Diagram ini bersifat statis. Diagram ini memperlihatkan organisasi serta kebergantungan pada komponen – komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas-kelas, antarmuka-antarmuka serta kolaborasi-kolaborasi.

9. *Deployment Diagram*

Diagram ini bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan *(run time)*. Dengan ini memuat simpul– simpul *(node)* beserta komponen-komponen yang ada di dalamnya. *Deployment diagram* berhubungan erat dengan diagram kompoen yang memuat satu atau lebih komponen–komponen. Diagram ini sangat berguna saat aplikasi berlaku sebagai aplikasi yang dijalankan pada banyak mesin *(distributed computing)*.

UML dapat pula berupa bentuk *Activity Diagram* yang digunakan untuk memahami perhubungan antar metode (Milano, 2011:190).

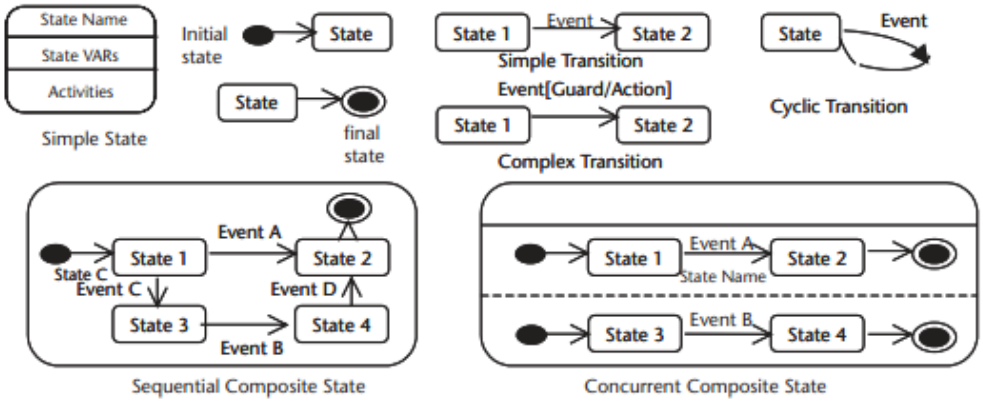


Gambar 60. Model UML *Activity Diagram*

UML merupakan bagian dari serangkaian perangkat lunak *(software)* yang digunakan dalam metode perencanaan *(design)*, termasuk sebuah perangkat lunak bernama *Object-Oriented Analysis and Design* (OOAD), *Structure Analysis and Design* (SAD) (Schulmeyer, 2008:243).

Sebuah UML *state diagram* terdiri dari dua kumpulan elemen: (1) sebuah kumpulan keadaan *nodes*, dan (2) sebuah kumpulan keadaan *transitions*. Setiap keadaan *node* biasanya merepresentasikan sebuah program dengan keadaan aktif, dimana dapat *precondition* atau *postcondition* pada operasional program *(event)*. Setiap diagram mempunyai hanya satu kondisi awal, dan nol atau satu pada kondisi akhir. Setiap keadaan *transition* biasanya merepresentasikan operasional program, sebuah transaksi, atau suatu

event yang mengubah sebuah status program dari satu ke yang lain (Gao, 2006:221).



Gambar 61. Berbagai Notasi UML *State Diagram*

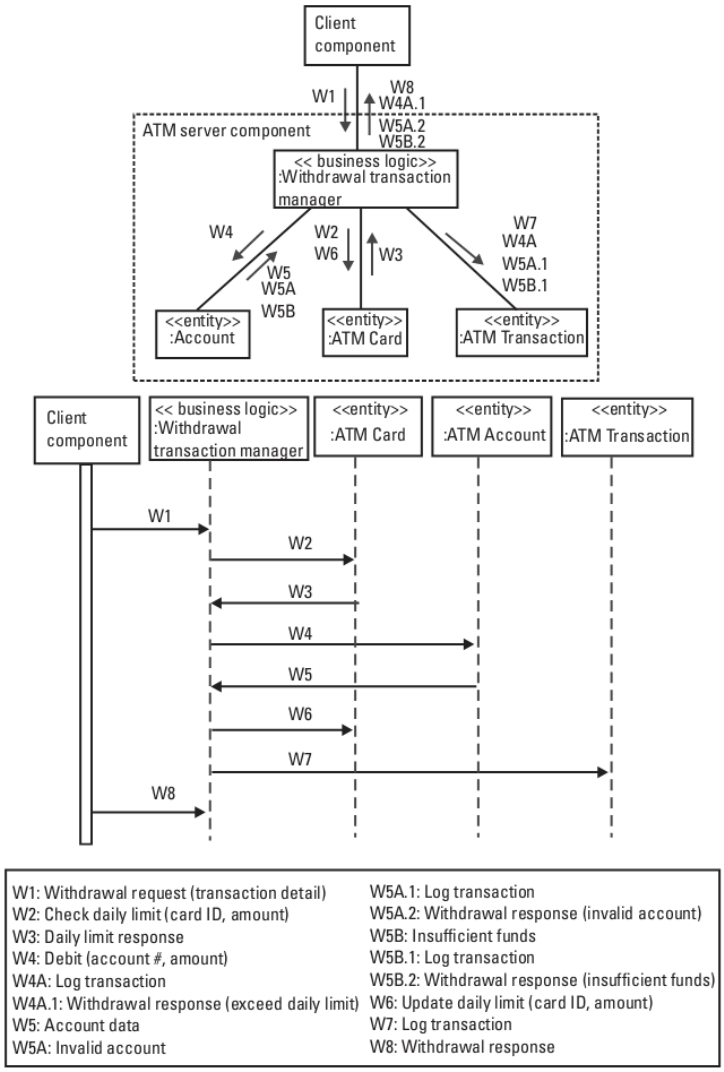
Sebagai pilihan dalam metode formal menggunakan sebuah bahasa pemodelan visual seperti UML dapat digunakan untuk mencoba dalam menangkap keperluan komponen dan desain kelas komponen dan *interface* lebih akurat (Cechich, 2003:131). UML *(Unified Modeling Language)* adalah sebuah bahasa pemodelan yang umum untuk menggambarkan sisi struktur *software* dan tingkah laku *software*. UML memiliki notasi grafis dan memungkinkan untuk perluasan dengan sebuah profile UML (Cash, 2012:30). UML adalah bahasa visual yang dapat digunakan untuk membuat sebuah pandangan dari arsitektur *software* (Cash, 2012:81).

Notasi yang digunakan dalam UML normalnya sanggup untuk mendeskripsikan sebuah desain secara lengkap. Hal tersebut dibutuhkan untuk menjelaskan rasional dari sebuah desain. Sebuah kumpulan *rational tool* merupakan hal utama yang disediakan sebuah *Unified Modeling*

Language (UML). Beberapa *tool* memungkinkan untuk lintas referensi diantara berbagai tipe diagram yang berbeda (Endres, 2003:59). Sebuah UML jenis diagram *use case* biasanya memperlihatkan setiap keadaan pengguna dalam sistem seperti sebuah *ellipse*, memperlihatkan pemeran utama untuk keadaan pengguna seperti gambar batang yang terhubung ke keadaan pengguna dengan sebuah garis yang disertai busur (Jalote, 2008:51). Sekarang banyak orang menggunakan UML untuk merepresentasikan sebuah arsitektur, yang menyediakan berbagai kemungkinan untuk menampilkan penjelasan penting dalam sebuah tampilan yang menyediakan kemampuan keterangan untuk mendukung dokumen. Kami percaya bahwa metode apapun dapat digunakan, selama terdapat sifat obyektif (Jalote, 2008:117).

Beberapa keuntungan dalam mengadopsi UML antara lain: (1) UML menyediakan informasi bersifat *high-level* yang merupakan karakteristik dari perilaku internal berbagai komponen, yang mana dapat berproses efisien dan efektif ketika kapan diuji; (2) UML dimunculkan sama seperti standar industri untuk notasi *software modeling*, dan berbagai varian diagram disediakan dari banyak penyedia komponen; (3) UML termasuk sebuah kumpulan model yang dapat menyediakan berbagai level kapasitas berbeda dan keakuratan untuk pemodelan komponen, dan dapat digunakan untuk memuaskan berbagai kebutuhan dalam dunia nyata; (4) UML adalah *extensible*, yang mana itu artinya jika ditambahkan peralatan temuan lain dan dianggap penting untuk basis komponen pada sistem *software*, maka dapat ditambahkan kedalam UML (Gao, 2003:78). Pada UML jenis *sequence*

diagrams, interaksi diatur berdasarkan waktu, sedangkan pada *collaboration diagrams*, informasi yang sama diperkenalkan bersama tatanan interaksi dalam aturan *numeric message sequence* (Gao, 2003:204).

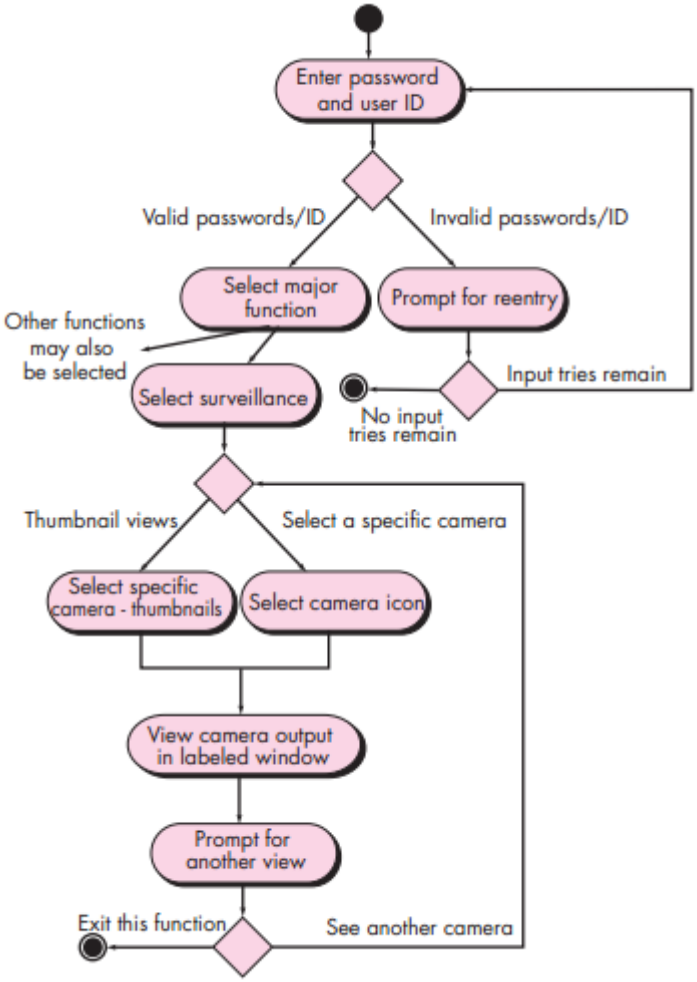


Gambar 62. Diagram UML *Collaboration* dan *Sequence* Sebuah *Mobile Banking*

Sebuah *use case* merepresentasikan sebuah urutan dari sebuah aksi atau langkah yang terjadi antara *user/actor* (pengguna) dan sistem ketika pengguna mencoba untuk memainkan satu dari fungsionalitas dalam sistem. Diagram *sequence* UML menggambarkan bagaimana sebuah obyek

berlangsung dalam sebuah *use case* (Deek, 2005:42). UML *collaboration diagrams* digunakan untuk menggambarkan sebuah interaksi antar obyek secara visual. UML *activity diagrams* digunakan untuk memahami sebuah logika dari *use case* dan penyelesaian perhitungan. Sebuah *state machine diagrams* menggambarkan perilaku obyek dalam merespon peristiwa *(event)* dan sama seperti fungsi dalam keadaan *internal* (Deek, 2005:46). UML (*Unified Modeling Language*) serupa dengan pemodelan grafis. UML merupakan standar bahasa untuk penetapan, penggambaran, perancangan, dan dokumentasi seluk beluk dari sistem perangkat lunak *(software)* dan proses model perhitungan dan sistem *nonsoftware* lain (Deek, 2005:223).

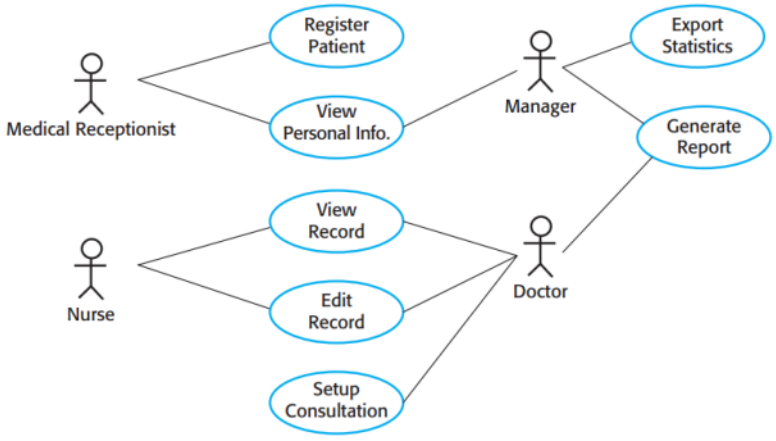
Varian UML memiliki ragam yang kompleks, tergantung pada karakteristik diagram apa yang ingin dibuat. UML memiliki varian tipe lain bernama *swimlane diagram*. UML *swimlane diagram* varian yang sangat berguna didalam diagram aktivitas dan memungkinkan untuk menggambarkan aliran uraian aktivitas dari *use case* serta dalam waktu yang sama menunjukkan aktor yang mana atau kelas analisis untuk uraian aksi dari aktivitas. UML diagram *swimlane* merepresentasikan aliran sebuah aksi dan keputusan yang menunjukkan setiap performa aktor/pengguna (Pressman, 2010:162). Sebagai contoh ada tiga kelas analisis, yaitu: *Homeowner*, *Camera*, dan *Interface*. Sebuah sistem menggambarkan sebuah aktivitas untuk akses dengan kamera pemantauan melalui fungsi pandangan kamera via tampilan internet. Sisi pengguna *(user)* dapat untuk masuk *userID* dan *password* yang dibatasi oleh waktu.y



Gambar 63. Diagram UML *Swimlane* pada Akses Sebuah *Hardware*

Notasi pada UML *Class Diagram* memiliki penjabaran *subset*, diantaranya adalah: *Interface inheritance, Inheritance, Dependencies, Simple association* dan *Composition* (Morris, 2010:459-460). Diagram *use case* sekarang ini menjadi sebuah fitur pokok dalam *Unified Modeling Language* (UML). Pada format yang paling sederhana, sebuah *use case* mengidentifikasikan aktor *(user)* yang melibatkan diri dalam interaksi dan nama dari tipe interaksi. Deskripsi teks pada *use case* atau model grafis sama

seperti UML *sequence* atau *state charts*. Keadaan pengguna merupakan dokumentasi menggunakan sebuah diagram *high-level use case*. Kumpulan dari *use case* merepresentasikan semua interaksi yang mungkin dan akan diuraikan didalam keperluan sistem. Diagram *use case* mengidentifikasikan interaksi individu antara sebuah sistem dan *user* bersangkutan atau dengan sistem lain (Sommerville, 2011:106-107).



Gambar 64. Diagram UML *Use Case* pada *Medical Information System*

Skenario dan *use case* merupakan teknik yang efektif untuk keperluan pengungkapan dari siapa pemegang peran *(stakeholder)* yang berinteraksi secara langsung dengan sistem. Setiap tipe dari interaksi dapat digambarkan melalui sebuah *use case*. Sebuah *use case* fokus dalam hal interaksi dengan sistem, dan kurang efektif untuk pengungkapan paksa atau perhitungan tingkat tinggi dan keperluan *nonfunctional* atau untuk keperluan *discovering domain* (Sommerville, 2011:108).

UML (*Unified Modeling Language*) memiliki pemodelan yang dapat dihidupkan seperti praktik yang cukup baik didalam pengembangan *software*.

UML menyediakan sebuah bahasa yang komperhensif untuk pemodelan. Sifat ekspresif, fleksibel, dan kegunaan UML sama halnya seperti bahasa pemrograman *(programming languages)* yang digunakan untuk *coding*. UML dapat membuat ketepatan untuk melihat *object-oriented models* seperti realisasi sebuah pembuatan kode dalam pemodelan bisnis, *workflow design*, dan *enterprise architecture* (Evitts, 2000:26). UML adalah tentang pemodelan dan pembangunan model, bukan tentang menyusun sebuah kode, *project management*, atau *production support*. UML dapat digunakan untuk pemodelan bisnis serta untuk pembuatan model untuk berbagai sistem dan bukan *software systems* (Evitts, 2000:33).

Komponen model yang ada didalam UML adalah representasi utuh dari sebuah sistem berdasarkan sudut pandang yang teliti. Sistem secara logis tersusun dari banyak model yang mendekati independen, merepresentasikan banyak sudut pandang yang berbeda dan bisa jadi secara fisik tersusun dari banyak subsistem independen. Sebuah sistem sendiri secara implisit direpresentasikan dari sebuah model *top-level* dengan berbagai model tambahan yang merepresentasikan gambaran khusus. Setiap model adalah terbuat dari berbagai diagram dan teks (Evitts, 2000:39). Model dari UML adalah sebuah gambaran umum, sebuah representasi utuh dari sebuah sistem fisik. Model adalah tentang berbagai hal, *relationships*, *behaviors*, dan interaksi didalam sistem (Evitts, 2000:40).

B. Rekayasa Perangkat Lunak

Perangkat lunak *(software)* kini telah menjadi sebuah kebutuhan yang berpengaruh dalam kehidupan manusia. Pengambilan keputusan di dalam dunia bisnis, medis, penelitian keilmuan, transportasi, telekomunikasi, militer, proses industri, semuanya tidak lepas dari peran dan sistem kerja perangkat lunak. Kehidupan dunia modern tidak dapat lepas dari teknologi perangkat lunak, demikian pula dalam dunia pendidikan.

Pemahaman mengenai perangkat lunak dalam sumber yang dikutip dari (Pressman, 2012:10), menguraikan bahwa perangkat lunak adalah; *(1) perintah (program komputer) yang bila dieksekusi memberikan fungsi dan unjuk kerja seperti yang diinginkan, (2) struktur data yang memungkinkan program memanipulasi informasi secara proporsional, dan (3) dokumen yang menggambarkan operasi dan kegunaan program*.

Karakteristik perangkat lunak sangat beragam, hal ini seiring dengan proses kreatif manusia (analisis, desain, konstruksi, pengujian). Perangkat lunak lebih kepada elemen logika dan algoritma, bukan merupakan elemen sistem fisik. Sehingga karakteristik perangkat lunak *(software)* memiliki ciri yang sangat berbeda dengan perangkat keras *(hardware)*. Kompleksitas perangkat lunak dapat dikelompokkan dalam beberapa kategori, diantaranya adalah sebagai berikut (Pressman, 2010:7-8);

- **Perangkat Lunak Sistem *(System Software)*** : merupakan perangkat lunak yang memiliki sistem kerja erat kaitannya dengan perangkat keras, misalnya sistem operasi, *driver*, prosesor telekomunikasi, utilitas pengaturan *file*, dll.

Perangkat lunak jenis ini dirancang untuk melayani program-program yang lain.

- **Perangkat Lunak Aplikasi *(Application Software)*** : perangkat lunak yang digunakan dalam menjawab kebutuhan bisnis yang terinci. Perangkat lunak jenis ini melakukan pemrosesan data bisnis atau data teknis yang mendukung berjalannya operasi bisnis atau pengaturan dan pengambilan keputusan secara teknis. Selain untuk kebutuhan pemrosesan data konvensional, perangkat lunak jenis ini juga melakukan fungsi-fungsi pengendalian bisnis secara aktual. Sebagai contoh pada transaksi penjualan (*trading* bursa saham), pemrosesan kendali produksi pada pabrik secara aktual.
- **Perangkat Lunak Rekayasa/Ilmiah *(Engineering/Scientific Software)*** : perangkat lunak yang memiliki ciri khas berupa algoritma "*number crunching*". Jangkauan perangkat lunak jenis ini meliputi keilmuan teknik astronomi, otomotif, pesawat ruang angkasa, pabrik otomatisasi dan sejenisnya. Sebagai contoh perangkat lunak jenis ini antara lain, *Computer-aided desain*, simulasi sistem, dan *software* interaktif yang lain.
- **Perangkat Lunak Tertanam *(Embedded Software)*** : perangkat lunak yang berada dalam suatu produk atau sistem dan digunakan untuk menjalankan dan mengendalikan berbagai fitur dan fungsi bagi pengguna akhir dan bagi sistem itu sendiri. Misalnya *embedded software* untuk *key pad control* , tombol kontrol *microwave*, kontrol bahan bakar kendaraan, sistem rem, dan sejenisnya.

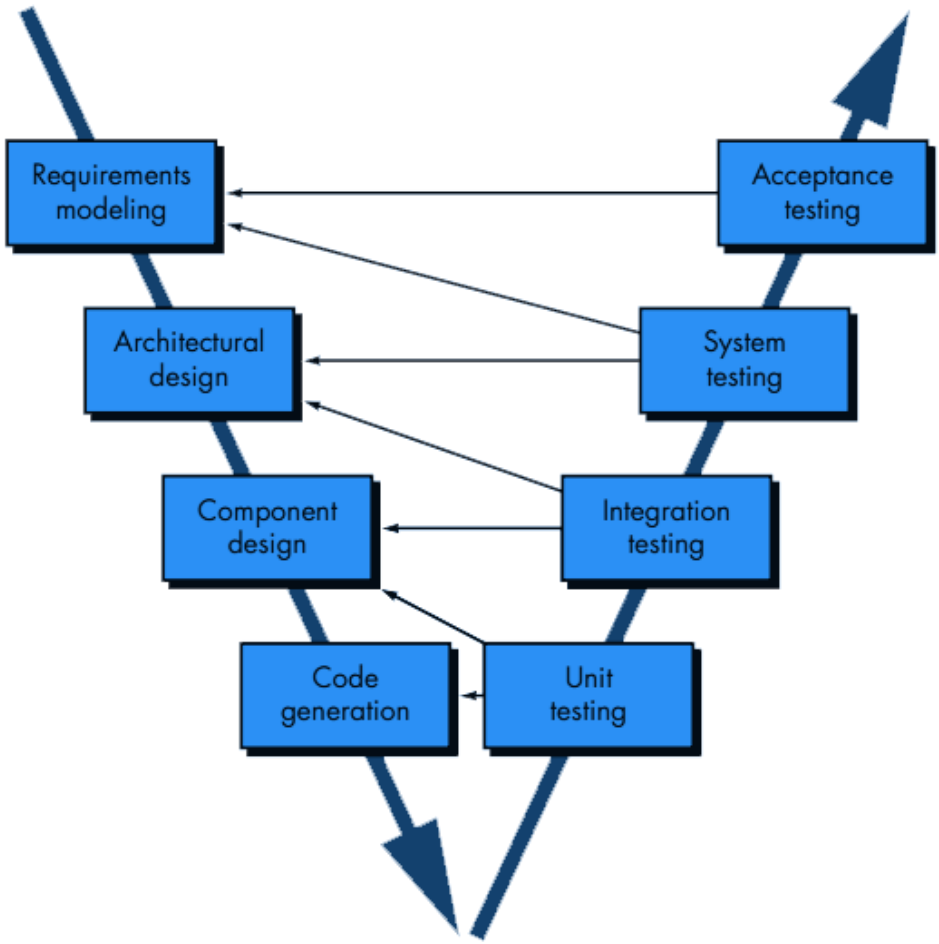
- **Perangkat Lunak Produk *(Product-line Software)*** : perangkat lunak yang dirancang untuk menyediakan kemampuan khusus untuk digunakan oleh pelanggan yang beranekaragam. Perangkat lunak jenis ini dapat berkonsentrasi pada pasar tertentu yang bersifat terbatas. Contohnya perangkat lunak pengolah kata, lembar kerja, grafik komputer, multimedia, hiburan, pengaturan basis data, aplikasi personal dan keuangan bisnis.
- **Aplikasi Web *(Web Applications)*** : perangkat lunak yang yang beroperasi pada jaringan komputer, yang menyediakan sederetan aplikasi-aplikasi luas. Aplikasi ini tidak hanya sekedar berupa sekumpulan *file hypertext* yang saling berhubungan untuk menunjukkan berbagai informasi tertentu dengan berbagai grafis, namun sebuah sajian komputasi yang canggih yang terintegrasi dengan basis data yang dimiliki oleh perusahaan dan juga terintegrasi dengan berbagai aplikasi bisnis lain untuk pengguna akhir.
- **Perangkat Lunak Kecerdasan Buatan *(Artificial Intelligence Software)*** : perangkat lunak yang menggunakan algoritma non-numeris untuk memecahkan masalah kompleks yang tidak dapat diselesaikan secara perhitungan dan analisis langsung. Area perangkat lunak kecerdasan buatan yang aktif adalah sistem pakar, disebut juga sistem berbasis ilmu pengetahuan. Aplikasi lainnya yang termasuk dalam kecerdasan buatan adalah pengenalan pola *(image* dan *voice)*, serta jaringan syaraf tiruan *(artificial neural network)* yakni perangkat lunak yang memiliki sistem kerja layaknya syaraf otak, yang berdasarkan pengalaman belajar sebelumnya.

1. Model Pengembangan Perangkat Lunak

Teknik pengembangan sebuah perangkat lunak atau lazim disebut *Software Development Life Cycle (SDLC)* terdiri dari berbagai model, diantaranya; *Waterfall Development Life Cycle (The Classical SDLC), Prototyping, Rapid Application Development* (Everett, 2007:29). Model *waterfall* memiliki varian model yang bernama *v-model* (Pressman, 2010:39). Jenis *v-model* memiliki alur produksi perangkat lunak hingga pada tahap pengujian yang spesifik. Perbedaan lain *v-model* dengan *waterfall model* adalah terletak pada alur produksinya, pada *waterfall model* memiliki alur kerja sekuensial sedangkan pada *v-model* bercabang membentuk pola kerja pada sisi pembuatan *software* dan sisi tahap detail pengujiannya. Karakteristik perbedaan kedua model tersebut yang membuat waterfall *model* disebut pula dengan istilah *classic life cycle*. Model *waterfall* disebut pula paradigma kuno pada sebuah teknik rekayasa perangkat lunak *(software engineering)* (Pressman, 2010:40). Model *waterfall* berperan dasar dalam proses menjabarkan, mengembangkan validasi dan evaluasi pembuatan perangkat lunak hingga implementasi, pengujian (Sommerville, 2011:29).

Model pengembangan *v-model* merupakan model *Software Development Life Cycle (SDLC)* yang digunakan dalam pengembangan *software* simulasi teknik digital "DigiChip" pada penelitian ini. Kelebihan dari *v-model* dibandingkan dengan SDLC yang lain diantaranya memiliki ketahanan terhadap perubahan yang diinginkan, validasi yang tepat untuk mengambil tempat disetiap tahap, *engineer* jika sekaligus berperan sebagai

tester maka terlibat langsung sejak awal pembuatan produk (Balaji, 2012:4). Pada *v-model* memiliki alur hubungan kuat antara proses pembuatan perangkat lunak *(software engineering)* dengan proses pengujian perangkat lunak *(software testing).* Alur proses rekayasa perangkat lunak dibagi dalam dua garis besar diantaranya tahap Verifikasi dan Validasi. Penjelasan secara detail dapat dilihat pada dilihat pada gambar skema *v-model* berikut ini;



Gambar 65. Tahapan *V-model*
(sumber: Buku *"Software Engineering - A Practitioner's Approach 7th Edition"*, Roger S. Pressman)

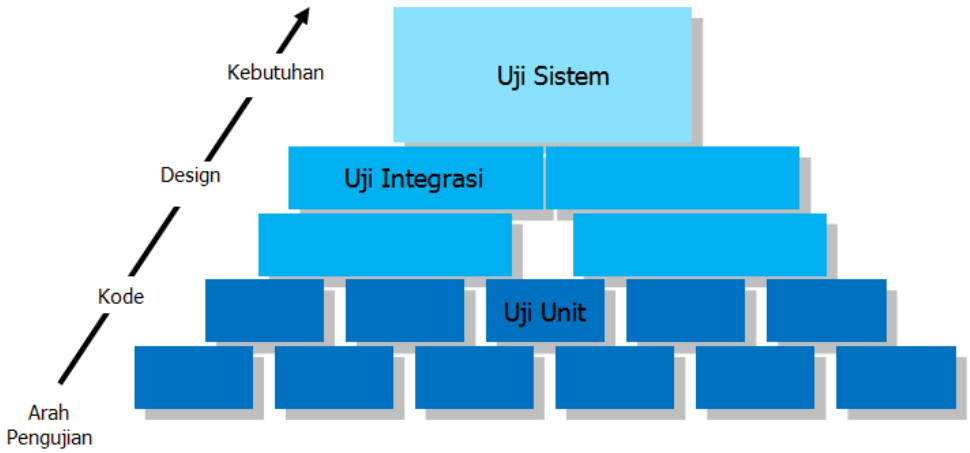
Gambar tersebut menjelaskan tentang bagaimana sebuah proses produksi perangkat lunak dari tahap awal hingga akhir harus dilakukan. Jalur kiri merupakan jalur verifikasi, dimana sebuah produk perangkat lunak akan dibuat tahap demi tahap sesuai petunjuk gambaran tanda panah yang menurun tersebut. Jalur kanan adalah jalur validasi, gambaran berbagai tahapan pengujian perangkat lunak yang harus dilakukan.

Tahap verifikasi merupakan serangkaian tahapan yang terdiri dari proses analisis kebutuhan akan spesifikasi perangkat lunak yang akan dibuat, proses penentuan *design* yang sesuai dengan kebutuhan *user*, hingga sampai pada tahap pembangunan desain dan kode instruksi program. Serangkain tahapan verifikasi tersebut diuraikan sebagai berikut:

1. *Requirement modeling*, merupakan tahapan untuk mencari informasi kebutuhan perangkat lunak dengan cara konsultasi dengan pengguna. Sebuah *software* yang dibuat idealnya menyesuaikan dengan kebutuhan *user*/pengguna. Tahapan ini mendefinisikan secara rinci fungsi-fungsi, batasan, spesifikasi, dan tujuan dari perangkat lunak yang dibuat.
2. *Architectural design*, merupakan tahapan perancangan sistem yang berfokus pada desain perangkat lunak, struktur data, antarmuka pengguna, dan prosedur pengkodean. Tentu hal ini menyangkut pada aspek *software* dan *hardware* serta detail teknologi yang akan digunakan sebagai media untuk mewujudkan kebutuhan pengguna. Istilah lain pada tahapan ini sering disebut dengan *High Level Design.*

3. *Component design*, merupakan tahapan perancangan perangkat lunak menjadi serangkaian unit program. Bagian yang tentunya memiliki porsi pengerjaan lebih kecil dari *architectural design.* Desain dan algoritma pembuatan kode program tentunya sesuai dengan desain yang telah dibuat pada tahap sebelumnya yaitu tahap *architectural design*. Istilah lain pada tahapan ini sering disebut dengan *Low Level Design.*
4. *Code generation*, merupakan tahapan yang berarti tentang proses pembangunan sebuah kode instruksi program. Hal ini berhubungan erat sekali dengan karakter bahasa program yang digunakan dan detail algoritma program yang ingin diterapkan.

Tahap validasi merupakan serangkaian proses pengujian produk perangkat lunak dari berbagai aspek, dimulai dari lingkup yang terkecil hingga lingkup yang komperhensif dari perangkat lunak yang bersangkutan (Rosa & Shalahuddin, 2011:211). Tahap validasi terdiri dari berbagi proses pengujian perangkat lunak, diantaranya;



Gambar 66. Hirarki Pengujian Sistem

1. *Unit testing*, merupakan tahapan untuk menguji setiap bagian terkecil dari perangkat lunak (unit) telah sesuai dengan desain dan spesifikasi yang diharapkan. Proses ini dilakukan oleh *engineer/programmer* yang bersangkutan untuk mengetahui kesalahan pada aspek *source-code* dan *syntac* program. Karena seringkali dalam pembuatan instruksi program, seorang pengembang perangkat lunak terjadi kesalahan pada aspek *syntac* program. Demikian pula penggunaan *source-code* dan *syntac* yang tidak efektif pada algoritma program akan menimbulkan kapasitas file program yang berlebih dan berat dalam memproses perintah. Pengujian unit dilakukan karena pada dasarnya sebuah perangkat lunak yang dibuat akan terdiri dari berbagai unit algoritma dan *syntac*. Oleh karena itu strategi pengujian unit per unit perlu untuk dilakukan.
2. *Integration testing*, merupakan tahapan untuk menguji perangkat lunak yang telah mengalami penggabungan antar unit/bagian. *Integration* berarti penggabungan, dapat dimaknai yaitu sebuah pengujian gabungan. Dalam penerapannya memang sebuah strategi pengujian perangkat lunak dengan menggabungkan antar unit pada perangkat lunak untuk melakukan kinerja dengan baik. *Integration testing* dilakukan untuk mengetahui interaksi antar unit perangkat lunak apakah mampu beroperasi dengan baik. *Integration testing* memiliki dua cara, yaitu *bottom-up integration testing* dan *top-down integration testing*. *Bottom-up integration testing* dimulai dari *unit testing* dan berlanjut ke pengujian yang lebih tinggi areanya. *Top-down integration testing*

sebaliknya, yakni pengujian berawal dari modul yang lebih tinggi areanya, berlanjut ke pengujian unit yang lebih rendah dibawahnya. Dalam proses pembuatan perangkat lunak, pengujian *bottom-up* lazimnya dilakukan terlebih dahulu dan berlanjut pengujian *top-down*. Proses ini juga dilakukan oleh *engineer/programmer* yang bersangkutan.

3. *System testing*, merupakan tahapan proses pengujian perangkat lunak ketika antar bagian telah tergabung dalam satu kesatuan sistem yang utuh. *System testing* merupakan pengujian keseluruhan kinerja dari produk perangkat lunak yang telah dibuat. Proses ini dilakukan oleh *engineer/programmer* yang bersangkutan dan lingkup para ahli, baik ahli pemrograman maupun algoritma.
4. *Acceptance testing,* merupakan tahapan proses pengujian perangkat lunak yang dilakukan dari sisi *user/customer*. Hal ini sangat berhubungan dengan kemudahan dan kenyamanan ketika mereka mengoperasikan perangkat lunak tersebut. *Acceptance testing* dilakukan pengembang perangkat lunak guna mengetahui produk yang dibuat telah sesuai dengan kebutuhan pengguna. Metode yang digunakan dalam *acceptance testing* adalah metode pengujian *black-box*. Karena dalam pengujian ini tidak menguji aspek algoritma program dan *syntac*, namun lebih kepada pengujian kesesuaian perangkat lunak yang dibuat agar sesuai dengan kebutuhan *user* (pengguna).

2. Pembangunan Perangkat Lunak *(Software Construction)*

Pembangunan perangkat lunak merupakan pekerjaan detail dari proses pembuatan perangkat lunak yang meliputi proses desain, konsep, pemprograman, integrasi program, *debugging* dan *run-test* program. Dalam proses pembangunan perangkat lunak, *route map* yang dijadikan pedoman adalah analisa kebutuhan pengguna. Hal ini tentu mengarah pada produk yang dihasilkan agar lebih memiliki nilai *usable* atau tepatguna. Aspek yang paling menonjol dalam proses pembangunan perangkat lunak adalah sistematika dan algoritma dalam membangun sebuah sistem yang diinginkan. Terlebih sebuah sistem yang ingin dibangun memiliki ruang lingkup yang besar, tentu hal ini diperlukan kemampuan lebih detail dalam membangun sistem yang bersangkutan. Pada akhirnya sebuah produk yang dihasilkan diharapkan memiliki nilai yang sistematis yang tinggi dan sesuai kebutuhan pengguna.

Terdapat empat aspek penting dalam proses pembangunan sebuah produk perangkat lunak (Rizky, 2011:196), diantaranya;

1. Meminimalkan kompleksitas

   Di dalam proses pembangunan perangkat lunak, selalu terdapat dua sudut pandang yang berbeda, yakni dari sudut pandang pengembang perangkat lunak dan dari sudut pandang pengguna. Umumnya pengembang perangkat lunak khususnya programmer, memandang sebuah proses bisnis sebagai sebuah proses kompleks dalam kegiatan pemprograman.

Salah satu cara untuk meminimalkan proses yang kompleks yaitu dengan pemprograman berorientasi obyek sehingga terbentuk sebuah standar.

2. Mengantisipasi perubahan

   Khususnya dalam sebuah proyek pengembangan perangkat lunak yang kompleks dan berskala besar serta memakan waktu pengerjaan cukup lama, perubahan spesifikasi sangat mungkin terjadi. Maka pengembang tidak boleh bersifat defensif tapi justru sebaliknya, yaitu bersifat antisipatif.

3. Verifikasi

   Verifikasi disini dimaksud dalam ruang lingkup verifikasi di level aktivitas pemprograman. Sebagai contoh ketika dalam sebuah proyek pengembangan yang berupa tim atau melibatkan banyak programer, maka dalam proses pembangunan perangkat lunak harus terjalin kerjasama yang solid agar perangkat lunak yang dibangun benar-benar terbebas dari kesalahan.

4. Menetapkan standar

   Standar dalam pembangunan perangkat lunak seharusnya ditetapkan sejak proses analisa dan desain. Misalnya dengan menggunakan notasi UML *(Unified Modeling Language)* sejak awal proses analisa dan perancangan, sehingga pada saat aktivitas pemprograman dilakukan dapat secara mudah diimplementasikan oleh para programer.

3. Pengujian Perangkat Lunak *(Software Testing)*

Proses pengujian perangkat lunak adalah sebuah aktivitas untuk menjaga kualitas perangkat lunak sejak awal proses pembangunan perangkat lunak tersebut. Hal ini dilakukan guna memastikan kualitas perangkat lunak yang dibuat apakah telah sesuai dengan kebutuhan teknis di awal desain produk yang telah ditetapkan. Proses pengujian perangkat lunak tentunya dilakukan berkesinambungan hingga diperoleh kesempurnaan yang diharapkan. Sangatlah salah jika sebuah tim pengembang perangkat lunak beranggapan bahwa perangkat lunak yang dibangun oleh mereka telah sempurna. Bahkan vendor raksasa seperti Microsoft, Oracle maupun IBM tidak pernah berfikir perangkat lunak telah mencapai level sempurna (Rizky, 2011:235).

Definisi paragraf diatas dapat disimpulkan bahwa terdapat dua kata kunci, yaitu kebutuhan yang harus dipenuhi perangkat lunak *(software requirement)* dan kualitas perangkat lunak *(software quality)*. Berbagai tahapan yang harus dilakukan dalam pengujian perangkat lunak diantaranya *verification* (verifikasi) and *validation* (validasi), (Galin, 2004:133) berikut adalah definisi lengkapnya;

1. Verifikasi

   Merupakan sebuah proses pemeriksaan perangkat lunak dari sisi pengembang, apakah produk yang dibuat telah sesuai dengan desain awal proses perancangan produk. Hal ini tentu menyangkut pada aspek desain grafis, instruksi pemprograman, algoritma dll. Pada tahap ini produk

terus mengalami penyempurnaan hingga sesuai dengan rancangan desain awal yang diharapkan.

2. Validasi

   Merupakan proses pemeriksaan perangkat lunak dari pihak *requirements*, atau dalam hal ini adalah sisi pengguna produk. Sebagai sarana untuk mendapatkan *feedback* dari sisi pengguna, bahwa apakah produk perangkat lunak yang telah dibuat sudah sesuai dengan ekspektasi pengguna. Dalam tahapan ini pula produk perangkat lunak terus mengalami penyempurnaan dan perbaikan hingga sesuai dengan kebutuhan pengguna.

Proses pengujian perangkat lunak secara berkelanjutan dari tahap demi tahap adalah sangat prioritas. Sebagai upaya untuk tindakan preventif dari munculnya berbagai permasalahan selama proses pembuatan perangkat lunak, antara lain adalah kesalahan *(error)*, kerusakan *(fault)*, kegagalan *(failure)* dan kecelakaan *(incident)*. Standar yang harus dipenuhi dari sebuah produk perangkat lunak adalah diantaranya terbebas dari masalah *failure*, *fault*, *error* dan *incident* (Rizky, 2011:241).

1. *Failure*

   Sebuah kegagalan perangkat lunak dalam melakukan proses yang seharusnya menjadi kebutuhan perangkat lunak tersebut. *Failure* merupakan efek terakhir ari kejadian *fault.*

2. *Fault*

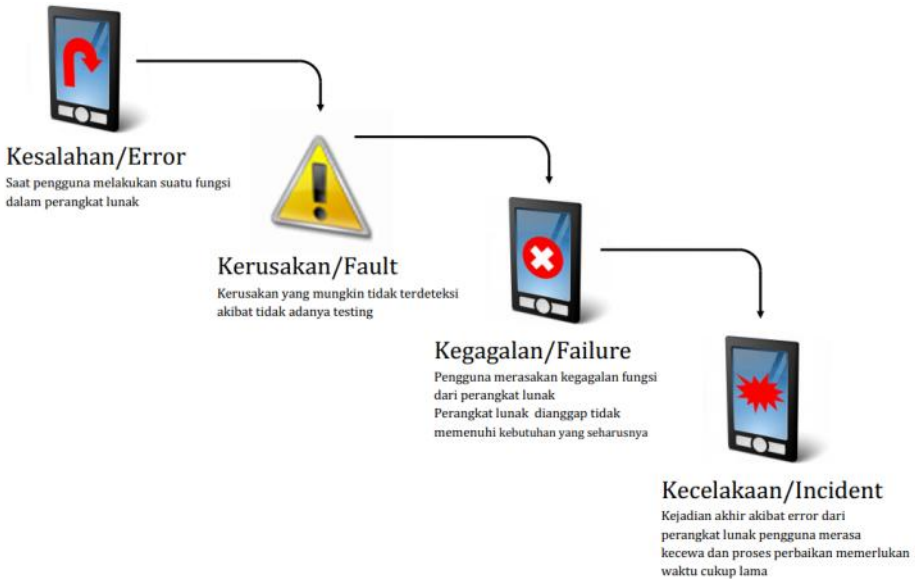
   Dalam lingkungan pengujian perangkat lunak sebuah *fault* bisa jadi tidak akan terdeteksi kecuali jika suatu tindakan akan membuatnya muncul dalam sebuah proses perangkat lunak. Ini berarti *fault* adalah potensi dari terjadinya sebuah *error,* dan saat *error* tersebut terjadi akibat tindakan pengguna, maka kan timbul *failure* atau kegagalan dalam proses perangkat lunak.

3. *Error*

   Adalah sebuah keadaan dari sistem yang disebabkan oleh tindakan pengguna yang pada akhirnya menyebabkan kegagalan dalam pelaksanaan sebuah fungsi perangkat lunak. *Error* atau kesalahan adalah akibat dari adanya *fault* atau kerusakan yang kemudian dipicu oleh perilaku pengguna.

4. *Incident*

   Dalam konteks pengujian perangkat lunak, *incident* merupakan hasil akhir yang terjadi akibat dari *error* yang berkelanjutan dan tidak diperbaiki atau terdeteksi dalam proses pengembangan perangkat lunak. Hal tersebut harus segera dapat diatasi, walaupun kondisi yang demikian biasanya akan banyak memakan waktu untuk proses perbaikan. Karena apabila masalah demikian tidak terdeteksi sumber permasalahannya, pengguna akan memberikan vonis terhadap kualitas perangkat lunak secara keseluruhan memiliki kinerja buruk. Padahal bisa jadi *incident* tersebut muncul karena hanya dari beberapa unit sistem yang bermasalah.



Gambar 67. Kegagalan Standard Perangkat Lunak

4. Jaminan Kualitas Perangkat Lunak *(Software Quality Assurance)*

Kualitas tidak hanya sekedar penilaian ketika sesuatu dipandang lebih baik dari yang lainnya, namun lebih kompleks dapat mewakili kebutuhan dan selera pengguna hingga pada aspek efektivitas dari kehadiran sebuah produk yang bersangkutan. Sebuah perangkat lunak yang berkualitas dapat didefinisikan sebagai suatu proses perangkat lunak yang efektif diterapkan dalam arti kata proses perangkat lunak yang menyediakan nilai yang dapat diukur untuk mereka yang menghasilkan dan untuk mereka yang menghasilkannya (Pressman, 2012:485). Pada definisi tersebut memberikan penekanan pada 3 aspek yang penting, diantaranya sebagai berikut :

a. Suatu *proses perangkat lunak efektif* yang menetapkan infrastruktur yang mendukung setiap usaha untuk mengembangkan produk perangkat lunak yang berkualitas tinggi. Aspek manajemen proses sebagai upaya untuk

menjauhkan produk dari berbagai permasalahan produksi, perubahan teknis produksi dan faktor pemicu terciptanya kualitas produk perangkat lunak yang rendah.

b. Suatu *produk yang bermanfaat* memiliki didalamnya isi, fungsi-fungsi, serta fitur-fitur yang diinginkan oleh para pengguna akhir serta bersifat handal dan bebas dari kesalahan-kesalahan. Produk tersebut memenuhi kebutuhan pengguna secara detail, seperti kemudahan penggunaan dan faktor praktis yang lain.

c. Melakukan *penambahan nilai pada produsen maupun pengguna* produk perangkat lunak. Sebuah produk perangkat lunak yang baik adalah mampu memenuhi kebutuhan pengguna akan aktivitas tertentu. Perangkat lunak yang berkualitas tinggi juga membutuhkan usaha pemeliharaan yang kecil serta memiliki berbagai kesalahan yang harus diperbaiki dalam jumlah kecil pula. Hal ini memungkinkan perekayasa perangkat lunak menghabiskan waktu yang sangat berharga lebih banyak untuk proses pembuatan aplikasi.

Proses pembuatan perangkat lunak yang berkualitas tidak lepas dari serangkaian proses kendali kualitas. Aspek kendali kualitas merupakan aktivitas rekayasa perangkat lunak yang bisa digunakan untuk memastikan apakah masing-masing produk kerja yang dihasilkan sesuai dengan sasaran kualitas yang telah ditetapkan sebelumnya (Pressman, 2012:500). Hasil desain perlu ditinjau ulang sebelum berlanjut ke tahap pengujian perangkat lunak. Kode-kode program, logika memprograman hingga sampai pada aspek

antarmuka pengguna *(user interface/UI)* perlu dievaluasi ulang terhadap kegagalan proses yang lebih parah lagi ketika tahap pengujian perangkat lunak. Sebuah kendali kualitas semata-mata dilakukan untuk terwujudnya sebuah jaminan kualitas perangkat lunak yang baik, walaupun dalam kenyataan kita tahu bahwa sebuah perangkat lunak kualitas yang tinggi memerlukan waktu dan biaya yang sangat tinggi pula dalam proses pembuatannya (Pressman, 2012:493).

5. Teknik Kualitas Taguchi *(Taguchi's Quality Engineering)*

Parameter kualitas sebuah perangkat lunak dalam sudut pandang lain berhubungan erat dengan faktor efisiensi. Sebuah produk manufaktur dalam hal ini adalah perangkat lunak harus memiliki kualitas desain yang efisien atau dapat menekan segala jenis biaya dan *resources* produksi, hal ini kita kenal dengan metode *robust design*. Metode *'Robust Design'* atau *'Robust Engineering'* ditemukan oleh insinyur asal jepang bernama Dr. Genichi Taguchi pada tahun 1950-an, sebagai jawaban atas ketatnya persaingan industri manufaktur dunia kala itu. Hal tersebut bersamaan dengan era kebangkitan industrialisasi jepang setelah berbagai infrastruktur negeri sakura tersebut dihancurkan oleh amerika, tepatnya di Heroshima dan Nagasaki. Dalam kondisi yang harus memulai dari nol, sebuah kata kunci efektif dan efisien dalam aspek *cost* maupun *resources* adalah sebuah konsep yang sangat membantu.

Metode *robust engineering* biasa kita kenal dengan istilah *'Taguchi Metode'*. Genichi Taguchi memiliki latarbelakang ilmu teknik serta

mendalami ilmu statistik khususnya untuk analisa dunia industri. Metode *'Robust Engineering'* Taguchi memiliki filosofi perbaikan kualitas secara terperinci menekankan pada reduksi variasi faktor. Sehingga memberikan dampak yang efisien pada aspek *cost* maupun *resources* proses produksi *(manufacturing process).* Tentu konsep ini begitu tepat diadopsi oleh negara indonesia yang relatif minim *cost, resources* maupun infrastruktur pendukung proses produksi.

Metode Taguchi memiliki definisi sebagai upaya mewujudkan kualitas pada aspek efisiensi dan berbagai faktor *resources* selama proses produksi. Hal ini seperti yang dikemukakan Taguchi dalam buku '*Computer-Based Robust Engineering'* , beliau mengatakan bahwa;

> *Robustness can be defined as designing a product in such a way that the level of its performance under various customer usage conditions is same as under nominal conditions. Robust engineering methods are intended as cost-effective methods to improve the performance of a product by reducing its variability in customer usage conditions. Because they are intended to improve companies' competitive position, these methods have attracted the attention of many industries and academic communities across the globe.* (Taguchi, 2005:3).

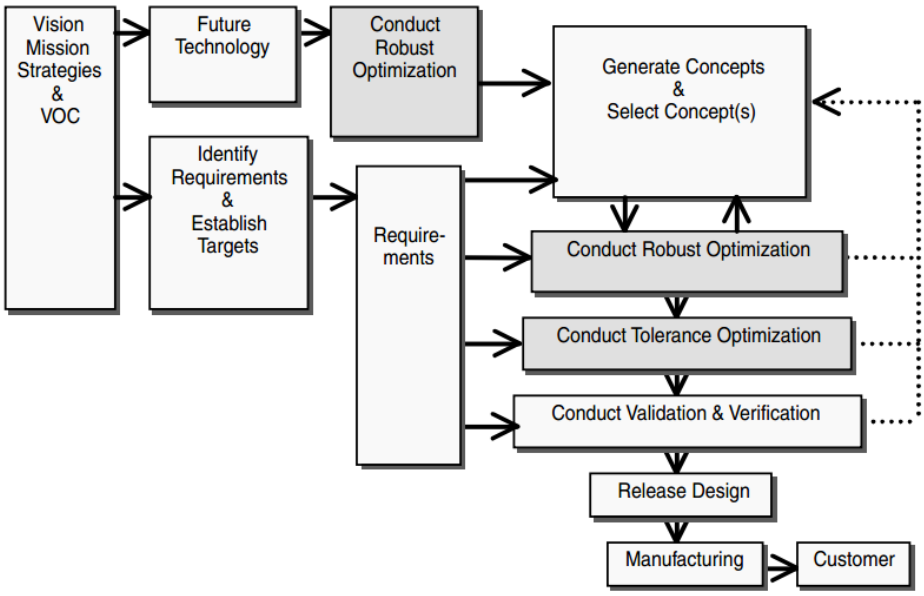
Parameter kualitas pada aspek ketahanan sebuah produk dapat didefinisikan sebagai merancang sebuah produk sedemikian rupa bahwa tingkat kinerja dalam berbagai kondisi pemakaian pelanggan adalah sama di bawah kondisi nominal. Metode rekayasa yang kuat dimaksudkan sebagai metode hemat biaya untuk meningkatkan kinerja suatu produk dengan mengurangi variabilitas dalam kondisi penggunaan pelanggan. Karena mereka dimaksudkan untuk meningkatkan posisi kompetitif perusahaan, metode ini

telah menarik perhatian banyak industri dan akademisi masyarakat di seluruh dunia.

Kualitas dalam kontes rekayasa ketahanan produk *(robust engineering)* di klasifikasikan menjadi dua jenis, diantaranya adalah; (1) Kualitas pada sisi pengguna produk *(customer-driven quality)*, dan (2) Kualitas pada sisi pembuat produk *(engineered quality)*. Parameter kualitas pada sisi *customer-driven* lebih kepada kesesuaian produk yang dibuat dengan kebutuhan segmen *market* atau pengguna, seperti fitur fasilitas yang terdapat pada produk, desain, warna, ukuran, tampilan dan fungsi *(functionality)*. Parameter *customer-driven quality* diarahkan untuk hal kenyamanan pengguna dalam menggunakan produk dan menjaga produk yang dibuat dari berbagai ketidaknyamanan pada sisi pengguna, seperti kegagalan sistem, kerumitan tampilan *UI (user interface)* dan berbagai bentuk *noise*. Pada parameter *engineered quality* , Taguchi menjelaskan ada 3 faktor yang menjadi permasalahan dalam proses pembuatan produk, diantaranya adalah; (1) *Various usage conditions - Environmental conditions*, (2) *Deterioration and wear - Degradation over time,* (3) *Individual differences - Manufacturing imperfections.*

Metode *robust engineering* atau Metode Taguchi bertujuan untuk meningkatkan kualitas pada sisi pembuatan produk *(manufacturing process)*. Perancangan produk baru yang memilik nilai optimasi untuk ketahanan produk dapat dicapai melalui tiga tahapan;

1. *Concept design*, merupakan pondasi awal sebuah produk diawali berdasarkan ide-ide, metode baru, inovasi baru, pemikiran teknis, sebagai upaya untuk memberikan nilai tambah dan peningkatan kualitas produk yang akan dibuat. Sebuah konsep bisa saja berasal dari percobaan sebelumnya, pengetahuan teknik, inovasi berdasarkan perubahan baru dan berbagai inspirasi non-plagiasi lain.
2. *Parameter design,* merupakan tahap pembuatan secara fisik atau prototipe berdasarkan pada perhitungan matematis ditahap percobaan sebelumnya, untuk mengetahui gambaran secara statistik mengenai produk yang akan dihasilkan. Tujuannya adalah mendapatkan informasi setting parameter yang akan dijadikan acuan performansi rata-rata pada target atau pengguna, dan menentukan berbagai faktor gangguan sistem *(noise factors)* pada aspek variatif produk yang akan dibuat.
3. *Tolerance design,* merupakan tahap penentuan toleransi pada aspek parameter kerugian pada sisi pengguna produk akibat penyimpangan produk dari target parameter yang semestinya. Pada tahap toleransi desain sebuah kualitas ditingkatkan dengan mengetatkan toleransi pada target parameter produk yang semestinya dengan capaian proses yang telah dihasilkan untuk kebutuhan pengguna produk *(customer)*.



Gambar 68. Pengembangan Produk Menggunakan *Robust Engineering*
(sumber: *"Taguchi's Quality Engineering Handbook"* - *Product development with robust engineering*)

Pembuatan sebuah produk perangkat lunak yang baik tentu tidak lepas dari mengacu pada parameter kualitas dan kekuatan desain *(robust design)*. Pertanyaannya adalah, seperti apa gambaran secara spesifik sebuah produk perangkat lunak yang dapat dikatakan memiliki nilai kualitas dan nilai *robust design*. Pada buku berjudul *'Taguchi's Quality Engineering Handbook'*, Taguchi mengatakan bahwa;

> *As a generic term, quality or robust design has no meaning; it is merely an objective. A product that functions under any conditions is obviously good. Again, saying this is meaningless. All engineers attempt to design what will work under various conditions. The key issue is not design itself but how to evaluate functions under known and unknown conditions.* (Taguchi, 2005:57).

Sebuah produk yang memiliki nilai kualitas dan nilai kekuatan desain *(robust design)* adalah produk yang dapat berfungsi dalam berbagai kondisi. Oleh karena itu seorang *engineer* harus memiliki kemampuan untuk merancang

produk yang mampu berfungsi dalam berbagai kondisi para penggunanya, baik dalam kondisi yang dapat diketahui maupun dalam kondisi yang tak terduga.

C. Mekanisme *Software Testing* dengan *Software Quality*

Proses menjaga kualitas secara utuh dari rekayasa perangkat lunak *(software engineering)* adalah tahapan pengujian perangkat lunak yang bersangkutan *(software testing)* yang mengacu pada kualitas perangkat lunak yang diharapkan *(software quality)*. Serangkain proses tersebut dilakukan sebagai upaya untuk memberikan jaminan kualitas *(software quality assurance)* pada produk perangkat lunak yang dibuat. Acuan parameter kualitas perangkat lunak dalam penelitian ini adalah ISO 25010. Standar kualitas ISO 25010 terdiri dari berbagai aspek diantaranya adalah aspek *functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability,* dan *portability*. Tahapan pengujian perangkat lunak *(software testing)* terdiri dari berbagai strategi pengujian, diantaranya adalah *unit testing, integration testing, system testing, acceptance testing.* Teknik pengujian yang dilakukan antara lain teknik *black-box testing (external)* dan *white-box testing (internal).*

Pendekatan aspek kualitas bertujuan tidak lain hanyalah untuk menghasilkan perangkat lunak yang berkualitas tinggi. Kualitas mengacu pada karakteristik yang dapat diukur, sesuatu yang dapat dibandingkan dengan standar yang sudah diketahui. Tetapi perangkat lunak, yang sebagian besar merupakan entitas intelektual, lebih menantang untuk dikarakterisasi daripada objek fisik

(Pressman, 2002:218). Ada dua jenis kualitas yang ada, yaitu kualitas desain dan kualitas konformansi.

Kualitas desain mengacu pada karakteristik yang ditentukan oleh desainer terhadap suatu item tertentu. Nilai material, toleransi, dan spesifikasi kinerja, semua memberikan kontribusi terhadap kualitas desain. *Kualitas konformansi* adalah tingkat dimana spesifikasi desain terus disempurnakan selama proses pembuatan. Definisi kualitas perangkat lunak menurut Pressman menjelaskan bahwa konformansi terhadap kebutuhan fungsional dan kinerja yang dinyatakan secara eksplisit, standar perkembangan yang didokumentasikan secara eksplisit, ke karakter implisit pada semua perangkat lunak yang dikembangkan secara profesional.

Definisi kualitas perangkat lunak didasarkan pada tiga hal berikut : (1) kebutuhan perangkat lunak merupakan fondasi sebagai acuan kualitas. Kurang sesuainya dengan kebutuhan menunjukkan rendahnya kualitas. (2) Standar yang telah ditentukan mencerminkan proses apa saja yang harus dilakukan dalam merekayasa perangkat lunak. Jika kriteria pada standar acuan tidak diikuti, dapat dipastikan akan menimbulkan kualitas perangkat lunak yang kurang baik. (3) Adanya sebuah kebutuhan implisit, misalnya kebutuhan akan pemeliharaan yang baik. Bila perangkat lunak mampu menyesuaikan kebutuhan eksplisitnya namun gagal memenuhi kebutuhan implisitnya, maka kualitas perangkat lunak tersebut perlu diragukan.

Parameter kualitas perangkat lunak *(software quality)* yang digunakan mengacu pada standar kualitas ISO 25010, dengan aspek faktor yang digunakan

diantaranya adalah *functional suitability, maintainability, portability,* dan *usability.* Penentuan aspek faktor yang digunakan berdasarkan pada kesesuaian karakteristik perangkat lunak yang dibuat. Aspek standar kualitas yang tidak digunakan tersebut diantaranya adalah *performance efficiency, compatibility, reliability, security.* Aspek *performance efficiency* tidak digunakan karena karakteristik perangkat lunak yang dibuat tidak beroperasi pada jaringan internet, dan aktivitas operasional tidak memerlukan akses ke *database server* (Jamo Solutions, 2013:5). Aspek *reliability* tidak digunakan karena perangkat lunak yang dibuat tidak beroperasi pada jaringan internet dan tidak berbasis *web (web-enabled applications)* (Watkins, 2011:26).

Aspek *security* tidak digunakan karena dalam operasional perangkat yang dibuat tidak berupa penyimpanan data *(data accessible)* maupun berbasis *account* (Watkins, 2011:324). Aspek *reliability* dan *security* berhubungan erat dengan layanan berbasis jaringan internet *(service provider)* (Sommerville, 2011:525). Hubungan diantara mekanisme pengujian perangkat lunak *(software testing)* dan standar acuan kualitas perangkat lunak *(software quality)* ISO 25010 tersebut secara detail diuraikan sebagai berikut.

1. Tahapan Pengujian

Upaya untuk menghasilkan sebuah produk perangkat lunak yang baik, tentu membutuhkan berbagai serangkaian pengujian. *Software testing* berbeda dengan *debugging*. Proses *debugging* mengakomodasi berbagai teknik *software testing* (Agarwal, 2010:161). Strategi *software testing* mengakomodasi dua jenis pengujian perangkat lunak, yaitu pengujian lingkup terkecil dari bagian perangkat lunak *(low-level test)* dan pengujian perangkat lunak dalam lingkup yang komperhensif *(high-level test)*.

a. Unit Testing

Pengujian unit merupakan tahapan pengujian pertama yang dilakukan *engineer* (pembuat) pada bagian terkecil dari perangkat lunak yang dibuat. Tahap *unit testing* menggunakan teknik *white-box testing* (Watkins, 2011:52). Ranah yang diuji dalam *unit testing* adalah *Basis-path testing* yang terdiri dari penguraian *Flowgraph, Cyclomatic Complexity (CC), Independent Path* hingga merancang *Tes case*. Penghitungan *Cyclomatic Complexity (CC)* menggunakan V(G) = $e - n + 2$, dengan elemen berupa *edge* dan *node* (Najadat, 2012:2).

b. Integration Testing

Tahap pengujian integrasi merupakan pengujian bagian terkecil dari perangkat lunak yang telah disatukan. Pengujian *integration* dilakukan oleh *engineer* bersangkutan dengan menggunakan teknik *black-box testing*. Tahap pengujian *integration testing* dilakukan dengan

pemberian kasus uji *(test case)* yang telah dibuat pada akhir pengujian *unit testing*. Pengujian *integration testing* dilakukan untuk mengukur tingkat *functional suitability* pada parameter kualitas perangkat lunak yang dipakai, yaitu ISO 25010 (Watkins, 2011:60).

c. System Testing

Proses pengujian selanjutnya adalah *system testing*, yaitu pengujian yang dilakukan *engineer* yang bersangkutan dengan menggunakan teknik *black-box testing* ketika semua komponen perangkat lunak telah menjadi satu kesatuan sistem yang utuh. Pengujian *system testing* dilakukan untuk mengukur tingkat *maintainability* dan *portability* perangkat lunak yang dibuat. Aspek *maintainability* melalui pengukuran *Duplication Source Code, Line of Code (LoC), Cyclomatic Complexity (CC)* sedangkan pada aspek *portability* menggunakan aktivitas instalasi pada berbagai jenis konfigurasi perangkat keras. Penghitungan *Line of Code (LoC)* merupakan penghitungan *size* dari *software* yang dibuat. *Size* dari *software* yang dibuat merupakan faktor untuk penilaian diagnostik dan keperluan analisis pada perangkat lunak dengan menggunakan metrik *Line of Code (LoC)* (Koyya, 2013:3).

Proses penghitungan *Duplication Source Code* sebagai cara untuk mengetahui ketidakefektifan dalam merancang sebuah *syntax source code*, sehingga berdampak pada jumlah *Line of Code (LoC) source code* yang semakin besar. Hasil penghitungan *duplication code* selanjutnya dikonversikan dengan tabel *rating* untuk mengetahui kualitas *source*

code yang telah dibuat dari aspek *duplication*. Berikut tabel klasifikasi penilaian untuk *duplication* (Heitlager, 2007:7).

Tabel 12. Klasifikasi Penilaian *Duplication Source Code*

rank		duplication
++	very small	0-3%
+	small	3-5%
o	moderate	5-10%
-	big	10-20%
--	extremely big	20-100%

Penghitungan duplikasi memberikan kemudahan untuk menganalisis akar penyebab duplikasi berada diposisi mana, serta dapat mendaftar dan melacak bagian mana saja yang lebih banyak terjadi duplikasi. Penyelesaian masalah duplikasi sering melibatkan lebih dari sekedar penguraian atau menetralisir bagian terkecil dari *source code* kedalam format yang dapat digunakan kembali (Heitlager, 2007:7).

d. Acceptance Testing

Tahapan pengujian berikutnya adalah *acceptance testing*, yaitu pengujian perangkat lunak yang dilakukan oleh pihak ahli, baik dari pihak ahli media maupun pihak ahli materi. Proses *acceptance testing* dilakukan dengan menggunakan teknik *black-box testing*, yang juga dilakukan oleh pihak pengguna *(user)* dalam hal ini adalah siswa Teknik Mekatronika. Proses pengujian *acceptance testing* dilakukan untuk mengukur aspek *usability* perangkat lunak (Watkins, 2011:82). Aspek *usability* diukur menggunakan angket yang telah dibuat dengan menggunakan *USE Questionnaire* yang sudah terstandar.

2. Metode Pengujian

Proses pengujian perangkat lunak memiliki strategi dan proses pengujian secara terstruktur. Strategi pengujian perangkat lunak memberikan gambaran peta jalan bagi pengembang produk perangkat lunak untuk menghasilkan sebuah kualitas produk yang baik. Strategi pengujian perangkat lunak juga cukup fleksibel untuk mewadahi kreativitas dan kustomisasi yang diperlukan pengembang dalam menyempurnakan produk perangkat lunak yang dibuat. Ada beberapa teknik pengujian perangkat lunak yang lazim dilakukan sebagai pengembang produk perangkat lunak. Teknik-teknik pengujian perangkat lunak tersebut diantaranya adalah sebagai berikut;

a. White-Box Testing

White-Box testing merupakan teknik pengujian perangkat lunak pada aspek internal sistem dari perangkat lunak tersebut. Segmen dari aspek internal sistem diantaranya adalah, algoritma program dan struktur *source-code*. Istilah lain dari teknik *white-box testing* ini biasa disebut juga dengan teknik *glass testing* atau *open-box testing.*

b. Black-Box Testing

Black-Box testing merupakan teknik pengujian perangkat lunak tanpa harus mengetahui proses kerja internal dari perangkat lunak tersebut. Pengembang produk perangkat lunak tidak menguji hingga ke sistem maupun *source-code*. Pengembang produk hanya menguji kinerja

output yang dihasilkan ketika serangkaian *input* telah diberikan. Pihak penguji lebih dominan mengamati dan menilai aspek *user interface (UI)*.

c. Alpha Testing

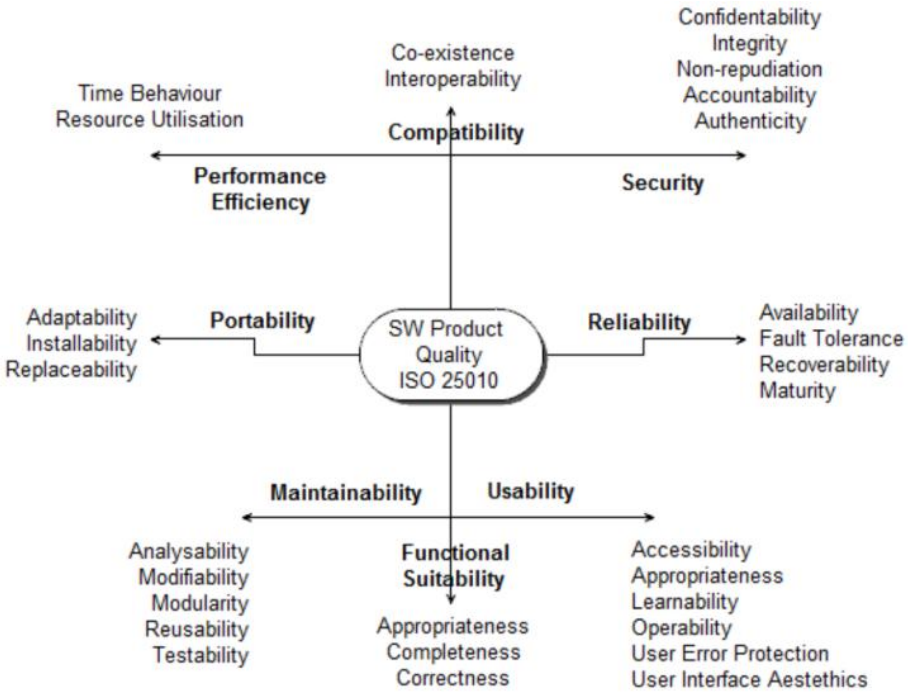
Pengujian alpha pada rekayasa perangkat lunak adalah serangkaian pengujian sistem yang dilakukan pada sisi pihak pengembang (Agarwal, 2010:172). Pengujian yang termasuk dalam zona *alpha testing* adalah *system testing*, *acceptance testing* karena produk telah sampai pada kondisi aktif dan *usable* secara operasional.

d. Beta Testing

Pengujian beta adalah pengujian yang dilakukan pada pengguna sasaran dibuatnya produk tersebut. Pengujian beta merupakan aspek pengujian dalam mengukur tingkat *usability* perangkat lunak (Chemuturi, 2011:183). Pengujian beta merupakan representasi dari aspek *usability* pada ISO/25010. Pengujian beta dilakukan dengan terlebih dahulu proses pengujian alpha telah dilakukan. Pengujian beta dilakukan dengan tujuan utama ingin mendapatkan masukan dan *feedback* bahan evaluasi terhadap produk yang telah dibuat. Hal ini tentunya menyangkut banyak hal, seperti kemudahan proses instalasi, kemudahan operasional, dan faktor kenyamanan lain dari sisi pengguna.

3. ISO/IEC 25010

ISO/IEC 25010 merupakan standar pengujian perangkat lunak yang dikembangkan oleh *International Organization for Standardization-International Electrotechnical Commision*. ISO/IEC 25010 merupakan pengembangan dari standar sebelumnya yaitu ISO/IEC 9126, yang merupakan turunan dari model Boem & McCall (Wagner, 2013:60). ISO/IEC 25010 merupakan bagian dari ruang lingkup *Software Quality Assurance (SQA)* atau jaminan kualitas perangkat lunak (Schulmeyer, 2008:63). Aspek pengujian kualitas pada ISO/IEC 25010 terdiri dari delapan standar pokok, diantaranya adalah: *functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, portability*. Penjelasan lebih detail bisa dipahami dari gambar berikut;



Gambar 69. ISO/IEC 25010
(sumber: Jurnal *"Test software quality issues and connections to international standards"*)

a. Functional Suitability

Pengertian dari standar kualitas ini adalah sebuah *software* harus memiliki fungsional yang sesuai dengan kebutuhan *user*. Faktor komposisi dari *functional suitability* diantaranya adalah kelayakan produk terhadap sasaran pengguna *(appropriateness)*, kelengkapan fitur produk dengan kebutuhan pengguna *(completeness)*, ketepatan konten produk dengan segmen penggunanya *(correctness)*.

b. Performance Efficiency

Kualitas sebuah perangkat lunak untuk dapat merespon permintaan pengguna dan bagaimana tingkat efisiensi ketika perangkat lunak melakukan eksekusi. Faktor *performance efficiency* erat sekali hubungannya dengan waktu. Subfaktor dari *performance efficiency* diantaranya adalah kemampuan operasional perangkat lunak dalam menempuh waktu *(time behavior)*, kemampuan produk perangkat lunak sebagai sumber yang dapat digunakan *(resource utilization)*.

c. Compatibility

Kemampuan *software* untuk beroperasi dan menjalankan berbagai fungsinya dalam berbagai jenis lingkungan *software* dan *hardware* (Kovács, 2013:98). Faktor *compatibility* berhubungan erat dengan kemampuan produk perangkat lunak dalam beroperasi pada berbagai lingkungan sistem operasi *(cross platform/interplatform)*. Faktor pendukung *compatibility* antara lain kemampuan produk perangkat

lunak untuk dapat aktif dalam suatu sistem operasi *(co-existence)*, dan kemampuan produk perangkat lunak untuk dapat beroperasi dalam berbagai lingkungan sistem operasi *(interoperability)*.

d. Usability

Karakteristik perangkat lunak mudah dioperasikan dan mudah dipahami berbagi fiturnya oleh pengguna. Hal ini mengacu pada tingkat kualitas *user interface* untuk digunakan dapat memberikan kemudahan pada setiap penggunanya. Faktor yang terdapat dalam *usability* diantaranya adalah kemampuan produk perangkat lunak memiliki kemudahan dalam proses diakses *(accessibility)*, kemampuan produk perangkat untuk mudah dipelajari *(learnability)*, kemampuan produk perangkat untuk mudah dioperasikan *(operability)*, kemampuan produk perangkat untuk terlindung dari kesalahan pengguna dalam mengoperasikan *(user error protection)* dan kemampuan produk perangkat untuk memiliki nilai yang tinggi dalam hal artistik tampilannya *(user interface aesthetics)*.

e. Reliability

Tingkat kemampuan perangkat lunak untuk dapat dioperasikan dalam berbagai fungsi, waktu, dan kondisi tanpa mengalami kesalahan atau bahkan kegagalan sistem. Tingkat *reliability* dalam perangkat lunak sering dipahami sebagai sisi kehandalan produk perangkat lunak dalam berbagai kondisi penggunaan.

f. Security

Karakteristik perangkat lunak untuk memiliki sistem proteksi tertentu, baik terhadap data maupun informasi yang ada didalamnya. Produk dan sistem bersangkutan memiliki akses dan kewenangan tertentu untuk dioperasikan.

g. Maintainability

Karakteristik sebuah produk perangkat lunak untuk dapat diubahsuaikan/modifikasi ketika dibutuhkan perbaikan dan pengembangan ke komposisi sistem yang baru. Berbagai aspek yang ada dalam *maintainability* diantaranya adalah *modifiability, modularity, reusability, analyzability.*

h. Portability

Tingkat kemampuan produk perangkat lunak untuk dapat dioperasikan dalam berbagai perangkat keras tertentu, hal ini tentu menyangkut spesifikasi prosesor, resolusi layar, dan sejenisnya. Faktor pendukung dalam *portability* diantaranya adalah kemampuan produk perangkat lunak untuk mampu beradaptasi dalam berbagai konfigurasi perangkat keras *(adaptability)*, kemampuan produk perangkat lunak untuk mudah dipasang pada perangkat *(installability)*, dan kemampuan produk perangkat lunak untuk dapat dihapus dan di pasang kembali *(replaceability)*.

D. Kajian Penelitian yang Relevan

Sunawi (2010). Berjudul "*Pengaruh Penggunaan Media Pembelajaran Panel Hidrolik Terhadap Hasil Belajar Mata Diklat Hidrolik Siswa Kelas XI Program Keahlian Teknik Alat Berat SMK Negeri 1 Singosari*". Tesis, Program Studi Pendidikan Kejuruan, Program Pascasarjana Universitas Negeri Malang. Pendekatan yang digunakan adalah pendekatan kuantitatif. Instrumen pengumpul data dengan tes tertulis dalam bentuktes objektif. Pengumpulan data menggunakan metode eksperimen. Analisisdata menggunakan analisis ANOVA dua jalur. Hasil penelitian menunjukkan bahwa: (1) ada pengaruh yang signifikan hasil belajar sistem hidrolik, antara kelompok siswa yang menggunakan media pembelajaran panel terpisah dan media pembelajaran panel utuh; (2) ada pengaruh terhadap hasil belajar, antara kelompok siswa yang motivasi berprestasi tinggi dengankelompok siswa yang motivasi berprestasi rendah; (3) tidak ada interaksi antara penggunaan media pembelajaran panel dan motivasi berprestasi yang berpengaruh terhadap hasil belajar sistem hidrolik. Berdasarkan hal tersebut, guru perlu mengusai penggunaan media pembelajaran sehingga dalam proses pembelajaran, dapat memaksimalkan informasi yang disampaikan kepada siswa, SMK hendaknya lebih inovatif dan variatif dalampembuatan media pembelajaran, perlunya pemenuhan kebutuhan media pembelajaran yang sesuai dengan tujuan pembelajaran oleh pihak terkait.

Bambang Setiyo Hari Purwoko (2009). Berjudul "*Pengembangan Mesin CNC Virtual Sebagai Media Interaktif untuk Melayani Pembelajaran Pemrograman CNC*". Seminar Nasional Electrical, Informatics, and It's

Educations 2009, Jurusan Pendidikan Teknik Mesin, Fakultas Teknik - Universitas Negeri Yogyakarta. Metode yang digunakan dalam penelitian ini adalah merupakan penelitian pengembangan. Obyek penelitian adalah rekayasa pemrograman dengan bahasa Visual Basic 6 guna menghasilkan mesin CNC Virtual. Subyek penelitian ini adalah dosen CNC, guru CNC SMK, mahasiswa, siswa SMK, ahli teknologi pembelajaran, dan ahli media pendidikan. Pengumpulan data dilakukan dengan observasi, kusioner, dan tes hasil belajar. Alat pengumpulan data seperti lembar observasi, angket atau kuisioner, dan soal tes hasil belajar, dikembangkan oleh peneliti. Analisis data dilakukan dengan analisis deskriptif. Hasil penelitian menunjukkan: (1) prototype media interaktif mesin CNC virtual berhasil diwujudkan sesuai perencanaan. Tampilan dapat menghadirkan lingkungan fisik mesin CNC yang terdiri, kontrol panel, layar komputer, dan sistem penjepitan benda kerja, (2) prototype media dapat mensimulasikan fungsi komponen virtual seperti saklar utama, saklar pemutar spindel utama, dan tombol-tombol pada mode pengoperasian CNC dengan baik dan mampu menampilkan simulasi gerakan pahat meskipun sebatas pada kode program tertentu, dan (3) prototype mesin CNC virtual memenuhi uji kelayakan untuk digunakan sebagai media interaktif pembelajaran pemrograman CNC.

Hui-Chun Chu, Gwo-Jen Hwang, Shu-Xian Huang dan Ting-Ting Wu (2007). "*A knowledge engineering approach to developing e-libraries for mobile learning*". Department of Information and Learning Technology, National

University of Tainan, Tainan City, Taiwan. Desain penelitian adalah *research & development*.

Ju-Ling Shih, Gwo-Jen Hwang, Yu-Chung Chu dan Chien-Wen Chuang (2010). Berjudul "*An investigation-based learning model for using digital libraries tosupport mobile learning activities*". Department of Information and Learning Technology, National University of Tainan, Tainan City, Taiwan. Desain penelitian adalah *research & development*.

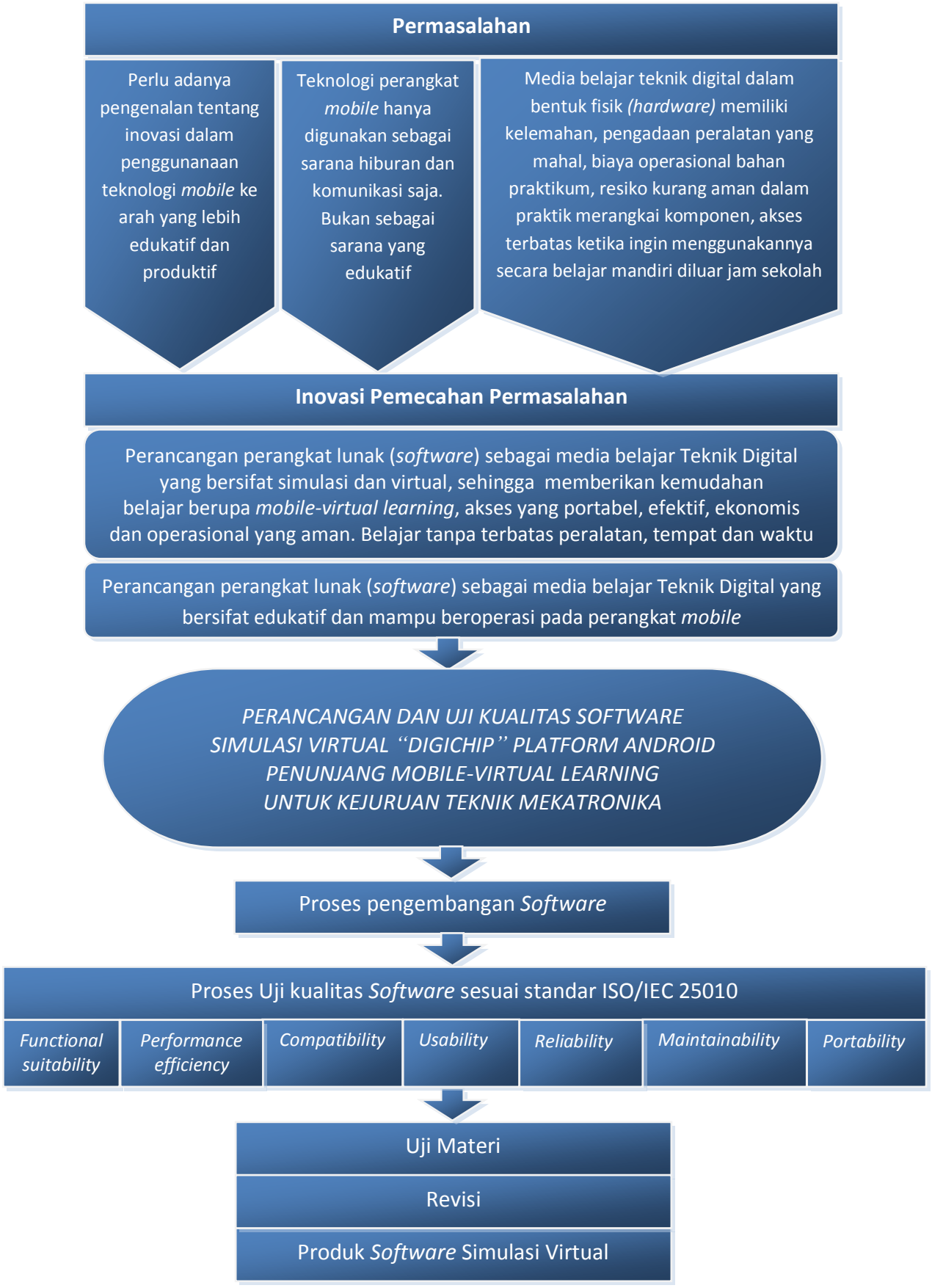
E. Kerangka Pikir

Penggunaan media belajar yang lebih praktis, menarik dan bisa digunakan kapan saja dan dimana saja akan memberikan kemudahan tersendiri dalam belajar serta akan memiliki pengaruh terhadap motivasi belajar dalam mewujudkan kualitas belajar yang baik. Siswa tidak terbatas pada ketersediaan alat praktikum, bahan praktikum, waktu (jam belajar dikelas/lab/bengkel) dan sistem belajar yang harus bergilir/berkelompok. Siswa dapat melakukan kegiatan belajar kapan saja dan dimana saja dengan menggunakan perangkat lunak simulasi yang berbasis *mobile learning*.

Kerangka pikir yang berisikan gambaran logis bagaimana variabel-variabel saling berhubungan (berkorelasi). Korelasi hubungan tersebut idealnya dikuatkan oleh teori atau penelitian sebelumnya. Dalam menyusun kerangka pikir, dimulai dari variabel yang mewakili masalah penelitian. Proses pengembangan media pembelajaran simulasi pada mata pelajaran teknik digital untuk *platform* Android ini hanya akan mengacu pada uji materi dan kriteria

software quality menurut ISO 25010 yakni *functionality*, *reliability*, *usability*, *efficiency*, *portability* dan *maintainability*. Pemilihan versi Android *Gingerbread*, *Honeycomb*, *Ice Cream*, dan *Jelly Bean* karena versi tersebut adalah sistem minimal untuk menjalankan *framework*. Alasan lain adalah pada penguasaan pangsa pasar yang dimiliki serta kemiripan fitur teknologinya.

Penelitian ini diawali dengan adanya permasalahan yang muncul sehingga diperlukan altenatif penyelesaian masalah. Adapun penyelesaian masalah adalah dengan membuat *software*/aplikasi media belajar simulasi pada mata pelajaran teknik digital untuk *platform* Android. Setelah aplikasi dibuat, dilakukan uji kualitas *software* dan uji materi terhadap aplikasi yang telah dibuat oleh peneliti dan validator ahli yang dirujuk. Setelah dilakukan uji dilanjutkan dengan revisi kemudian implementasi (pengujian ke siswa) sampai menghasilkan *software* dengan kualitas yang baik dan telah memenuhi aspek *functionality*, *reliability*, *efficiency*, *usability*, *portability* dan *maintainability* menurut standar ISO 25010. Adapun kerangka pikir dalam penelitian ini adalah sebagai berikut:



Gambar 70. Kerangka Pikir

F. Pertanyaan Penelitian

1. Bagaimana tahapan perancangan dan pengujian *software* simulasi "DigiChip" yang dibuat ?
2. Bagaimana kualitas perancangan *software* simulasi "DigiChip" yang dibuat, berdasarkan faktor kualitas *functional suitability* ?
3. Bagaimana kualitas perancangan *software* simulasi "DigiChip" yang dibuat, berdasarkan faktor kualitas *maintainability* ?
4. Bagaimana kualitas perancangan *software* simulasi "DigiChip" yang dibuat, berdasarkan faktor kualitas *portability* ?
5. Bagaimana kualitas perancangan *software* simulasi "DigiChip" yang dibuat, berdasarkan faktor kualitas *usability* ?

BAB III
METODE PENELITIAN

A. Jenis Penelitian

Penelitian ini menggunakan metode penelitian *Research and Development (R&D)*. Proses perancangan dan pengujian *software* menggunakan metode *Software Development Life Cycle (SDLC)* jenis *V-model*, yang terdiri dari (Pressman, 2010:40): (1) analisis kebutuhan *(Requirement modeling)*, (2) desain rancangan *(Architectural design)*, (3) desain komponen *(Component design)*, (4) implementasi kode program *(Code generation)*, (5) pengujian unit *(Unit testing)*, (6) pengujian integrasi *(Integration testing)*, (7) pengujian sistem *(System testing)*, (8) pengujian penerimaan *(Acceptance testing)*.

Pengujian kualitas perangkat lunak yang mengacu ISO 25010 pada aspek *functional suitability* diuji menggunakan kuesioner daftar *run test* dan kasus uji *(test case)*. Aspek *maintainability* diukur melalui pengukuran *Duplication Source Code, Line of Code (LoC), Cyclomatic Complexity (CC), Maintainability* Index *(MI)*, sedangkan pada aspek *portability* menggunakan aktivitas instalasi pada berbagai jenis konfigurasi perangkat keras dan berbagai versi kernel, mulai dari *Ginggerbread, Honeycomb, Ice Cream Sandwich, JellyBean, KitKat, Lolipop, Marshmallow, Nougat* hingga *Oreo*. Aspek *usability* diuji menggunakan *USE Questionnaire* pada *user* dan penghitungan *Cronbach's Alpha*.

B. Proses Perancangan dan Pengujian

1. Requirement Modeling

Analisis kebutuhan dalam penelitian ini didasarkan pada berbagai potensi kebutuhan pengguna *(user)* melalui tahap investigasi dilingkungan sekolah. Tahap *requirement modeling* dilakukan dengan observasi pengamatan pada aktivitas belajar teori maupun praktikum pelajaran teknik digital. Analisis kebutuhan dan observasi berlanjut pada pengamatan berbagai peralatan belajar dan praktikum pelajaran teknik digital. Proses analisis kebutuhan dilakukan pula dengan metode wawancara pada peserta didik, pendidike, pengelola laboratorium, yang berhubungan dengan pembelajaran teknik digital yang ingin diteliti.

2. Architectural Design

Tahap *architectural design* merupakan proses analisis dan penggalian informasi terhadap konsep desain media yang dibutuhkan dalam pelajaran teknik digital yang diteliti. Tahap selanjutnya melakukan analisis perangkat lunak seperti apa yang bisa memecahkan masalah yang dihadapi dalam pelajaran teknik digital yang diteliti, serta piranti apa yang bisa menjalankan perangkat lunak yang dikembangkan. Pengumpulan informasi dilakukan dengan aktivitas observasi, wawancara.

3. Component Design

Setelah kebutuhan sistem diketahui, maka selanjutnya dilakukan desain komponen dan sistem yang merupakan gambaran dari analisis

kebutuhan, meliputi desain: *Unified Modelling Language (UML)*, diagram alir *(flowchart)* dan tampilan *(user interface)*. Proses desain sistem pada tahap ini meliputi 2 sisi, yaitu sisi *software* simulasi dan sisi antarmuka pengguna yang akan dioperasikan melalui tablet PC atau *smartphone*.

4. Code Generation

Implementasi kode program dilakukan sesuai dengan desain sistem yang telah dibuat supaya hasilnya dapat sesuai dengan tujuan pembuatan sistem. Rancangan program dan algoritma yang telah disiapkan kemudian diimplementasikan ke dalam bahasa pemrograman, sehingga semua fungsi dapat dijalankan sesuai analisis kebutuhan.

5. Unit Testing

Setelah perangkat lunak dibuat langkah selanjutnya adalah pengujian. Pengujian pada tahap *unit testing* atau bagian terkecil dari perangkat lunak adalah menguji ranah *basis-path*, yang terdiri dari menguraikan *Flowgraph, Cyclomatic Complexity (CC), Independent Path*.

a. Pembuatan *Flowgraph*

Flowgraph merupakan gambaran dari arus logika program perangkat lunak. Komponen pada *flowgraph* terdiri bentuk lingkaran *(node)* dan tanda panah *(edge)*. Bentuk lingkaran adalah simbol pernyataan program. Tanda panah merupakan simbol dari arah kerja program.

b. Penghitungan *Cyclomatic Complexity (CC)*

Simbol *cyclomatic complexity* pada rumus dinotasikan dengan V(G), McCabe (1976:308). Penghitungan *cyclomatic complexity* adalah dengan rumus sebagai berikut;

$$V(G) = e - n + p$$

V(G) = *Cyclomatic Complexity*
e = jumlah *edge* pada *flowgraph*
n = jumlah *node* pada *flowgraph*
p = komponen terkoneksi

Connected Components (p) memiliki nilai konstanta berjumlah "2" (Laplante, 2007:176).

c. Pembuatan *Independent Path*

Independent Path adalah jalur pada program yang menghubungkan *node* awal dengan *node* akhir. Jalur sebuah *independent path* adalah sebuah jalur yang melewati sebuah tanda panah *(edge)* baru, dan merupakan jalur yang belum pernah dilalui sebelumnya.

6. Integration Testing

Tahap pengujian selanjutnya adalah *integration testing,* yang merupakan pengujian *run test* pada bagian terkecil dari perangkat lunak yang telah disatukan. Pengujian *integration testing* dilakukan oleh *engineer* bersangkutan dengan menggunakan metode *checklist* pemberian kasus uji *(test case)* yang telah dibuat pada akhir pengujian *unit testing*. *Test case*

merupakan serangkaian panduan kasus uji yang digunakan untuk mengetahui kinerja setiap alur program. *Test case* berisi berbagai pernyataan tertulis mengenai arus logika program perangkat lunak yang telah dibuat. Pengujian *integration testing* dilakukan untuk mengukur tingkat *functional suitability* pada parameter kualitas perangkat lunak yang dipakai, yaitu ISO 25010. Pengujian aspek *functionality suitability*, berfokus pada kesesuaian satu kesatuan fungsi untuk dapat melakukan tugas-tugas tertentu sesuai dengan yang diharapkan.

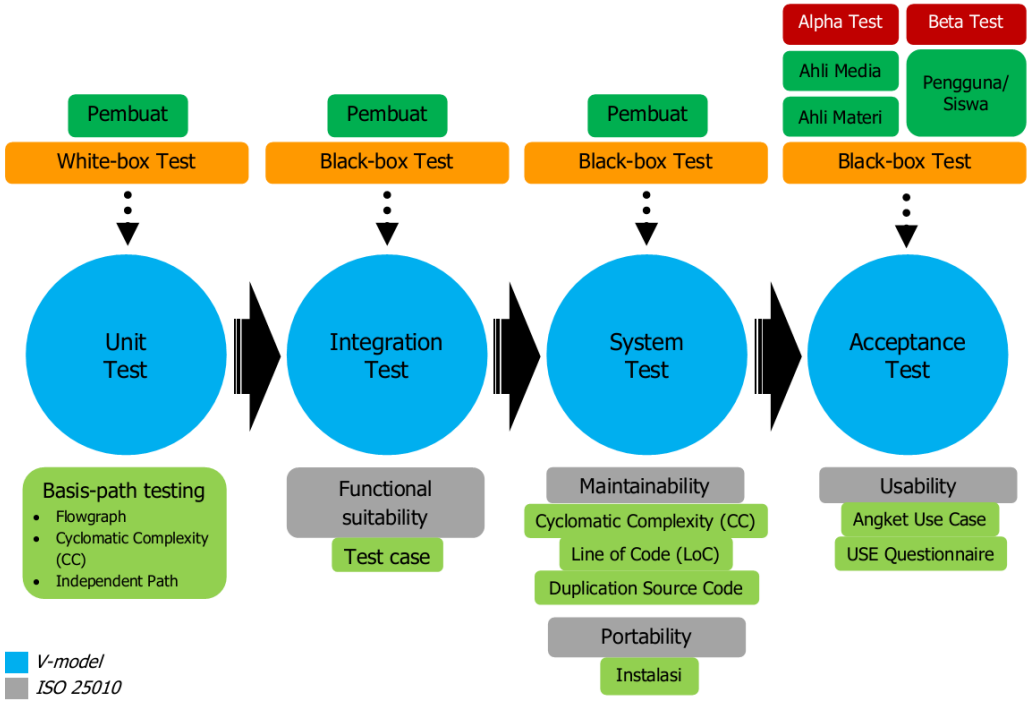
7. System Testing

Metode pada tahap pengujian *system testing*, yaitu pengujian yang dilakukan *engineer* yang bersangkutan ketika semua komponen perangkat lunak telah menjadi satu kesatuan sistem yang utuh. Pengujian *system testing* dilakukan untuk mengukur tingkat *maintainability* dan *portability* aspek ISO 25010 pada perangkat lunak yang dibuat. Aspek pengujian parameter *maintainability* melalui pengukuran *Code Duplication*, *Line of Code (LoC)*, *Cyclomatic Complexity (CC)* untuk menghitung nilai *Maintainability Index (MI)*. Penghitungan nilai MI dilakukan dengan metode analisis menggunakan perangkat lunak yang digunakan pada proses *code genertion* yang memiliki fitur *analyzer*. Aspek *portability* menggunakan aktivitas instalasi yang dilakukan pada berbagai jenis konfigurasi perangkat keras dan berbagai versi kernel OS dari android.

8. Acceptance Testing

Proses pengujian berikutnya adalah *acceptance testing*, yaitu pengujian perangkat lunak yang dilakukan oleh pihak ahli, baik dari pihak ahli media maupun pihak ahli materi dan juga dilakukan oleh pihak pengguna *(user)* dalam hal ini adalah responden siswa Teknik Mekatronika. Proses pengujian *acceptance testing* dilakukan untuk mengukur aspek *usability* perangkat lunak yang telah dibuat, yang mengacu pada standar ISO 25010. Uji *usability* menggunakan *USE Questionnaire* J.R. Lewis yang sudah terstandar yang akan diisi oleh pengguna *(user)*.

Tahapan pengujian *software* "DigiChip" dalam penelitian ini yang mengacu pada metode pengembangan perangkat lunak jenis *v-model*, proses alur pengujian yang dilakukan adalah sebagai berikut;



Gambar 71. Grafik Alur Pengujian Perangkat Lunak "DigiChip"

Aspek standar kualitas ISO 25010 yang digunakan dalam penelitian ini terdiri dari 4 standar aspek kualitas, diantaranya adalah *functional suitability*, *maintainability*, *portability*, dan *usability*. Aspek standar kualitas yang lain tidak digunakan karena menyesuaikan dengan karakteristik perangkat lunak yang dibuat. Aspek standar kualitas yang tidak digunakan tersebut diantaranya adalah *performance efficiency, compatibility, reliability, security*. Aspek *performance efficiency* tidak digunakan karena karakteristik perangkat lunak yang dibuat tidak beroperasi pada jaringan internet, dan aktivitas operasional tidak memerlukan akses ke database server. Aspek *reliability* tidak digunakan karena perangkat lunak yang dibuat tidak beroperasi pada jaringan internet dan tidak berbasis *web (web-enabled applications)*. Aspek *security* tidak digunakan karena dalam operasional perangkat yang dibuat tidak berupa aktivitas penyimpanan data *(data accessible)* maupun berbasis *account*. Aspek *reliability* dan *security* berhubungan erat dengan layanan berbasis jaringan internet *(service provider)*.

C. Subyek Penelitian

Sebagai tempat observasi dan pengambilan data dilakukan di SMK-SMTI Yogyakarta yang beralamat di Jl. Kusumanegara No.3 Kota Yogyakarta 55166, Propinsi D.I.Yogyakarta. Subjek untuk penelitian aspek *functional suitability*, *maintainability*, *portability* dan materi adalah media pembelajaran Simulasi Teknik Digital yang telah dibuat. Sedangkan subjek penelitian untuk aspek *usability* adalah siswa kelas XI Jurusan Teknik Mekatronika sejumlah 59 siswa.

D. Teknik Pengumpulan Data

Teknik pengumpulan data yang digunakan adalah sebagai berikut:

1. Observasi

Teknik observasi dilakukan untuk menggali informasi dan mengumpulkan data yang terkait dengan analisis potensi kebutuhan di lapangan.

2. Wawancara

Teknik pengumpulan data wawancara dilakukan pada tahap analisis kebutuhan. Teknik pengumpulan data dengan wawancara memiliki keuntungan dalam menggali kebutuhan *user* secara lebih leluasa mendapatkan informasi. Wawancara lebih disukai, dengan alasan komunikasi dilakukan lebih terbuka, dapat menjalin kepercayaan antara pengguna dan pengembang, serta memberi kesempatan untuk mengungkapkan pandangan yang berbeda secara leluasa. Wawancara dilakukan kepada siswa kelas XI, guru mata pelajaran teknik digital, pengelola laboratorium serta pengelola bidang sarana dan prasarana.

3. Studi literatur

Studi ini dikerjakan untuk mengumpulkan temuan riset dan informasi lain yang bersangkutan dengan pengembangan produk yang direncanakan.

E. Instrumen Penelitian

Pengujian validitas instrumen adalah suatu ukuran yang menunjukkan tingkat-tingkat kevalidan atau kesahihan suatu instrumen (Arikunto, 2013:211). Valid berarti instrumen tersebut dapat digunakan untuk mengukur apa yang hendak diukur. Instrumen yang valid berarti alat ukur yang digunakan untuk mendapatkan data itu valid. Dalam uji validitas, validitas konstruksi *(construct validity)* digunakan melalui konsultasi pendapat dari ahli *(judgment expert)*.

1. Instrumen Pengujian *Functional Suitability*

Pengujian aspek *functional suitability* dilakukan menggunakan daftar *test case*. Daftar ini berisi seluruh fungsi fitur yang ada pada perangkat lunak "DigiChip". Berikut adalah *test case* yang mengacu pada format Williams (2006:44). Hasil dari pembuatan *test case* dapat dilihat pada **(Lampiran 1).**

Tabel 13. *Test case Functional Suitability*

Identitas	Keterangan	Hasil tes

2. Instrumen Pengujian *Maintainability*

Pengujian aspek *maintainability* dengan cara menghitung besaran nilai *Maintainability Index (MI)* yang terdiri dari 3 komponen diantaranya adalah penghitungan *Line of Code (LoC), Cyclomatic Complexity (CC)* dan *Duplication Source Code*. Pada tahap pengukuran *Line of Code (LoC)* dilakukan dengan cara menghitung *source code* secara manual. Penghitungan

Cyclomatic Complexity menggunakan rumus V(G). Proses penghitungan *Duplication Source Code* menggunakan perangkat lunak Gendarme 2.11.

3. Instrumen Pengujian *Portability*

Pengujian aspek *portability* terdiri dari proses instalasi, mengoperasikan, dan *uninstall* aplikasi pada berbagai versi kernel android dan berbagi konfigurasi *hardware*. Instrumen yang digunakan yaitu berupa *checklist* untuk mencatat hasil uji. Berikut adalah instrumen uji *portability*;

Tabel 14. Pengujian *Portability*

No	Versi OS	Tipe	*Install*	*Running*

4. Instrumen Pengujian *Usability*

Pengujian aspek *usability* pada penelitian ini menggunakan alat ukur berupa kuesioner yang diberikan pada para siswa sebagai *user* perangkat lunak yang telah dibuat. Pengujian aspek *usability* untuk mengetahui kualitas perangkat lunak dari sudut pandang pengguna. Kuesioner yang digunakan untuk menguji aspek *usability* dalam penelitian ini menggunakan kuesioner *USE Questionnaire* (Lund, A.M, 2001:4). Angket *USE Questionnaire* relevan dengan berbagai aspek yang ada dalam standar pengujian kualitas ISO 25010. Angket *USE Questionnaire* sangat direkomendasikan karena berisi berbagai aspek diantaranya adalah *Usefulness*, *Satisfaction*, *Ease of Use*, dan *Ease of Learning* serta memiliki 7 skala likert untuk levelnya (Tullis, 2013:142). Berikut adalah gambaran dari *USE Questionnaire*;

Tabel 15. *USE Questionnaire (Arnold Lund – General Electric)*

Usefulness
01. It helps me be more effective.
02. It helps me be more productive.
03. It is useful.
04. It gives me more control over the activities in my life.
05. It makes the things I want to accomplish easier to get done.
06. It saves me time when I use it.
07. It meets my needs.
08. It does everything I would expect it to do.
Ease of Use
09. It is easy to use.
10. It is simple to use.
11. It is user friendly.
12. It requires the fewest steps possible to accomplish what I want to do with it.
13. It is flexible.
14. Using it is effortless.
15. I can use it without written instructions.
16. I don't notice any inconsistencies as I use it.
17. Both occasional and regular users would like it.
18. I can recover from mistakes quickly and easily.
19. I can use it successfully every time.
Ease of Learning
20. I learned to use it quickly.
21. I easily remember how to use it.
22. It is easy to learn to use it.
23. I quickly became skillful with it.
Satisfaction
24. I am satisfied with it.
25. I would recommend it to a friend.
26. It is fun to use.
27. It works the way I want it to work.
28. It is wonderful.
29. I feel I need to have it.
30. It is pleasant to use.

5. Instrumen *Alpha Testing* untuk Ahli Media

Pengujian pada tahap ini bertujuan untuk mengetahui kualitas media perangkat lunak dari sudut pandang ahli media. Sesuai dengan kajian teori pada bab sebelumnya, yang menyatakan bahwa karakteristik media yang baik dalam *mobile learning* adalah memiliki karakteristik yang sesuai dengan perangkat yang bersangkutan *(mobile tools)*, yaitu perangkat *mobile (mobile tools)* yang memiliki karakteristik berupa ukurannya yang kecil dan resolusi layar *(screen)* yang kecil pula, tentu memerlukan perangkat lunak yang mampu menyesuaikan (Pachler, 2010:68).

Karakteristik perangkat *mobile* yang demikian tentu mengarah pada spesifikasi media perangkat lunak yang harus dibuat berorientasi pada penyesuaian desain *user interface* dan tampilan visualnya, yaitu berupa gambar, warna, tombol, animasi, huruf, yang sesuai dengan karakteristik perangkat *mobile* yang mayoritas berukuran kecil tersebut. Pernyataan Pachler tentang aspek ukur dari kesesuaian media *mobile learning* dapat disimpulkan antara lain indikator kesesuaian desain visualnya, animasi, gambar, simbol, huruf, warna, dan lebih lengkapnya diuraikan dalam bentuk instrumen **(Tabel 16)**. Setelah kisi-kisi instrumen penilaian ahli media telah disusun, maka langsung selanjutnya adalah menyusun kisi-kisi dalam bentuk uraian pernyataan **(Tabel 17)**.

Tabel 16. Kisi-kisi Instrumen Penilaian Ahli Media

Indikator	Keterangan	Nomor Item	Jumlah Item
Warna	Kesesuaian penggunaan warna teks	1	1
	Kesesuaian penggunaan warna angka	2	1
	Kesesuaian penggunaan warna *background*	3	1
	Kesesuaian antara *foreground* dan *background*	4	1
Gambar	Tingkat kejelasan gambar	5	1
	Kesesuaian ukuran gambar	6	1
	Kesesuaian penempatan posisi gambar	7	1
Huruf	Tingkat kejelasan huruf	8	1
	Kesesuaian ukuran huruf	9	1
Angka	Tingkat kejelasan angka	10	1
	Kesesuaian ukuran angka	11	1
Animasi	Kesesuaian ukuran animasi gambar	12	1
	Kesesuaian ukuran animasi tombol	13	1
	Tingkat artistik efek animasi	14	1
Total Jumlah Soal			14

Tabel 17. Instrumen Penilaian Ahli Media

No	Pernyataan	Jawaban				
		SS	S	N	TS	STS
1	Penggunaan warna teks terlihat sudah proporsional					
2	Penggunaan warna angka terlihat sudah proporsional					
3	Penggunaan warna *background* sudah proporsional					
4	Kekontrasan antara *foreground* dan *background*					
5	Gambar dapat terlihat dengan jelas					
6	Ukuran gambar sudah proporsional					
7	Penempatan posisi gambar sudah proporsional					
8	Huruf dapat terlihat jelas					
9	Ukuran huruf sudah proporsional					
10	Angka dapat terlihat jelas					
11	Ukuran angka sudah proporsional					
12	Ukuran animasi gambar sudah proporsional					
13	Ukuran animasi tombol sudah proporsional					
14	Efek animasi terlihat artistik					

6. Instrumen *Alpha Testing* untuk Ahli Materi

Pengujian pada tahap ini bertujuan untuk mengetahui kesesuaian perangkat lunak yang telah dibuat dari sudut pandang ahli materi. Berikut diuraikan dalam bentuk instrumen;

Tabel 18. Kisi-kisi Instrumen Penilaian Ahli Materi

Indikator	Keterangan	Nomor Item	Jumlah Item
Simulasi Aljabar Boolean	Kesesuaian simulasi *input-output* (I/O) *logic gate* NOT	1	1
	Kesesuaian simulasi *input-output* (I/O) *logic gate* AND	2	1
	Kesesuaian simulasi *input-output* (I/O) *logic gate* NAND	3	1
	Kesesuaian simulasi *input-output* (I/O) *logic gate* OR	4	1
	Kesesuaian simulasi *input-output* (I/O) *logic gate* NOR	5	1
	Kesesuaian simulasi *input-output* (I/O) *logic gate* XOR	6	1
Simbolik *logic gate*	Kesesuaian simbol *logic gate* NOT	7	1
	Kesesuaian simbol *logic gate* AND	8	1
	Kesesuaian simbol *logic gate* NAND	9	1
	Kesesuaian simbol *logic gate* OR	10	1
	Kesesuaian simbol *logic gate* NOR	11	1
	Kesesuaian simbol *logic gate* XOR	12	1
	Kesesuaian rumus Aljabar Boolean	13	1
Skematik IC	Kesesuaian skematik IC NOT	14	1
	Kesesuaian skematik IC AND	15	1
	Kesesuaian skematik IC NAND	16	1
	Kesesuaian skematik IC OR	17	1
	Kesesuaian skematik IC NOR	18	1
	Kesesuaian skematik IC XOR	19	1
Total Jumlah Soal			19

Setelah kisi-kisi instrumen penilaian ahli materi telah disusun, maka langsung selanjutnya adalah menyusun kisi-kisi dalam bentuk uraian pernyataan. Berikut selengkapnya;

Tabel 19. Instrumen Penilaian Ahli Materi

No	Pernyataan	Jawaban	
		Ya	Tidak
1	Hasil simulasi *input* terhadap *output* pada *logic gate* NOT sudah benar		
2	Hasil simulasi *input* terhadap *output* pada *logic gate* AND sudah benar		
3	Hasil simulasi *input* terhadap *output* pada *logic gate* NAND sudah benar		
4	Hasil simulasi *input* terhadap *output* pada *logic gate* OR sudah benar		
5	Hasil simulasi *input* terhadap *output* pada *logic gate* NOR sudah benar		
6	Hasil simulasi *input* terhadap *output* pada *logic gate* XOR sudah benar		
7	Simbol *logic gate* NOT sudah benar		
8	Simbol *logic gate* AND sudah benar		
9	Simbol *logic gate* NAND sudah benar		
10	Simbol *logic gate* OR sudah benar		
11	Simbol *logic gate* NOR sudah benar		
12	Simbol *logic gate* XOR sudah benar		
13	Penulisan rumus Aljabar Boolean sudah benar		
14	Gambar skematik IC NOT sudah benar		
15	Gambar skematik IC AND sudah benar		
16	Gambar skematik IC NAND sudah benar		
17	Gambar skematik IC OR sudah benar		
18	Gambar skematik IC NOR sudah benar		
19	Gambar skematik IC XOR sudah benar		

7. Instrumen *Beta Testing*

Pengujian beta merupakan aspek pengujian dalam mengukur tingkat *usability* perangkat lunak (Chemuturi, 2011:183). Pengujian beta pada penelitian ini adalah menggunakan angket *USE Questionnaire* dengan skala penilaian menggunakan *likert scale*. Penjelasan sebelumnya telah memaparkan bahwa angket *USE Questionnaire* relevan dengan berbagai aspek yang ada dalam standar pengujian kualitas ISO 25010. Angket *USE Questionnaire* sangat direkomendasikan karena berisi berbagai aspek diantaranya adalah *Usefulness, Satisfaction, Ease of Use,* dan *Ease of Learning* serta memiliki 7 skala likert untuk levelnya (Tullis, 2013:142). Uji reliabilitas angket penelitian *USE Questionnaire* dilakukan dengan menggunakan metode analisis *Alpha Cronbach*. Kriteria suatu instrumen penelitian dikatakan reliabel dengan menggunakan teknik ini, bila angka indeks koefisien reliabilitas (r_{11}) > 0,6 (Arikunto, 2013:319).

F. Teknik Analisis Data

1. Analisis Data Aspek *Functional Suitability*

Teknik analisis data pada pengujian aspek *functional suitability* dengan memberikan *checklist* pada masing-masing *test case* yang mampu berfungsi dengan baik. Setelah data hasil pengujian diperoleh, langkah selanjutnya adalah menghitung presentase banyaknya fitur dalam perangkat lunak yang berfungsi dengan baik, dengan rumus sebagai berikut;

$$Persentase\ kelayakan\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

Hasil penghitungan yang diperoleh selanjutnya dikonversikan menjadi definisi kelayakan. Kategori kelayakan mengacu pada tabel kelayakan menurut Arikunto (2009:35).

Tabel 20. Klasifikasi Kelayakan

Angka (dalam %)	Klasifikasi
< 21	Sangat Tidak Layak
21 - 40	Tidak Layak
41 - 60	Cukup
61 - 80	Layak
81 - 100	Sangat Layak

2. Analisis Data Aspek *Maintainability*

Teknik analisis data pada aspek *maintainability* dengan cara menghitung nilai *Maintainability Index (MI)* yang dihasilkan dari 3 parameter nilai, yaitu; *Line of Code (LoC)*, *Cyclomatic Complexity* dan *Duplication Source Code*. Penghitungan nilai *Maintainability Index* menggunakan *software* yang digunakan untuk menyusun *script* kode bahasa C# pada proses pembuatan media pembelajaran "DigiChip", yang memiliki fitur *analyzer*. Nlai *Maintainability Index* yang telah diperoleh selanjutnya dikonversikan pada rumus persentase kelayakan. Setelah diperoleh persentase hasil akhir kemudian dikonversikan menjadi kriteria tabel kategori kelayakan.

Berikut merupakan tabel klasifikasi nilai hasil penghitungan *Maintainability Index* (Microsoft, 2007);

Tabel 21. Klasifikasi Nilai *Maintainability Index (MI)*

Maintainability Index (MI)	**Klasifikasi**	**Simbol**
0 - 9	Perawatan Sulit	
10 - 19	Perawatan Sedang	
20 - 100	Perawatan Mudah	

3. Analisis Data Aspek *Portability*

Pengujian aspek *portability* terdiri dari proses instalasi, mengoperasikan, dan *uninstall* aplikasi pada berbagai versi kernel android dan berbagi resolusi ukuran layar. Apabila dalam semua proses tersebut berjalan dengan baik tanpa mengalami kegagalan, maka dapat disimpulkan perangkat lunak telah memenuhi pengujian aspek *portability*.

4. Analisis Data Aspek *Usability*

Teknik penilaian untuk aspek *usability* menggunakan skala Likert, karena berhubungan dengan aktivitas mengukur sikap, pendapat dan persepsi seseorang atau kelompok tentang kejadian atau gejala sosial, dengan memberikan 5 pilihan skala jawaban (Riduwan, 2013:13).

Tabel 22. Skala *Likert*

Alternatif Jawaban	Nilai
(STS) Sangat Tidak Setuju	1
(TS) Tidak Setuju	2
(N) Netral	3
(S) Setuju	4
(SS) Sangat Setuju	5

Setelah hasil penghitungan jumlah akhir telah didapat, maka selanjutnya menghitung nilai presentase menggunakan rumus persentase kelayakan. Hasil persentase kemudian dikonversikan menjadi kriteria tabel kategori kelayakan.

5. Analisis Data Aspek Ahli Media & Ahli Materi

Teknik analisis data pada pengujian aspek ahli media dan ahli materi dengan menggunakan kriteria skala penilaian *Likert*. Yaitu dengan menyediakan 5 pilihan jawaban yang terdiri dari kriteria "Sangat Setuju (SS), Setuju (S), Netral (N), Tidak Setuju (TS), Sangat Tidak Setuju (STS)". Penghitungan hasil dengan cara menjumlahkan semua total nilai yang di dapat dari para ahli, selanjutnya menghitung nilai persentasenya menggunakan rumus persentase kelayakan. Langkah berikutnya mengkonversikan nominal persentase menjadi kriteria tabel kategori kelayakan.

BAB IV
HASIL DAN PEMBAHASAN

A. Hasil Perancangan Media Pembelajaran

Penelitian ini menggunakan metode penelitian *Research and Development (R&D)*. Mekanisme penelitian dan pengembangan produk secara detail menggunakan *V-model*, yang terdiri dari 8 mekanisme pembuatan produk; *Requirement modeling, Architectural design, Component design, Code generation, Unit testing, Integration testing, System testing, Acceptance testing* (Pressman, 2010:40) :

1. *Requirement Modeling*

Tahapan ini merupakan aktivitas penggalian informasi untuk menentukan spesifikasi produk yang merepresentasikan kebutuhan *user* (pengguna). Proses penentuan detail produk *software* yang akan dibuat sangat dipengaruhi oleh aktivitas analisis kebutuhan pada tahap ini. Sasaran pengguna pada produk media pendidikan dalam penelitian ini adalah para peserta didik (siswa). Aktivitas penggalian informasi dilakukan dengan wawancara. Tahap wawancara dilakukan pada guru, pengelola laboratorium/bengkel pada Program Keahlian Teknik Mekatronika SMTI (Sekolah Menengah Teknik Industri). Hasil dari tahap mempelajari berbagai informasi yang telah didapatkan, para peserta didik sangat terbatasi dengan permasalahan keterbatasan alat, bahan, waktu (jam belajar dikelas/lab/bengkel) dan sistem belajar yang harus bergilir/berkelompok. Demikian pula permasalahan mahalnya peralatan pembelajaran untuk bidang

Teknik Digital apabila siswa harus memilikinya untuk belajar secara mandiri di rumah. Bidang keilmuan Teknik Digital pada SMK-SMTI dimasukkan dalam Mata Pelajaran Elektronika Digital. Pada dasarnya pembelajaran pada keilmuan Teknik Digital membutuhkan penyediaan perangkat keras *(hardware)* berupa Teknik Digital *Kit-Set* yang dapat membantu para siswa dalam memahami mekanisme kerja secara *hardware*, bukan hanya berupa pemaparan ceramah. Karena secara keilmuan, mekanisme kerja dari sebuah IC *(Integrated Circuit)* tidak dapat dipahami hanya dengan melihat bentuk fisik luarnya saja. Demikian pula pada IC *Logic Gate* pada Teknik Digital yang memiliki banyak varian sistem kinerja.

Hasil mempelajari berbagai permasalahan dari informasi yang telah didapat, dibuatlah konsep sebuah produk media pembelajaran yang dapat mensimulasikan mekanisme kerja dari *hardware* yang sesungguhnya. Konsep produk media pembelajaran Teknik Digital tersebut berupa perangkat lunak *(software)* yang dikemas dalam perangkat yang bersifat mudah diakses/dimiliki siswa, *flexible*, dan *portable*. Teknologi pembelajaran berbasis *mobile* dikerahkan dalam penelitian ini, sebagai teknologi untuk mengakomodasi sifat media pembelajaran yang *flexible* dan *portable* yang dikehendaki. Secara spesifik, produk media pembelajaran berupa perangkat lunak *(software)* dalam tesis ini menggunakan *operating system* (OS) berbasis *mobile device* yang secara data statistik dimiliki oleh mayoritas siswa.

Aktivitas perancangan *software* pembelajaran teknik digital diawali dengan pengumpulan informasi dan data tentang spesifikasi yang dibutuhkan pengguna. Proses pengumpulan data pada tahap *requirement modeling* dilakukan dengan prosedur; (1) pengamatan proses belajar mengajar saat menggunakan perangkat *hardware* teknik digital untuk mengetahui tingkat kelemahan dalam penggunaan media *hardware*, serta tingkat bahaya dan resiko saat proses belajar mengajar dilakukan, (2) melakukan wawancara pada peserta didik maupun guru tentang kesulitan dan kelemahan penggunaan metode belajar mengajar yang selama ini dilakukan, seperti masalah siswa tidak dapat belajar mandiri dirumah karena peralatan terbatas hanya berada di sekolah, aktivitas pembelajaran di sekolah yang harus bergilir karena keterbatasan media ajar, (3) diskusi tentang spesifikasi media efektif dan efisien yang dibutuhkan para peserta didik dan guru dalam proses pembelajaran teknik digital *logic gate*.

Informasi dan berbagai data yang diperoleh menuntut spesifikasi sebuah media pembelajaran teknik digital berupa; (1) perangkat lunak *(software)* berbasis simulasi yang dapat menggantikan keterbatasan perangkat keras *(hardware)* yang ada, serta mampu beroperasi layaknya *hardware* yang sesungguhnya, (2) *software* yang dibuat dapat beroperasi pada perangkat *smartphone*, bukan pada laptop maupun PC *(personal computer)*, karena hampir semua siswa memiliki *smartphone*, namun tidak semua siswa memiliki laptop atau PC, (3) *software* yang dibuat mengakomodasi kebutuhan dasar mata pelajaran teknik digital, seperti materi tentang 6 *logic gate* dasar

(NOT, AND, OR, NAND, NOR, XOR), penyediaan *truth table* sebagai materi pembantu memudahkan proses pembelajaran *logic gate*, (4) siswa dapat mengoperasikan dan memainkan berbagai mekanisme keja dari 6 sistem *logic gate* dasar, sehingga siswa dapat memahami karakteristik sistem dari 6 *logic gate* tersebut.

Proses perancangan dan realisasi berbagai spesifikasi analisis kebutuhan tersebut, maka dibutuhkan berbagai perangkat dan komponen pendukung dalam membuat *software* pembelajaran teknik digital "DigiChip". Berbagai software yang digunakan dalam proses pembuatan adalah sebagai berikut ;

- *Software* UML : digunakan untuk membuat desain alur *user interface* (UI) sebelum semua obyek 2D dan dan 3D yang ada pada *software* "DigiChip" dibuat. Desain UML menggambarkan alur algoritma dari *software* yang akan dibuat.
- *Software* desain 2D : digunakan untuk proses pembuatan berbagai obyek 2D yang ada pada *software* pembelajaran "DigiChip", seperti tombol menu, tombol keluar, dan berbagai tombol UI lainnya. Pembuatan obyek 2D digunakan juga untuk melengkapi proses desain obyek 3D, seperti pembuatan label, logo dan sebagainya.
- *Software* desain 3D : digunakan untuk proses pembuatan berbagai obyek 3D, sebagian besar obyek yang ada pada *software* pembelajaran "DigiChip" adalah berupa obyek 3 dimensi. Karena karakteristik *software* yang dibuat adalah jenis *software* simulasi dan sebuah konsep

virtual, yang tentu menyajikan visual yang sangat mendekati obyek komponen teknik digital yang sesungguhnya.

- *Software Coding* : digunakan untuk penyusunan *script source code* sebagai bahasa pengontrolan pada proses pembuatan *software* "DigiChip". Bahasa *coding* yang digunakan dalam pembuatan media pembelajaran teknik digital ini adalah bahasa C#. Berbagai *file script* yang dibuat, digunakan untuk menjalankan berbagai algoritma yang telah direncanakan pada semua obyek di dalam *software* "DigiChip".
- *Software Engine (Graphics Engine)* : digunakan untuk proses pengolahan dan *compiling* semua obyek 2D dan 3D yang telah dibuat, untuk diintegrasikan dengan *script* bahasa *coding* yang telah dibuat. Sehingga keluaran yang dihasilkan berupa file yang dapat dioperasikan pada sistem operasi (OS) yang dikehendaki.
- *Software* AVD : digunakan untuk menjalankan *software* "DigiChip" yang telah dibuat dalam bentuk tampilan virtual, sehingga mudah untuk diamati dan diujicobakan pada berbagai konfigurasi *hardware* android. AVD *(Android Virtual Device)* dapat mengakomodasi berbagai berbagai perangkat *(device)* yang tidak dimiliki oleh *user*, sehingga memudahkan untuk melakukan pengujian pada berbagai konfigurasi perangkat android.
- *Software Analyzer* : digunakan untuk mengetahui dan mendeteksi berbagai perilaku *source code* yang telah dibuat, sehingga berbagai *error* pada saat *software* dioperasikan pada berbagai kondisi pengoperasian dapat diminimalkan.

2. *Architectural Design*

Desain arsitektural merupakan tahap perencanaan detail dari rancangan produk *software* media pembelajaran yang akan dibuat. Desain produk *software* media pembelajaran pada tesis ini dirancang dalam bentuk 3D, sebagai upaya untuk menjaga kedekatan secara visual dengan *hardware* yang sesungguhnya. Nama dari produk *software* media pembelajaran yang dibuat pada penelitian ini adalah bernama "DigiChip". Nama "DigiChip" diambil dari filosofi kata “Digital” dan “Chip”. Kata “Digital” karena produk *software* yang dibuat pada penelitian ini ditujukan untuk media pembelajaran keilmuan Teknik Digital. Kata “Chip” yang menjelaskan pada sebuah komponen perangkat keras *(hardware)* yang lazim digunakan untuk penyebutan komponen berjenis IC *(integrated circuit)*. Karena pada dasarnya *software* media pembelajaran yang dibuat dalam penelitian ini memiliki karakteristik berupa mensimulasikan mekanisme kerja dari IC, lebih spesifiknya IC *Logic Gate*.

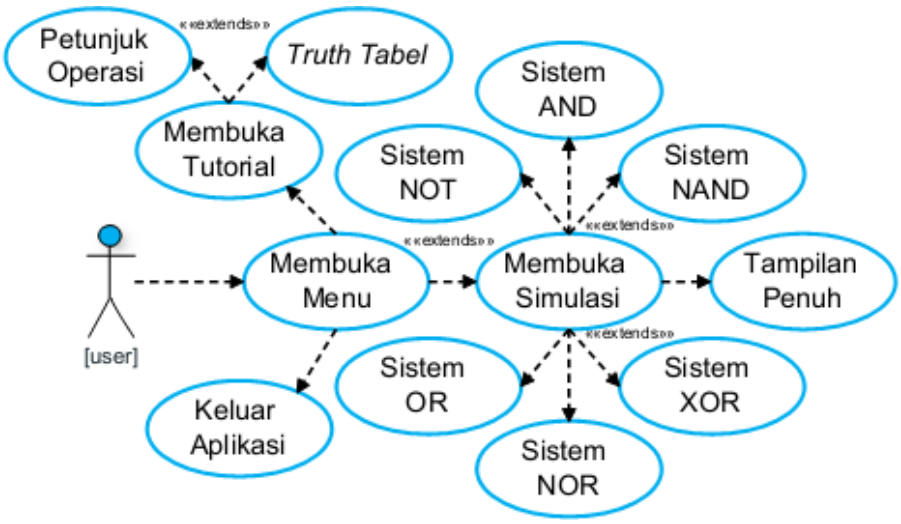
Desain arsitektural merupakan pembuatan sebuah ekosistem untuk mengakomodasi interaksi antara *user* (pengguna) dengan mesin (perangkat *mobile*). Desain terdiri dari berbagai fitur menu yang menyajikan kebutuhan pengguna dalam konteks keilmuan Teknik Digital. Faktor kemudahan akses dan kemudahan dalam memahami logo maupun simbol pada berbagai fitur sangat diperhitungkan dengan matang dalam tahapan proses ini. Tujuan utama adalah menciptakan sebuah sistem yang dapat saling dimengerti antara *user* dengan mesin. Sehingga berlangsung sebuah interaksi yang dapat

meminimalkan berbagai keanehan sistem antara *user* dan mesin dalam produk *software* media pembelajaran Teknik Digital yang dibuat pada penelitian ini. Desain arsitektural pada produk "DigiChip" ini terdiri dari dua elemen, yaitu perancangan desain *User Experience* (UX) dan desain *User Interface* (UI).

a. Desain *User Experience* (UX)

Proses perancangan desain *User Experience* (UX) pada produk "DigiChip" ini menggunakan bahasa UML dalam pemodelan rancangan sistem. Mekanisme sistem pada produk penelitian ini akan diuraikan dalam tiga diagram pokok bahasa pemodelan, yaitu berupa *use case diagram*, *activity diagram* dan *sequence diagram*.

1) *Use Case Diagram*



Gambar 72. *Use Case Diagram Software* "DigiChip"

Skema pada *use case diagram* memuat mekanisme kerja dari sistem produk "DigiChip" yang telah dirancang untuk dapat diakses dan dioperasikan oleh pengguna *(user).* Skema *use case diagram* menampilkan mekanisme pengguna untuk mengoperasikan berbagai fitur yang dirancang pada *software* "DigiChip" untuk perangkat *mobile*. Pemodelan dengan *use case diagram* memberikan gambaran sistematika sebuah operasional berbagai fitur dalam produk *software* media pembelajaran Teknik Digital yang dibuat.

a) Mekanisme Membuka Menu

Tabel 23. Mekanisme Membuka Menu

Aksi *User*	Reaksi *System*
1. Menekan tombol “Menu”	
	2. Menampilkan berbagai elemen sub menu dari tombol “Menu”

b) Mekanisme Membuka Simulasi

Tabel 24. Mekanisme Membuka Simulasi

Aksi *User*	Reaksi *System*
1. Menekan tombol “Mulai”	
	2. Menampilkan berbagai pilihan simulasi dari sistem gerbang logika dasar *(Logic Gate)* untuk dioperasikan, serta tombol untuk menampilkan secara keseluruhan dari *board* simulasi Teknik Digital

c) Mekanisme Membuka Sistem *Logic Gate* NOT

Tabel 25. Mekanisme Membuka Sistem NOT

Aksi *User*	Reaksi *System*
1. Menekan tombol "Sistem NOT"	
	2. Menampilkan komponen IC gerbang logika NOT beserta perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dari sistem gerbang logika dasar NOT	

d) Mekanisme Membuka Sistem *Logic Gate* AND

Tabel 26. Mekanisme Membuka Sistem AND

Aksi *User*	Reaksi *System*
1. Menekan tombol "Sistem AND"	
	2. Menampilkan komponen IC gerbang logika AND beserta perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dari sistem gerbang logika dasar AND	

e) Mekanisme Membuka Sistem *Logic Gate* NAND

Tabel 27. Mekanisme Membuka Sistem NAND

Aksi *User*	Reaksi *System*
1. Menekan tombol "Sistem NAND"	
	2. Menampilkan komponen IC gerbang logika NAND beserta perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dari sistem gerbang logika dasar NAND	

f) Mekanisme Membuka Sistem *Logic Gate* OR

Tabel 28. Mekanisme Membuka Sistem OR

Aksi *User*	Reaksi *System*
1. Menekan tombol "Sistem OR"	
	2. Menampilkan komponen IC gerbang logika OR beserta perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dari sistem gerbang logika dasar OR	

g) Mekanisme Membuka Sistem *Logic Gate* NOR

Tabel 29. Mekanisme Membuka Sistem NOR

Aksi *User*	Reaksi *System*
1. Menekan tombol "Sistem NOR"	
	2. Menampilkan komponen IC gerbang logika NOR beserta perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dari sistem gerbang logika dasar NOR	

h) Mekanisme Membuka Sistem *Logic Gate* XOR

Tabel 30. Mekanisme Membuka Sistem XOR

Aksi *User*	Reaksi *System*
1. Menekan tombol "Sistem XOR"	
	2. Menampilkan komponen IC gerbang logika XOR beserta perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dari sistem gerbang logika dasar XOR	

i) Mekanisme Membuka Tampilan Penuh

Tabel 31. Mekanisme Membuka Tampilan Penuh

Aksi *User*	Reaksi *System*
1. Menekan tombol "Tampilan Penuh"	
	2. Menampilkan komponen IC gerbang logika secara keseluruhan, lengkap bersama perlengkapan pendukung simulasi seperti, lampu LED, *switch*, soket IC, soket LED, soket *switch*, kabel VCC-GND
3. Mengoperasikan simulasi dengan tampilan secara global pada *board* gerbang logika Teknik Digital "DigiChip", dengan berbagai fasilitas *view (zoom-in*, *zoom-out*, *rotate*, *move*)	

j) Mekanisme Membuka Tutorial

Tabel 32. Mekanisme Membuka Tutorial

Aksi *User*	Reaksi *System*
1. Menekan tombol "Tutor"	
	2. Menampilkan berbagai pilihan dalam sub menu tutorial, berupa Petunjuk Pengoperasian dan *Truth Tabel* Gerbang Logika

k) Mekanisme Membuka Petunjuk Pengoperasian

Tabel 33. Mekanisme Membuka Petunjuk Pengoperasian

Aksi *User*	Reaksi *System*
1. Menekan tombol "Petunjuk Pengoperasian"	
	2. Menampilkan berbagai *slide* halaman tutorial petunjuk operasional *software* "DigiChip"
3. Membaca berbagai halaman petunjuk pengoperasian, dengan fasilitas tampilan *slide* untuk membaca halaman selanjutnya atau halaman sebelumnya	

l) Mekanisme Membuka *Truth Tabel* Gerbang Logika

Tabel 34. Mekanisme Membuka *Truth Tabel* Gerbang Logika

Aksi *User*	Reaksi *System*
1. Menekan tombol "*Truth Tabel* Gerbang Logika"	
	2. Menampilkan berbagai *Truth Tabel* dari enam karakteristik gerbang logika dasar (NOT, AND,NAND, OR, NOR, XOR) pada LCD obyek 3D "DigiChip"
3. Membaca berbagai *Truth Tabel* dengan tampilan LCD pada obyek 3D, dengan fasilitas tombol 3D pada *board* Teknik Digital "DigiChip"	

m) Mekanisme Keluar Aplikasi

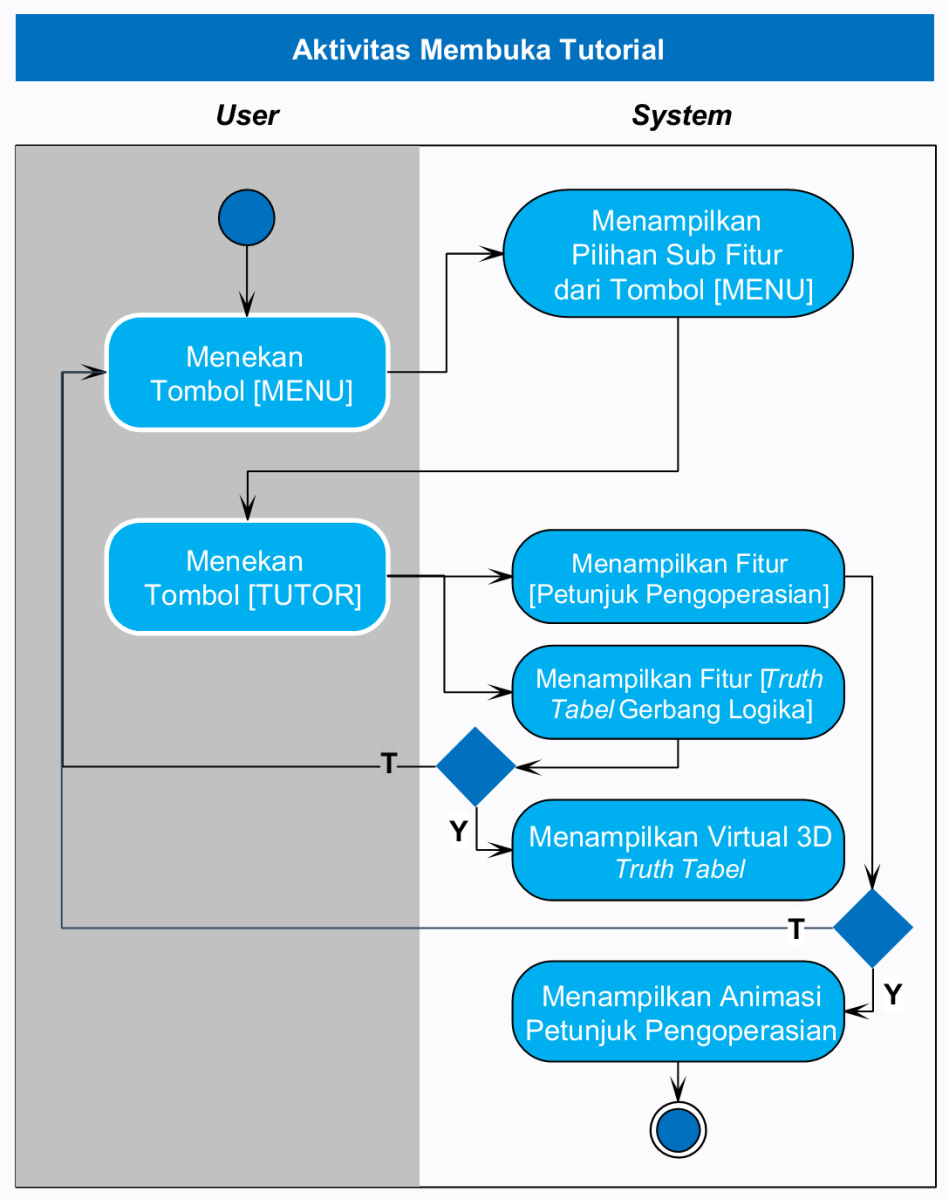
Tabel 35. Mekanisme Keluar Aplikasi

Aksi *User*	Reaksi *System*
1. Menekan tombol “Keluar”	
	2. Sistem akan menghentikan dan mengakhiri kerja aplikasi

2) *Activity Diagram*

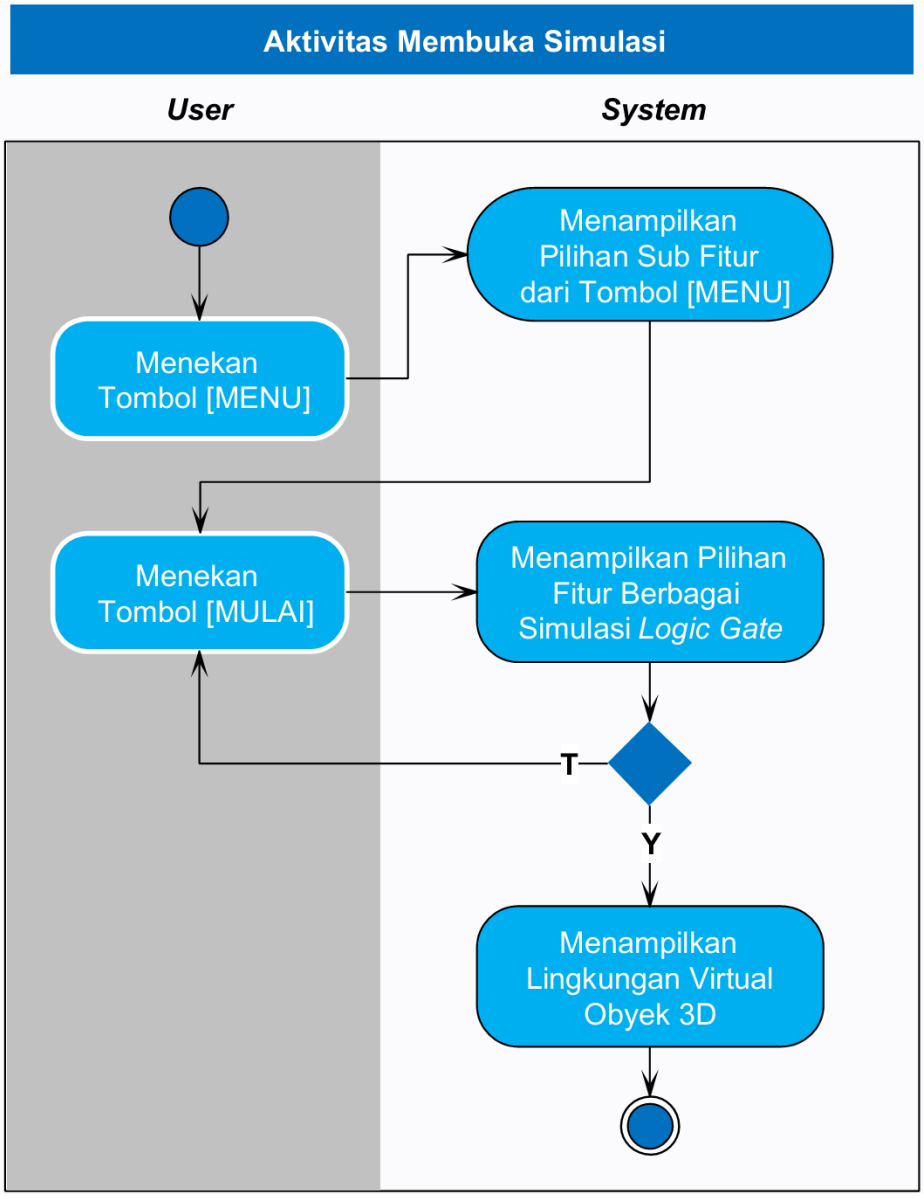
Proses berikutnya dalam perancangan *software* media pembelajaran Teknik Digital "DigiChip" adalah pembuatan diagram aktivitas. Sebuah diagram aktivitas menjelaskan bagaimana sebuah *software* bekerja secara responsif terhadap instruksi atau stimulus yang diberikan oleh pengguna *(user).* Diagram *activity* yang dibuat merupakan gambaran dari aktivitas interaksi antara *user* dengan sistem perangkat lunak seusai dengan rencana yang diharapkan. Diagram aktivitas dimaksudkan untuk menunjukkan kegiatan yang membentuk proses sistem dan aliran kontrol dari satu kegiatan ke kegiatan lain. Awal proses ditunjukkan oleh *filled circle*, diakhiri dengan *filled circle inside another circle*. Persegi panjang dengan sudut bulat mewakili aktivitas. Tanda panah pada diagram aktivitas mewakili aliran pekerjaan dari satu aktivitas ke aktivitas lainnya. Garis padat digunakan untuk menunjukkan koordinasi aktivitas, ketika aliran lebih dari satu aktivitas mengarah ke garis yang solid. Aliran dari garis padat yang mengarah ke sejumlah kegiatan, dapat dieksekusi secara paralel (Sommerville, 2011:123).

a) Aktivitas Tutorial



Gambar 73. Aktivitas Membuka Tutorial

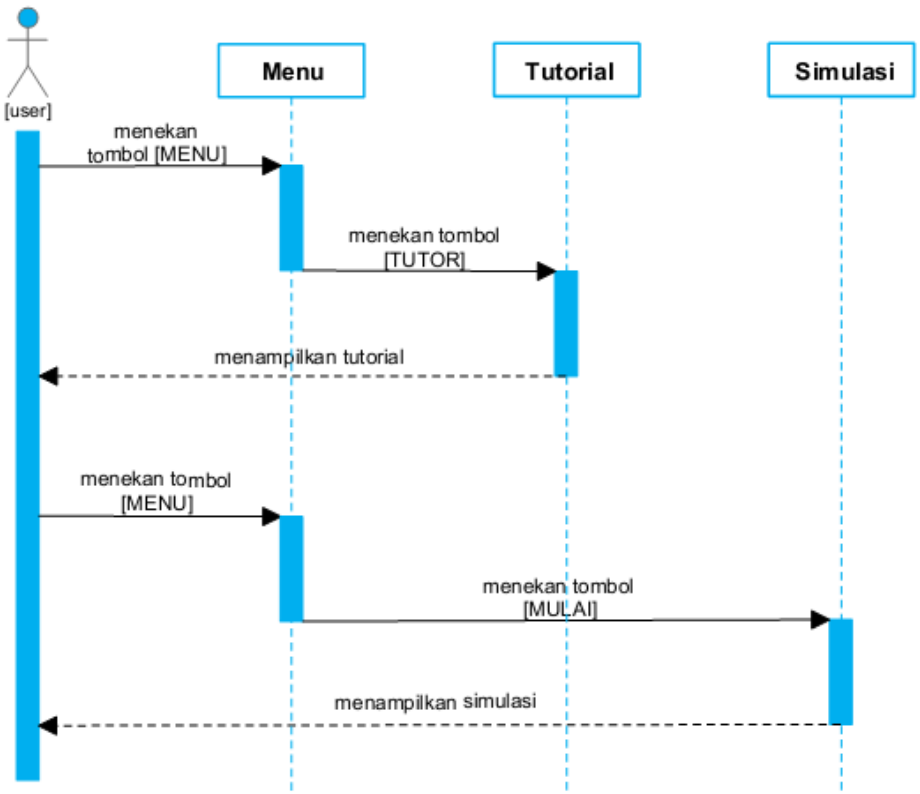
b) Aktivitas Simulasi



Gambar 74. Aktivitas Membuka Simulasi

3) *Sequence Diagram*

Diagram sekuensial dalam UML digunakan untuk memodelkan interaksi antara aktor dan obyek dalam sistem dan interaksi antara objek itu sendiri. Diagram sekuensial menunjukkan urutan interaksi itu terjadi selama *use case* tertentu atau *use case instance*. Aktor dan obyek yang terlibat, tercantum di bagian atas diagram dengan garis putus-putus secara vertikal. Interaksi antara obyek ditunjukkan oleh panah beranotasi. Persegi panjang pada garis putus-putus menunjukkan garis hidup obyek. Notasi pada panah menunjukkan panggilan ke obyek, parameternya, dan nilai kembali. Aktivitas interaksi *user* dengan sistem digambarkan dalam sebuah *timeline* (Sommerville, 2011:126).

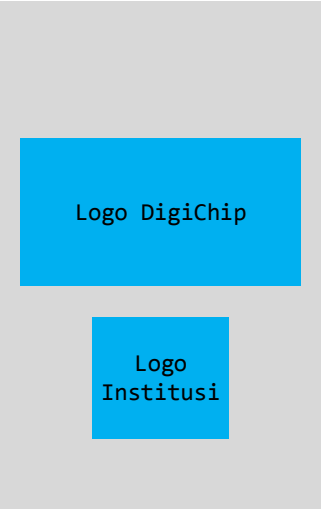


Gambar 75. *Sequence Diagram* "DigiChip"

b. Desain *User Interface* (UI)

Proses perancangan desain *user interface* (UI) pada *software* "DigiChip" dilakukan sebagai upaya untuk menyesuaikan kenyamanan *user* saat berinteraksi dengan perangkat *(mobile device)* secara langsung melalui *touchscreen*. Faktor kenyamanan operasional *(ergonomic)* mengacu pada penempatan berbagai posisi komponen UI pada layar sehingga nyaman untuk dioperasikan dengan satu genggaman tangan maupun dua genggaman tangan. Penempatan berbagai posisi komponen UI pada *software* "DigiChip" dirancang berdasarkan faktor pertimbangan lain juga, yaitu pertimbangan resolusi layar maupun rasio resolusi layar. Tampilan UI pada *software* "DigiChip" diuraikan dalam berbagai halaman/layar, diantaranya:

1) Layar *Splash*



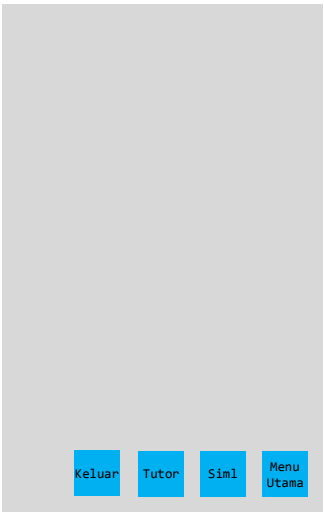
Gambar 76. UI Layar *Splash*

Desain pada layar *splash* berfungsi sebagai identitas secara visual ketika *software* mulai dioperasikan. Layar *splash* memuat

tentang logo *software* "DigiChip" dan logo vendor, dalam hal ini adalah institusi perguruan tinggi perancang produk. Layar *splash* yang dibuat memiliki ukuran resolusi layar 1280x720 pixel.

2) Layar *Home*

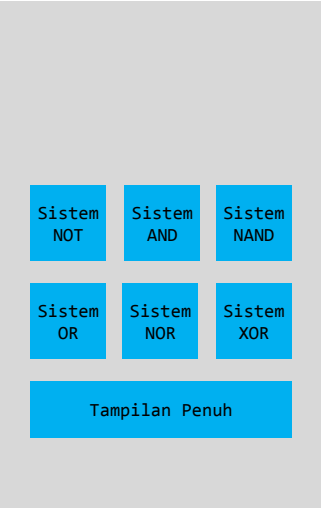
Desain pada layar *home* menyajikan berbagai menu dan fitur yang ada pada *software* "DigiChip". Sebagai pertimbangan faktor ergonomis, tombol menu utama diposisikan pada layar kanan bawah, karena semua sasaran pengguna *(user)* bukan penganut tangan kidal. Sehingga jempol tangan kanan lebih nyaman dalam mengakses tombol menu utama. Sebagai pertimbangan kenyamanan agar dapat menikmati tampilan layar secara keseluruhan saat menjalankan simulasi dan tanpa terhalang berbagai tombol menu, maka rangkaian tombol menu didesain berkarakter *show-hide*.



Gambar 77. UI Layar *Home*

3) Layar Simulasi

Desain pada layar simulasi pada *software* "DigiChip" menyajikan berbagai pilihan menu dari enam gerbang logika dasar beserta fitur untuk menampilkan *board* Teknik Digital secara utuh/keseluruhan. Ketika salah satu pilihan fitur dipilih oleh *user*, maka akan mengantarkan *user* pada sebuah dunia tiga dimensi (3D) untuk mengoperasikan simulasi *logic gate* secara *real-world*. Posisi semua menu simulasi *logic gate* didesain prioritas mendekat kearah bawah, sebagai wujud perhitungan desain secara ergonomis agar mudah dijangkau oleh jempol tangan kanan.



Gambar 78. UI Layar Menu Simulasi

4) Layar Tutorial

Desain pada layar tutorial berisi tentang berbagai fitur berupa *truth tabel* gerbang logika dan petunjuk operasional dari fitur simulasi *software* "DigiChip". Fitur *truth tabel* disediakan untuk membantu *user*

saat mengoperasikan simulasi IC gerbang logika ketika tidak begitu mengingat karakter kinerja sistem masing-masing *logic gate*. Fitur *truth tabel* disajikan dalam bentuk 3D yang menyatu pada komponen 3D berupa LCD monitor, agar memudahkan *user* ketika ingin melihat *truth tabel* karena harus sambil mengoperasikan simulasi.

Gambar 79. UI Layar Menu Tutorial

3. *Component Design*

Proses pada tahap desain komponen merupakan aktivitas perancangan berbagai komponen 2D dan 3D yang diperlukan dalam merealisasikan *software* media pembelajaran Teknik Digital "DigiChip". Aktivitas desain komponen pada perancangan *software* "DigiChip" ini melibatkan sekitar empat *software-builder* berupa spesialisasi untuk perancangan obyek 2D, pewarnaan obyek 2D, pembuatan *skin* 2D, perancang *shape* 3D, pewarnaan obyek 3D, serta pembuatan tekstur dan material 3D. Berbagai komponen

obyek 3D dalam *software* "DigiChip" seperti IC, soket IC, LED, *port* LED, *switch*, dirancang memiliki ukuran dimensi dan rasio skala sama persis dengan komponen obyek yang sebenarnya, dengan mengacu pada spesifikasi *datasheet* komponen. Proses demikian dilakukan sebagai upaya pembuatan sebuah dunia tiga dimensi yang memiliki realitas benar-benar menyerupai obyek yang sesungguhnya. Sehingga pada produk *software* "DigiChip" tidak akan dijumpai berbagai keanehan bentuk, rasio (perbandingan) skala, ukuran, dan keanehan visual yang lain pada obyek/komponen.

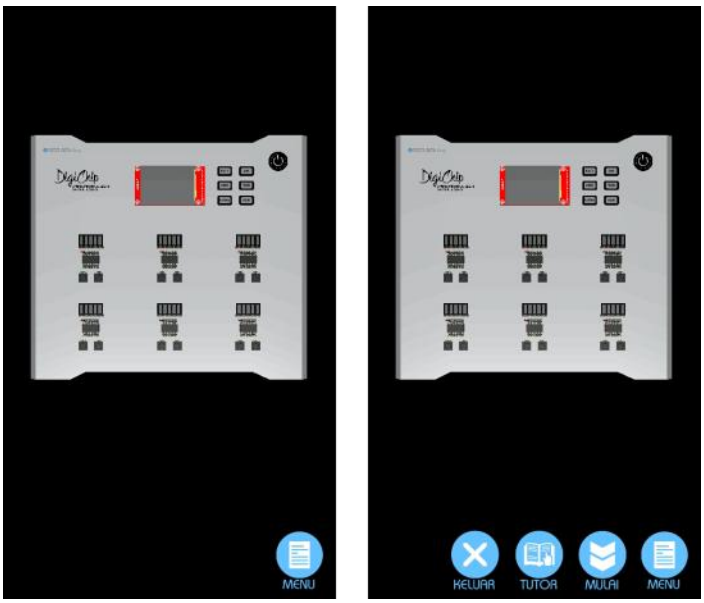
a. *Splash Screen*

Desain *splash screen* pada *software* "DigiChip" menampilkan berupa logo produk dan institusi pembuat produk. Pada bagian kiri atas menampilkan visualisasi proses *loading* dari sistem dalam 8 detik *countdown* untuk memuat semua data pada aplikasi "DigiChip".

Gambar 80. *Splash Screen* "DigiChip"

b. *Home*

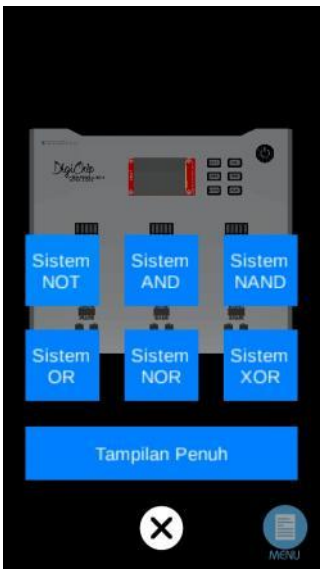
Istilah *home* disini adalah merujuk pada istilah yang sering digunakan para perancang *software* untuk penyebutan sebuah *layout* utama dalam sebuah sistem penampil berbagai rangkaian fitur. Produk *software* "DigiChip" menempatkan tombol menu utama pada halaman *home* ini. Tombol menu pada produk "DigiChip" ini dirancang berkarakter *show-hide*, sebagai pertimbangan kenyamanan agar dapat menikmati tampilan layar secara keseluruhan saat menjalankan simulasi, tanpa terhalang rangkaian tombol sub menu dari menu utama (tombol "Menu"). Halaman *home* pada "DigiChip" dirancang kolaboratif antara obyek-obyek 2D dengan 3D menjadi satu kesatuan. Berbagai tombol menu 2D memiliki *background* obyek 3D *board* Teknik Digital "DigiChip" secara langsung. Sebagai pertimbangan kenyamanan aksesibilitas serta unsur karakter artistik produk "DigiChip".



Gambar 81. *Home* "DigiChip" Sebelum-Sesudah Stimulus Tombol Menu

c. Simulasi

Fitur simulasi sebagai fitur utama pada *software* "DigiChip" dirancang dengan tampilan pengantar berupa komponen UI yang menyediakan pilihan sistem *logic gate* Teknik Digital yang ingin dimainkan/dioperasikan oleh *user*. Jika *user* sudah memilih salah satu



Gambar 82. Menu Simulasi "DigiChip"

fitur simulasi yang disajikan, *user* akan diantarkan pada sebuah dunia 3D untuk menikmati berbagai komponen elektronik virtual simulasi dari sistem *logic gate* Teknik Digital yang terdiri dari IC, soket IC, LED, *port* LED, *switch*, LCD *button*, LCD *screen*, yang semuanya dikemas dalam sebuah perangkat virtual bernama *virtual board* "DigiChip".

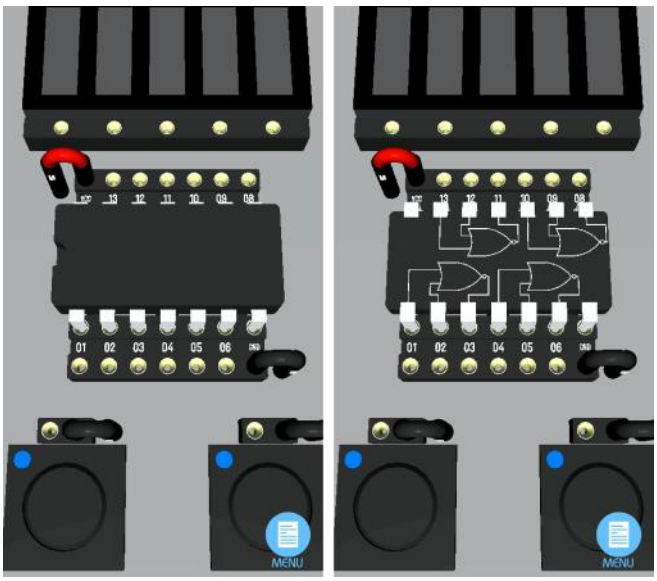
Obyek dan berbagai komponen 3D dapat dikendalikan melalui stimulus dari *user* yang diberikan pada sistem melalui *touchscreen*. Mekanisme kerja ini sepenuhnya dikerjakan oleh *script/source code* yang telah ditanamkan dalam sistem melalui aktivitas *coding (programming)*.

Mengenai aktivitas *coding* merupakan tahapan selanjutnya dalam proses perancangan *software* "DigiChip" ini, yang akan diuraikan pada bahasan selanjutnya. Aktivitas pengontrolan obyek 3D oleh *user* dilakukan dengan teknik/kaidah bahasa komunikasi antara manusia dengan mesin berupa sandi *gesture*. Teknik *gesture* pada *touchscreen* telah disajikan di kajian teori pada bab sebelunya, yang terdiri dari gerakan *swipe, pinch, tap, double-tap, tap+drag, long press, rotate* dan lainnya.

Gambar 83. Pengendalian Komponen Obyek 3D Menggunakan *Gesture*

Perancangan sebuah desain dari komponen IC *(integrated circuit)* menggunakan teknik animasi, agar dapat dipahami pengguna *(user)* untuk mengetahui tampilan skematik dari IC bersangkutan. Pada umumnya untuk mengetahui alur dari skematik sirkuit IC dilakukan dengan cara membuka *datasheet* spesifikasi IC yang disediakan produsen. Sebuah sirkuit pada IC tidak dapat dilihat secara langsung,

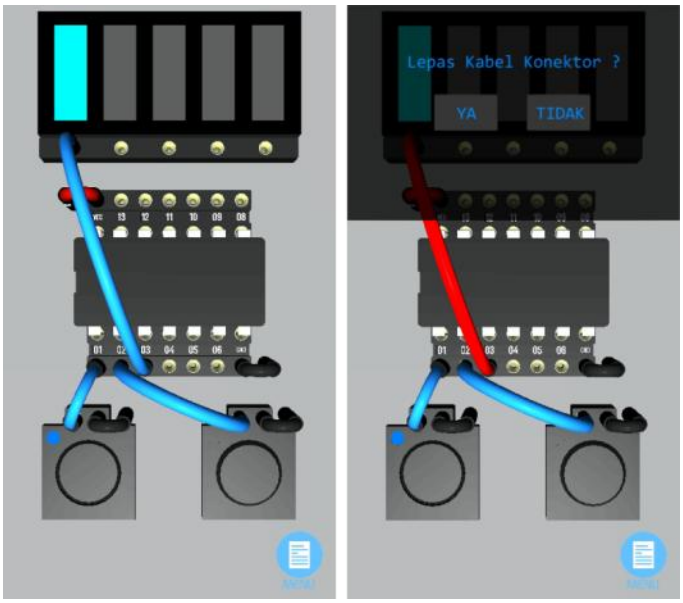
karena dalam kondisi tertanam didalam material silikon hitam pada pembungkus IC. Untuk mengetahui skematik sirkuit dari masing- masing IC, pada produk *software* "DigiChip" ini disediakan sebuah algoritma penampil sirkuit hanya dengan *user* melakukan sentuhan pada komponen IC yang diinginkan.



Gambar 84. Perancangan Algoritma Menampilkan Skematik Sirkuit IC

Algoritma lain yang rancang untuk komponen 3D pada produk "DigiChip" ini adalah pengaturan pada operasional instalasi kabel untuk aktivitas simulasi. Pemasangan kabel penghubung antar *port* dirancang memiliki algoritma untuk merespon instruksi dari *user* sesuai kehendak yang diinginkan *user*. Algoritma pada kabel penghubung disempurnakan dengan pemberian unsur animatif pada visualisasi warna kabel. Perancangan tersebut sebagai bentuk perhitungan aspek ergonomis untuk pengguna, agar memudahkan *user* pada aktivitas simulasi dalam

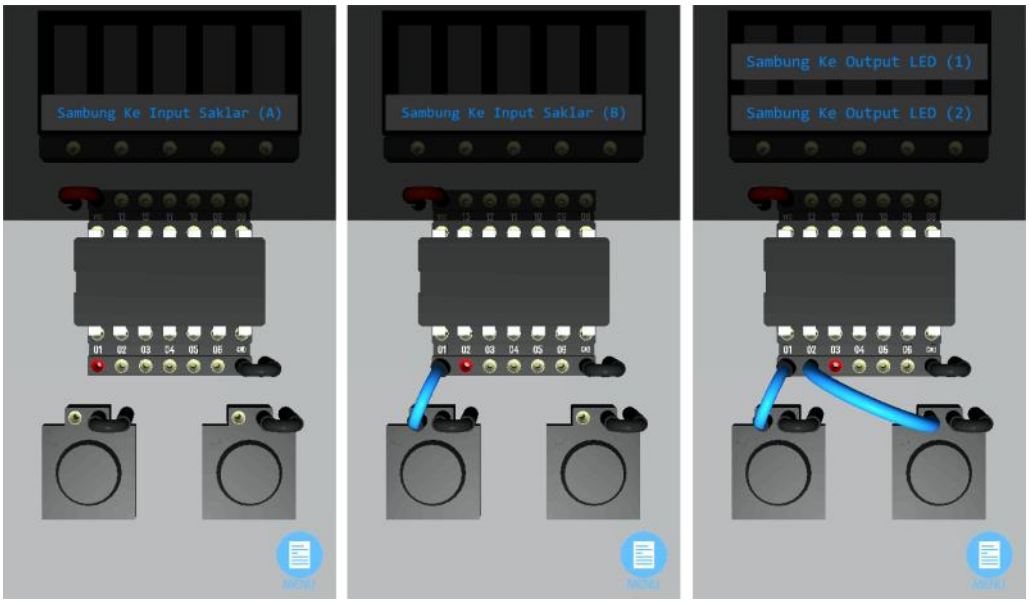
membedakan kabel mana yang telah menerima stimulus dari *user* dan kabel mana yang tidak. Kabel penghubung yang menerima stimulus dari *user* akan menjadi berwarna merah, dari warna semula yang berwarna biru.



Gambar 85. Perancangan Algoritma Untuk Animasi Kabel Penghubung

Perancangan algoritma animatif pada *software* media pembelajaran Teknik Digital "DigiChip" ini juga ditanamkan pada obyek 3D berupa komponen *port* IC, baik *port-in* maupun pada *port-out*. Sehingga apabila *user* memberikan stimulus/instruksi pada salah satu *port* yang dikehendaki, maka *port* bersangkutan akan berwarna merah dari yang sebelumnya memiliki warna alami kuning tembaga. Komponen *port-in* pada soket IC dalam *software* "DigiChip" ini dirancang untuk menerima dan terhubung dengan komponen *input*, dalam hal ini adalah *switch* melalui *port switch*. Komponen *port-out* dirancang untuk berinteraksi

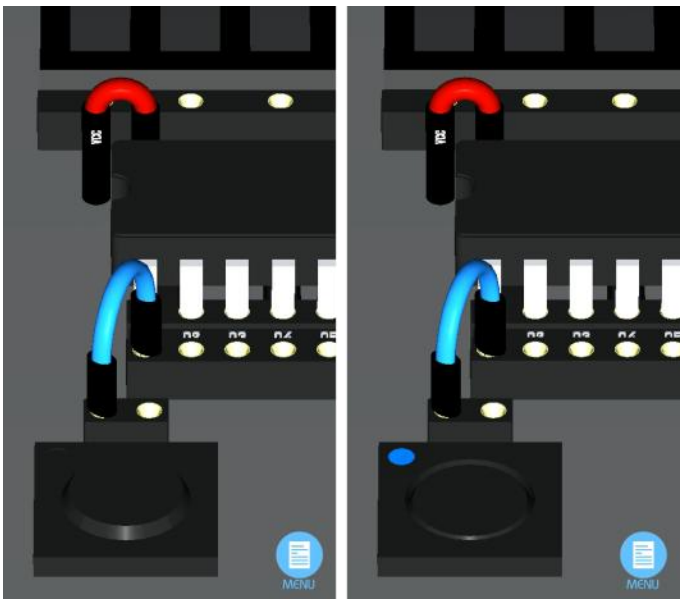
dengan LED melalui *port* LED. Stimulus dari *user* pada salah satu *port* hanya akan memberikan instruksi untuk aktivitas pemasangan kabel penghubung. Stimulus pada kabel penghubung akan memberikan konfirmasi tentang aktivitas pelepasan kabel penghubung atau tidak.



Gambar 86. Perancangan Algoritma Untuk Animasi *Port* IC

Perancangan desain komponen lainnya yang memberikan nuansa realis layaknya pada dunia yang sesungguhnya, pada *software* simulasi "DigiChip" ini adalah perancangan desain *switch* 3D. Desain *switch* dirancang memiliki mekanisme layaknya pada komponen sesungguhnya pada dunia nyata dalam *software* "DigiChip" ini. Komponen *switch* dalam bentuk 3D dirancang dapat memiliki mekanisme *push-down* saat diberi stimulus oleh pengguna *(user).* Komponen *switch* akan kembali ke posisi semula ketika diberi stimulus berikutnya oleh *user.* Komponen *switch* pada *software* "DigiChip" ini didesain pula memiliki indikator

LED kecil yang berada pada sudut kiri atas. Fungsi dari desain yang demikian bertujuan untuk memberikan indikator dan informasi kepada pengguna bahwa *switch* sedang dalam kondisi menyala (menghubungkan arus) atau tidak. Komponen 3D berupa *switch* dalam *software* "DigiChip" ini didesain akan memberikan informasi dengan menyalanya lampu LED indikator jika dalam kondisi *"on"*, sebaliknya jika dalam kondisi *"off"* lampu indikator LED mati.



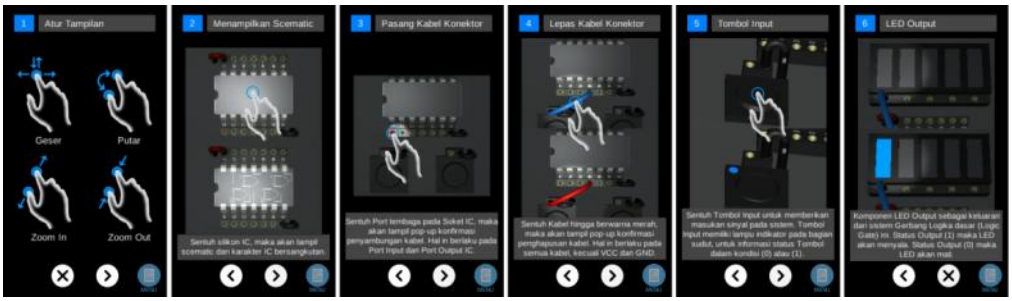
Gambar 87. Perancangan Mekanisme Realis Komponen *Switch*

d. Tutorial

Perancangan fitur tutorial pada *software* "DigiChip" ini menyajikan dua poin fasilitas yang dapat diakses *user* untuk kelancaran aktivitas simulasi gerbang logika dasar pada produk "DigiChip". Dua poin tersebut terdiri dari fasilitas tutorial Petunjuk Pengoperasian dan *Truth Tabel* Gerbang Logika. Pemilihan menu

Gambar 88. Menu Tutorial "DigiChip"

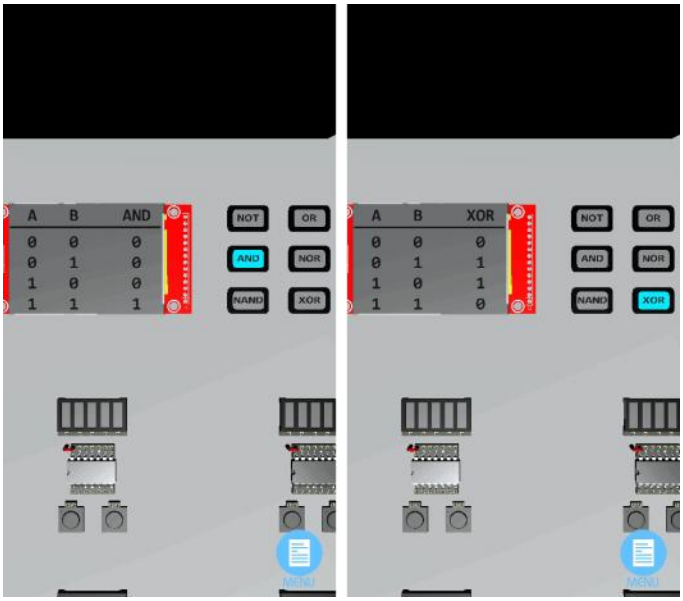
Petunjuk Pengoperasian oleh *user* akan memberikan akses pada serangkaian halaman *slide* yang berisi berbagai panduan pengoperasian *software* "DigiChip". Menu Petunjuk Pengoperasian memberikan fasilitas tutorial dalam pengoperasian fitur utama "DigiChip", yaitu simulasi gerbang logika dasar pada ilmu Teknik Digital.



Gambar 89. Tutorial Petunjuk Pengoperasian

Fasilitas yang kedua pada menu tutorial adalah *Truth Tabel* Gerbang Logika yang memberikan pedoman berupa karakteristik

masing-masing IC *logic gate* yang dapat membantu *user* dalam mengoperasikan fitur simulasi gerbang logika dasar.



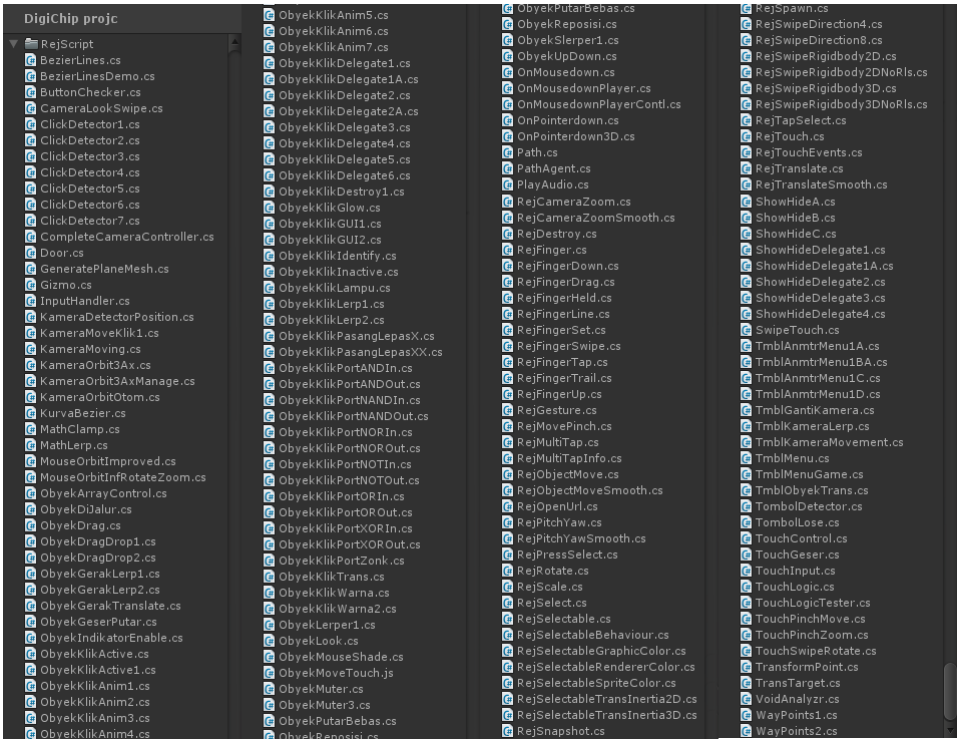
Gambar 90. Komponen LCD *Screen* Memuat *Truth Tabel*

4. *Code Generation*

Tahapan *code generation* pada proses perancangan *software* "DigiChip" merupakan aktivitas penyusunan skrip bahasa C# untuk ditanamkan pada sistem algoritma yang telah direncanakan. Tahapan penyusunan skrip kode untuk kemudian digunakan sebagai *generator* obyek 2D dan 3D merupakan pekerjaan yang paling menghabiskan banyak waktu dan segalanya pada proses perancangan *software* "DigiChip". Fenomena yang lazim dihadapi para perancang skrip kode *(programmer)* tidak terlewatkan juga pada proses perancangan *software* "DigiChip" pada tesis ini, seperti *crash-syntax* antar berbagai *script*, *crash-syntax* antar sesama *script* obyek 2D, antar sesama *script* obyek 3D, antara *script* obyek 2D dengan

script obyek 3D, antara *script* semua obyek dengan berbagai *animate effect*, dan masih banyak lagi yang tidak efektif disebutkan satu per satu.

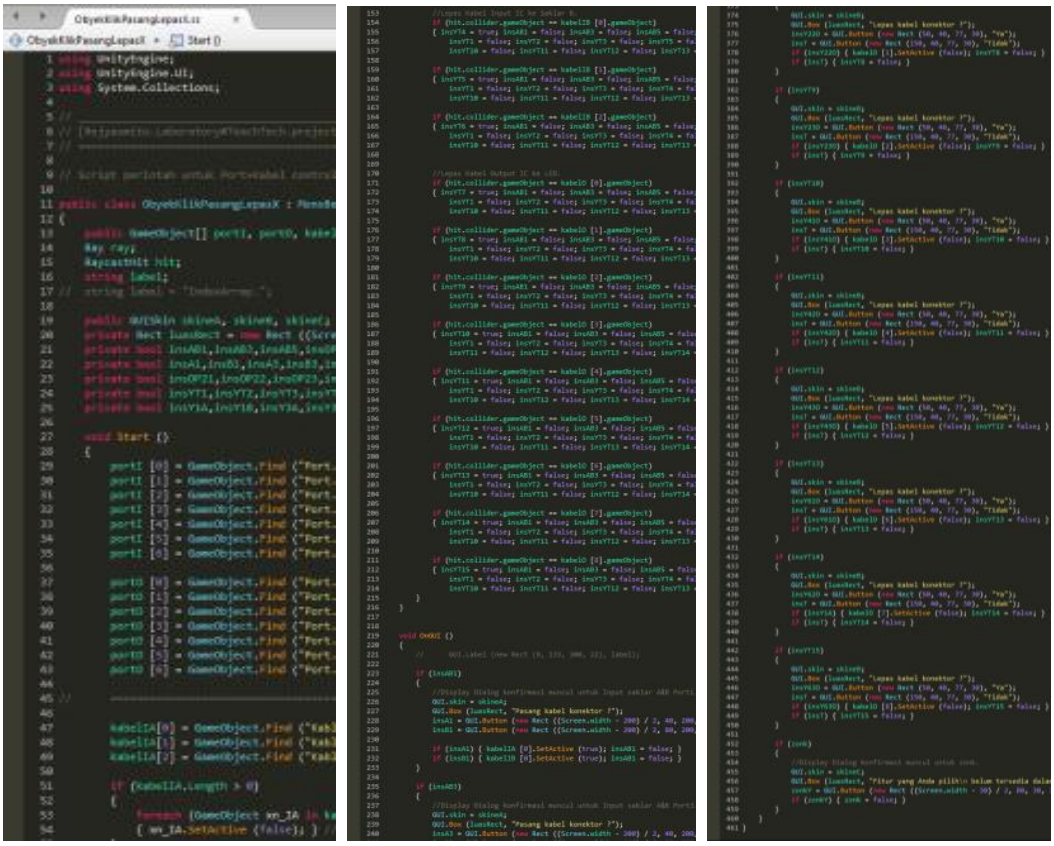
Proses perancangan *software* media pembelajaran Teknik Digital "DigiChip" ini membutuhkan *script* C# sekitar 180 *file*. Setiap *script* memiliki spesifikasi dan kerumitan tersendiri untuk aktivitas pengontrolan dan pengendalian obyek maupun efek pada *software* "DigiChip". Proses pembuatan *script* C# dengan ekstensi *file* (.cs) tidak berbeda dengan bahasa *coding* yang lain, memiliki struktur bahasa dan sistematika *syntax* yang berorientasi obyek untuk sasaran *embedded*.



Gambar 91. Skrip "DigiChip" Dalam *File* Bahasa C# (.cs)

Setiap *script* atau *file* bahasa C# (.cs) berisi serangkaian bahasa *coding (programming)* untuk digunakan sebagai pengatur setiap obyek yang terhimpun dalam *software* "DigiChip". Obyek yang terdapat pada *software*

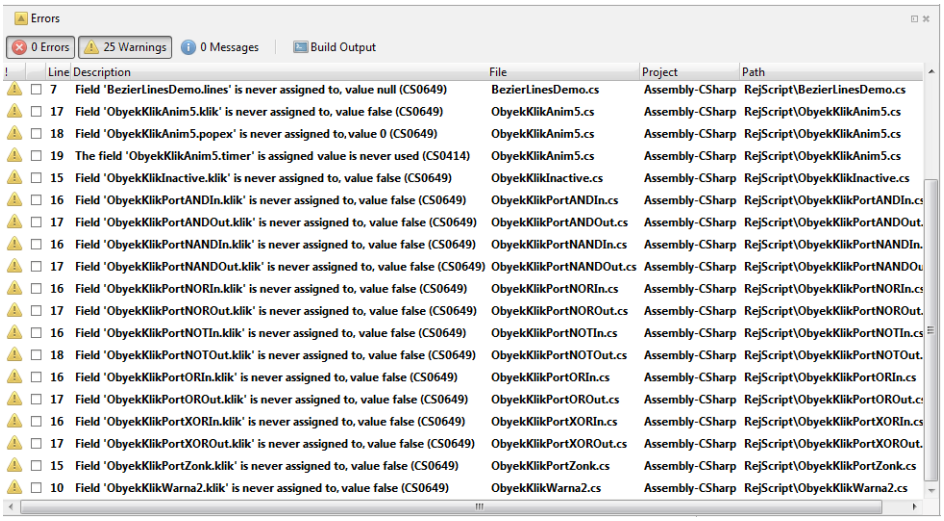
media pembelajaran “DigiChip” sangat beragam, dalam bentuk 2D, 3D, animasi, efek warna, efek material, partikel, dan lainnya. Susunan bahasa C# dalam sebuah *script* memiliki struktur bahasa yang tidak jauh berbeda dengan struktur bahasa *coding* yang lain, yaitu terdiri dari karakter berupa pendeklarasian *class*, *method*, *function*, *string* dan sebagainya. Berikut ini contoh salah satu dari 180 *script* yang digunakan dalam proses perancangan aplikasi media pembelajaran “DigiChip” ;



Gambar 92. Skrip Pengatur Simulasi 3D Koneksi Kabel

Proses perancangan algoritma *software* “DigiChip” berlanjut pada penyusunan *source-code* menjadi dalam bentuk *script* dengan bahasa *coding (programming)* C#. Contoh salah satu *file script* dari 116 *file script* yang

dibuat dalam perancangan “DigiChip” dapat dilihat pada **(Lampiran 5)**. Susunan sebuah *script* sangat tergantung mekanisme apa yang diinginkan/didesain pada obyek bersangkutan, baik ditinjau dari kapasitas dari *script*, sistematika *script*, maupun sisi kompleksitas *script* yang dibuat. Bahasa C# yang telah tersusun sesuai desain perancang *script (programmer)*, agar dapat dimengerti oleh mesin maka dilakukan *debugging*. Proses *debugging* sistem akan mengungkap apakah kemauan manusia perancang perintah sejalan dengan kemauan dari mesin (sistem). Jika terdapat ketidaksesuaian, maka sistem akan mengungkapkan dalam bentuk respon *debugging*, yang terdiri dari *listing errors*, *warnings*, *messages*. Proses perancangan *script* kode hingga pada proses *debugging* inilah yang dimaknai sebagai tahap pembangkitan kode atau *code generation*.



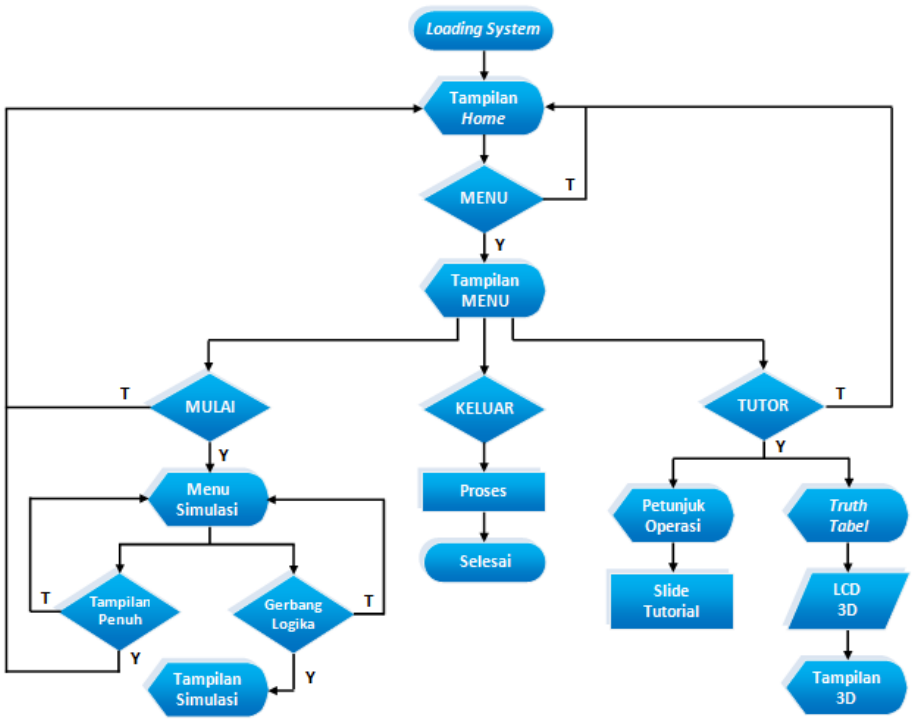
Gambar 93. Proses *Debugging Script* "DigiChip"

5. *Unit Testing*

Pengujian unit merupakan tahapan pengujian paling awal setelah desain *layout*, konten, algoritma *coding* selesai dilakukan pada proses perancangan *software* media pembelajaran "DigiChip". Tahap pengujian unit dilakukan dengan menggunakan prosedur pengujian *white box*. Pengujian unit memprioritaskan sasaran pada *source-code* dan sistematika penyusunan *source-code (syntax)*. Proses untuk mengetahui berbagai *source-code* dan *syntax* yang salah *(error)* bisa dilakukan dengan *debugging* pada saat penyusunan *script (coding)*. Proses untuk mengetahui mekanisme dan kinerja dari *source-code* perlu dengan pengujian *white box* dengan membuat *workflow basis path*.

a. *Flowgraph*

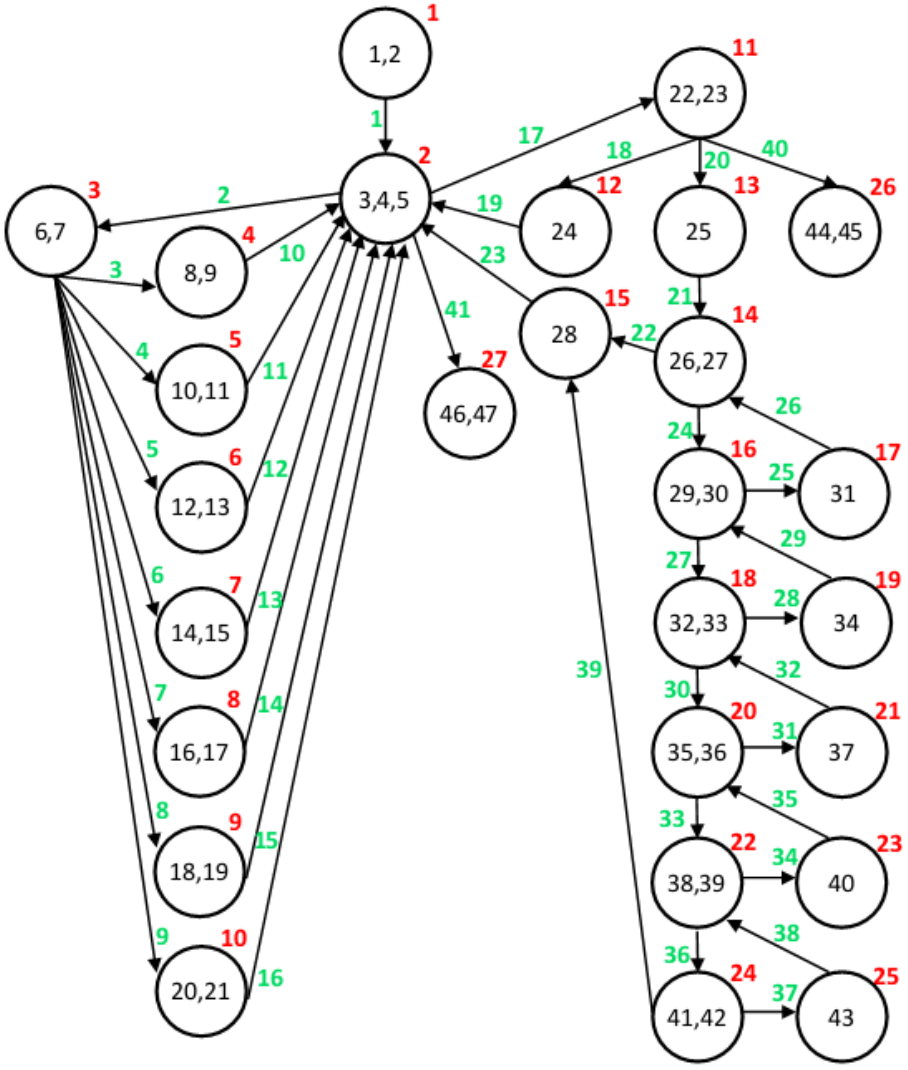
Pengujian unit pada *software* pembelajaran "DigiChip" agar berlangsung dengan sistematis maka terlebih dahulu dilakukan pembuatan *flowgraph*. Sebuah *flowgraph* merupakan gambaran dari laju sebuah algoritma *software*, mulai dari awal sistem menyajikan fitur hingga masuk pada fitur yang lebih kompleks yang telah disediakan *software*. Sebelum pembuatan *flowgraph* terlebih dahulu dibuat sebuah gambaran *flowchart* dari *software* media pembelajaran "DigiChip" pada laporan ini. Sebuah *flowchart* dirancang sebagai grafik pembantu dari *flowgraph* agar lebih mudah dipahami pembaca, sebab *flowgraph* hanya berisi kode angka pada sebuah *basis path* yang tentu lebih lama dipahami pembaca.



Gambar 94. *Flowchart* Media Pembelajaran "DigiChip"

Tahapan selanjutnya adalah pembuatan skema *flowgraph* dari *software* media pembelajaran “DigiChip”. Perancangan *flowgraph* sebagai aktivitas untuk memberikan gambaran lebih detail dari algoritma *software* media pembelajaran “DigiChip” yang telah dibuat. *Flowgraph* terdiri dari pemaparan dua elemen dalam sebuah mekanisme sistem yaitu *edge* dan *node*. *Edge* merupakan istilah untuk penggambaran laju sebuah sistem. *Node* merupakan sebuah fitur yang disajikan oleh rancangan sistem. Perancangan *flowgraph* pada *software* “DigiChip” yang dibuat menampilkan *edge* dengan inisial angka berwarna hijau, dan *node* dengan inisial angka berwarna merah. *Node* memberikan label pada fitur

software “DigiChip” yang disimbolkan dengan gambar berbentuk lingkaran berwarna hitam.



Gambar 95. *Flowgraph* Media Pembelajaran "DigiChip"

Tabel 36. Deskripsi *Flowgraph* "DigiChip"

No	Deskripsi		
1	Mulai	24	Keluar TUTOR
2	*Splashscreen loading system*	25	*Touch* Petunjuk Pengoperasian
3	Halaman *Home*	26	Atur Tampilan
4	*Touch* MENU	27	*Next* Atur Tampilan
5	Tampilan MULAI, TUTOR, KELUAR	28	Keluar Petunjuk Pengoperasian
6	*Touch* MULAI	29	Menampilkan *Scematic*
7	Tampilan Pilihan Simulasi	30	*Next* Menampilkan *Scematic*
8	*Touch* Sistem NOT	31	*Prev* Menampilkan *Scematic*
9	Simulasi Gerbang Logika NOT	32	Pasang Kabel Konektor
10	*Touch* Sistem AND	33	*Next* Pasang Kabel Konektor
11	Simulasi Gerbang Logika AND	34	*Prev* Pasang Kabel Konektor
12	*Touch* Sistem NAND	35	Lepas Kabel Konektor
13	Simulasi Gerbang Logika NAND	36	*Next* Lepas Kabel Konektor
14	*Touch* Sistem OR	37	*Prev* Lepas Kabel Konektor
15	Simulasi Gerbang Logika OR	38	Tombol *Input*
16	*Touch* Sistem NOR	39	*Next* Tombol *Input*
17	Simulasi Gerbang Logika NOR	40	*Prev* Tombol *Input*
18	*Touch* Sistem XOR	41	LED *Output*
19	Simulasi Gerbang Logika XOR	42	*Next* LED *Output*
20	*Touch* Tampilan Penuh	43	*Prev* LED *Output*
21	Keluar Pilihan Simulasi	44	*Touch Truth Tabel*
22	*Touch* TUTOR	45	Tampilan *Truth Tabel*
23	Tampilan Pilihan Tutorial	46	*Touch* KELUAR
		47	Aplikasi Berhenti dan Keluar

b. *Cyclomatic Complexity (CC)*

Proses selanjutnya dalam pengujian unit adalah menghitung terlebih dahulu *Cyclomatic Complexity*. Penghitungan *Cyclomatic Complexity* seperti yang telah dijelaskan pada bab 3 tentang prosedur pengujian *white-box*. *Cyclomatic Complexity* dinotasikan dengan V(G) = *e* – *n* + 2. Notasi *"e"* merupakan jumlah *edge* (laju sistem), dan *"n"* merupakan jumlah *node* (fitur sistem). Nilai dari *Cyclomatic Complexity* digunakan untuk proses selanjutnya yaitu penentuan *independent path*. Perangkat lunak "DigiChip" yang telah dibuat memiliki jumlah *edge*

sebanyak 41, dan jumlah *node* sebanyak 27. Maka *Cyclomatic Complexity (CC)* atau V(G) dapat dihitung sebagai berikut:

$e = 41$; $n = 27$

$V(G) = e - n + 2$

$V(G) = 41 - 27 + 2$

$= \mathbf{16}$

c. *Independent Path*

Pekerjaan berikutnya adalah menentukan *independent path* yang merupakan representasi dari laju sistem ketika diperintah oleh pengguna dari tahap awal hingga akhir. Total jumlah jalur seperti yang telah dihitung pada penghitungan *Cyclomatic Complexity (CC)*, yaitu 16. Penjelasan lebih detail dapat dlihat pada sajian tabel berikut:

Tabel 37. *Independent Path* "DigiChip"

No	*Independent Path*
1	1,2→3,4,5→6,7→8,9
2	1,2→3,4,5→6,7→10,11
3	1,2→3,4,5→6,7→12,13
4	1,2→3,4,5→6,7→14,15
5	1,2→3,4,5→6,7→16,17
6	1,2→3,4,5→6,7→18,19
7	1,2→3,4,5→6,7→20,21
8	1,2→3,4,5→22,23→24
9	1,2→3,4,5→22,23→25→26,27→28
10	1,2→3,4,5→22,23→25→26,27→29,30→31
11	1,2→3,4,5→22,23→25→26,27→32,33→34
12	1,2→3,4,5→22,23→25→26,27→35,36→37
13	1,2→3,4,5→22,23→25→26,27→38,39→40
14	1,2→3,4,5→22,23→25→26,27→41,42→43
15	1,2→3,4,5→22,23→44,45
16	1,2→3,4,5→46,47

d. *Test Case*

Proses selanjutnya adalah pembuatan laju algoritma sistem/program dengan mengacu pada *independent path* yang telah dibuat. Tabel *independent path* diuraikan dalam bentuk berbagai langkah menjalankan sistem *(test case)*. Pembuatan *test case* selanjutnya digunakan dalam tahap pengujian *software* "DigiChip" berupa *integration testing*. Tabel rincian daftar *test case* disajikan dalam **(Lampiran 1)**.

6. *Integration Testing*

Pengujian selanjutnya untuk menyempurnakan hasil dari perancangan *software* yang telah dibuat adalah *integration testing*, yaitu menjalankan algoritma sistem/program dengan mengacu pada daftar *test case* yang telah dibuat. *Integration testing* dilakukan menggunakan metode *black-box*. Proses pengujian *integration* dilakukan ketika semua elemen pada tahap sebelumnya seperti *architectural design*, *user interface*, *script* kode telah terintegrasi, karena itulah disebut dengan pengujian integrasi. Tahapan pengujian *integration* digunakan untuk mengetahui apakah algoritma dan sistematika *software* "DigiChip" yang telah dibuat telah sesuai dengan tahap perencanaan dengan tanpa terjadi *error*. Hasil dari pengujian integrasi digunakan sebagai acuan dalam penentuan aspek *functional suitability* dari standar kualitas *software* berdasarkan ISO/IEC 25010, yang berfokus pada kesesuaian satu kesatuan fungsi untuk dapat melakukan tugas-tugas tertentu

sesuai dengan yang diharapkan. Tabel daftar hasil *integration testing software* media pembelajaran "DigiChip" disajikan dalam **(Lampiran 2)**.

7. *System Testing*

Tahapan pengujian selanjutnya adalah *system testing,* merupakan tahapan proses pengujian *software* yang telah dibuat ketika antar bagian telah tergabung dalam satu kesatuan sistem yang utuh. Pengujian sistem merupakan pengujian keseluruhan kinerja dari produk perangkat lunak yang telah dibuat. Proses ini dilakukan oleh *engineer/programmer* yang bersangkutan. Proses pengujian sistem menggunakan teknik *black-box*, dengan menghitung parameter *Line of Code (LoC) source code* dan *duplication source code*.

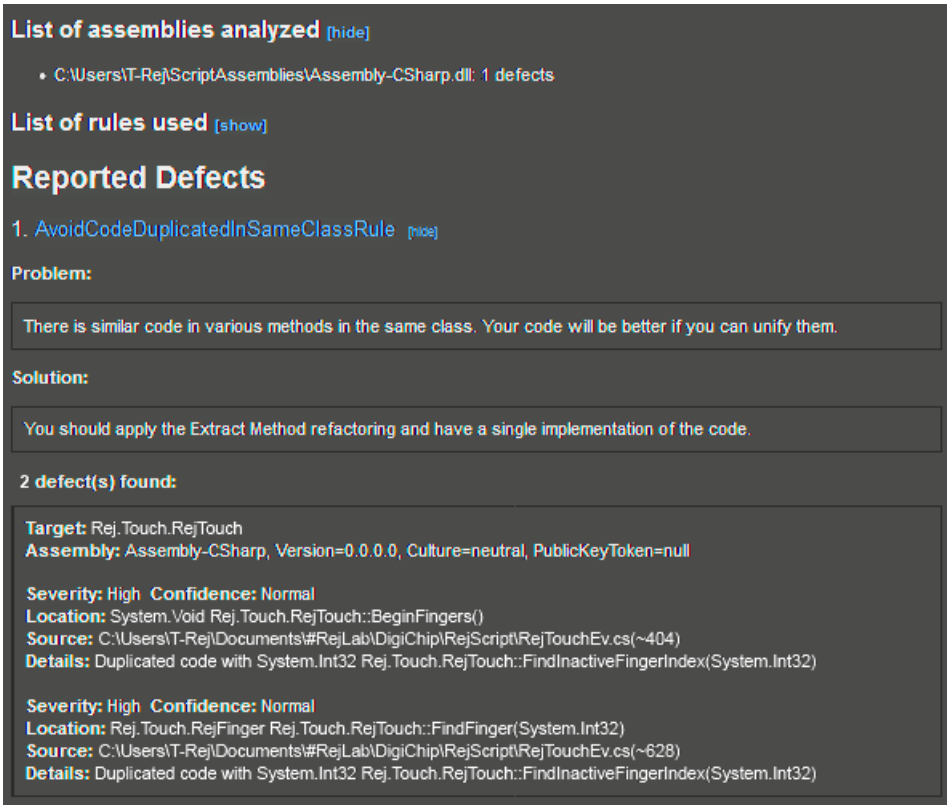
a. *Line of Code (LoC)*

Penghitungan *Line of Code (LoC)* merupakan proses akumulasi dari jumlah *source code* yang digunakan dalam pembuatan *software* media pembelajaran "DigiChip". Proses penghitungan *source code* dilakukan secara manual dengan mengakses satu per satu dari *file script* yang *embedded* pada obyek dalam perangkat lunak *graphics engine* yang digunakan. Daftar hasil rincian dari penghitungan *Line of Code (LoC) source code* pada berbagai *file script* yang digunakan dalam perancangan *software* media pembelajaran "DigiChip" disajikan dalam **(Lampiran 3).**

Hasil penghitungan total jumlah *source code* yang digunakan dalam perancangan *software* pembelajaran "DigiChip" adalah sebanyak **10.541** baris *source code* atau LoC *(Line of Code)*. Semua *source code* pada *file script* ditulis dalam bahasa C# *(Csharp)*.

b. *Duplication Source Code*

Proses pengujian berikutnya adalah penghitungan jumlah *duplication source code* pada semua *file script* yang bermuatan sebanyak 10.541 baris *source code*. Proses penghitungan dan analisis *duplication source code* agar lebih presisi dan tanpa ada kesalahan, dilakukan dengan menggunakan sistem perangkat lunak *analyzer*. Merek dan tipe perangkat lunak *analyzer* yang digunakan satu jenis dengan perangkat lunak yang digunakan dalam proses perancangan *source code*, hal ini untuk meminimalisir terjadinya *error analyze* dan keakuratan yang lemah akibat perbedaan format sistem antar perangkat lunak yang berbeda merek dan tipe. Hasil analisis *source code* duplikasi menunjukkan terdapat 2 baris *source code* dari 10.541 baris *source code* yang telah dibuat. Duplikasi *source code* 2 baris tersebut berada dalam *file script* bernama RejTouchEv.cs, yang merupakan ekstensi *file* berbahasa C#. Berikut merupakan uraian detail dari hasil analisis *duplication source code* pada semua *file script software* media pembelajaran "DigiChip" yang telah dibuat.



Gambar 96. *Screenshot* Hasil Analisis *Duplication Source Code* "DigiChip"

c. Uji Instalasi

Proses pengujian selanjutnya adalah aktivitas instalasi produk *software* "DigiChip" yang telah dibuat, pada berbagai jenis konfigurasi *hardware* perangkat *mobile*. Proses uji instalasi ini pada dasarnya lebih kepada analisa operasional *software* yang telah dibuat pada berbagai varian kernel atau versi *operating system* (OS) android. Parameter rasio layar *(screen)* tidak menjadi hal yang prioritas, karena *software* "DigiChip" yang telah dibuat dirancang dengan fitur kemampuan untuk *auto-extend* pada berbagai rasio layar. Karakteristik *software* "DigiChip" yang dibuat merupakan jenis *software* simulasi 3D *realtime*, yang tentu

membutuhkan kemampuan perangkat *mobile* yang memiliki kemampuan grafis yang mumpuni. Produk *software* "DigiChip" yang telah dibuat memiliki karakter memacu kinerja *hardware* grafis secara terus menerus selama aplikasi *"on"*, dan hal ini akan berhenti ketika keluar aplikasi. Karakter *software* yang demikian tentu membutuhkan kinerja *hardware* yang berbeda dibandingkan dengan produk *software* lain yang memiliki berkarakter kinerja yang statis. Sebab itu pengujian pada tahap uji instalasi ini menitikberatkan pada aspek kemampuan grafis perangkat *mobile* untuk menjalankan *software* media pembelajaran Teknik Digital "DigiChip".

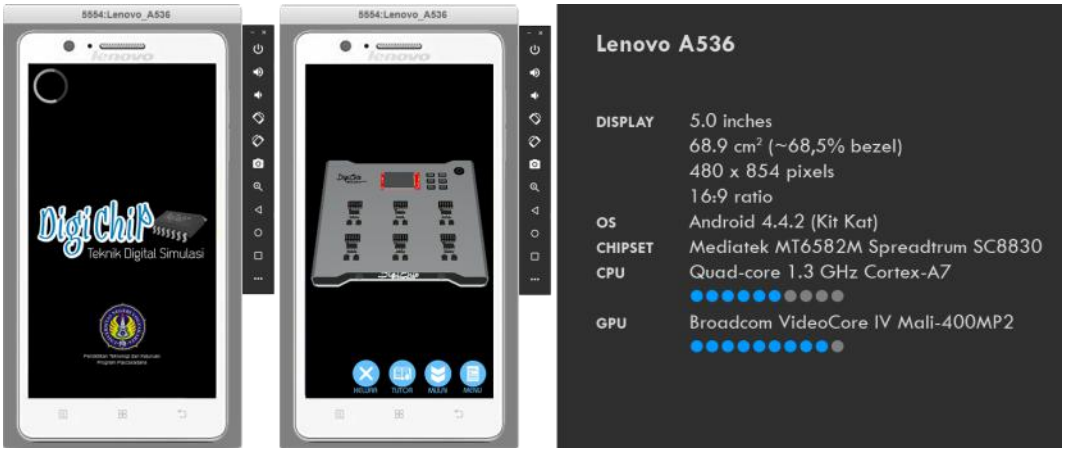
Pengujian instalasi tidak hanya berupa pengujian tingkat *compatibility* pada *software* yang telah dibuat, seperti yang disediakan oleh produk pengujian *compatibility* dan *portability* Google Play. Tahap pengujian instalasi ini benar-benar harus dapat mengungkap kemampuan grafis perangkat *mobile* untuk menjalankan *software* "DigiChip" secara *graphics run-test*. Metode yang digunakan adalah dengan melalui teknik *running test* yang dimiliki oleh perangkat *emulator* dan *simulator device*, seperti berupa AVD *(Android Virtual Device)*. Kelebihan lain dari penggunaan AVD adalah, mampu memuat berbagai konfigurasi dan spesifikasi perangkat *mobile* yang tidak didapati dalam penelitian secara langsung pada responden *(user)*. Berikut hasil uji instalasi dari berbagai konfigurasi perangkat *mobile* melalui *simulator/emulator*:

Tabel 38. Pengujian *Install* dan *Running* "DigiChip"

No	Versi OS	Tipe	*Install*	*Running*
1	4.2 Jelly Bean	Samsung Galaxy Grand Neo	berhasil	berjalan lancar
2	4.4 Kit Kat	Lenovo A536	berhasil	berjalan lancar
3	5.0 Lolipop	Asus ZenFone 2 Laser (ZE500KG)	berhasil	berjalan lancar
4	6.0 Marshmallow	Asus ZenFone 3 (Z552KL)	berhasil	berjalan lancar
5	7.0 Nougat	Xiaomi Redmi Note 4 Pro	berhasil	berjalan lancar
6	8.0 Oreo	Vivo Y71	berhasil	berjalan lancar

Hasil pengujian yang telah dilakukan pada berbagai konfigurasi perangkat *mobile*, tidak ditemukan adanya *error* pada proses *install* maupun pada saat pengoperasian *software* "DigiChip" secara terus menerus. Pengujian pada berbagai perangkat *mobile* yang memiliki konfigurasi *hardware* CPU *(central processing unit)* dan GPU *(graphics processing unit)* paling minim sekalipun tidak ditemukan adanya kemacetan pada sistem maupun *error* pada saat *running* berlangsung. Berikut keterangan lebih detail mengenai berbagai konfigurasi dan spesifikasi perangkat *mobile* yang berperan sebagai pengguna akhir dari *software* media pembelajaran Teknik Digital "DigiChip".

Gambar 97. Pengujian pada OS 4.2 Jelly Bean



Gambar 98. Pengujian pada OS 4.4 Kit Kat

Gambar 99. Pengujian pada OS 5.0 Lolipop

Gambar 100. Pengujian pada OS 6.0 Marshmallow

Gambar 101. Pengujian pada OS 7.0 Nougat

Gambar 102. Pengujian pada OS 8.0 Oreo

8. *Acceptance Testing*

Tahap pengujian paling akhir dari serangkaian tahapan proses uji produk *software* "DigiChip" adalah *acceptance testing*. Proses pengujian *acceptance* melibatkan hasil *assessment* dari para ahli, dalam hal ini adalah ahli materi dan ahli media. Proses selanjutnya merupakan penilaian dari para responden sebagai pengguna akhir dari produk *software* media pembelajaran Teknik Digital "DigiChip". Sebelum melakukan proses pengujian pada pengguna akhir dalam hal ini adalah siswa, terlebih dahulu dilakukan penilaian produk *software* "DigiChip" *(expert judgement)* dari para ahli. Penilaian yang dilakukan dari ahli materi merupakan aspek untuk menentukan kualitas produk *software* "DigiChip" dari parameter kesesuaian isi materi dengan materi ajar yang didesain dari sekolah. Penilaian dari ahli media merupakan aspek penilaian secara umum dari kualitas produk *software* "DigiChip" yang telah dibuat, lebih khusus tentang sudut pandang sebuah *software* untuk layak disebut sebagai alat belajar atau media dalam proses pembelajaran. Berikut dokumen dari para ahli sebagai penguji aspek *acceptance* ahli media dan ahli materi;

Tabel 39. Data Ahli Media dan Ahli Materi

	Ahli Media	**Ahli Materi**
Nama	Dr. Samsul Hadi, M.Pd., M.T	Muhammad Agung Wibowo, S.Si
Profesi	Dosen	Guru
Bidang Keahlian	Multimedia	Mekatronika

a. Pengujian *Alpha* untuk Ahli Media

Tabel 40. Hasil Penilaian Ahli Media

Aspek	Nomor Pernyataan	Nilai Ahli	Nilai Maksimal
Kesesuaian Warna	1	5	20
	2	5	
	3	4	
	4	4	
Kesesuaian Gambar	5	5	15
	6	5	
	7	4	
Kesesuaian Huruf	8	5	10
	9	5	
Kesesuaian Angka	10	5	10
	11	5	
Kesesuaian Animasi	12	3	15
	13	4	
	14	4	
Nilai Total		**63**	**70**

Hasil dari penilaian dan validasi ahli media dapat pelajari pada tabel. Instrumen penilaian berisi 14 item pernyataan yang digunakan untuk mengukur persepsi penilai dalam 5 skala penilaian. Nilai tertinggi (maksimal) tiap item pernyataan tentu memiliki nilai 5 dan sebaliknya nilai terendah adalah 1, sehingga jumlah nilai total dari 14 pernyataan adalah 70.

b. Pengujian *Alpha* untuk Ahli Materi

Tabel 41. Hasil Penilaian Ahli Materi

Aspek	Nomor Pernyataan	Nilai Ahli	Nilai Maksimal
Simulasi Aljabar Boolean	1	1	6
	2	1	
	3	1	
	4	1	
	5	1	
	6	1	
Simbolik *Logic Gate*	7	1	7
	8	1	
	9	1	
	10	1	
	11	1	
	12	1	
	13	1	
Skematik IC	14	1	6
	15	1	
	16	1	
	17	1	
	18	1	
	19	1	
Nilai Total		**19**	**19**

Proses akhir dalam tahapan pengujian *acceptance* adalah memperoleh informasi berupa data penilaian yang diperoleh dari *user* sasaran atau pengguna akhir dari produk *software* pembelajaran "DigiChip" ini, yang telah melewati serangkaian tahapan pengujian *software*. Pengguna produk *software* "DigiChip" ini adalah sebagai responden dalam kegiatan penelitian ilmiah

yang dilakukan. Para responden yang akan memberikan respon dan masukkan dari produk kegiatan penelitian ini adalah para siswa-siswi dalam sebuah sekolah kejuruan yang berstatus pelajar Program Keahlian Teknik Mekatronika. Populasi responden berjumlah 59 pelajar yang terdiri dari kelas XI TM-A dan kelas XI TM-B.

B. Hasil Pengujian Standar ISO/IEC 25010

Hasil dari serangkaian pengujian berdasarkan kaidah *v-model* yang meliputi pengujian *unit*, *integration*, *system*, dan *acceptance*, selanjutnya akan dikonversikan pada aspek pengujian ISO 25010. Pengujian kaidah ISO 25010 meliputi berbagai aspek, diantaranya *functional suitability*, *maintainability*, *portability*, *usability*.

1. Pengujian *Functional Suitability*

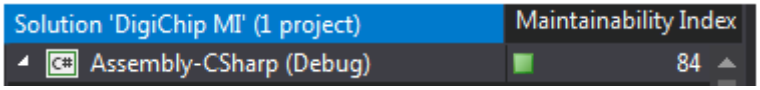
Menentukan kualitas aspek *functional suitability* pada perangkat lunak "DigiChip" yang telah dibuat, dilakukan dengan prosedur *integration testing*. Pengujian terintegrasi diakukan dengan menggunakan serangkaian daftar *test case* yang berisi seluruh fungsi fitur yang dimiliki *software* "DigiChip". Sehingga secara keseluruhan sistem dapat diketahui apakah beroperasi dengan baik ataukah tidak sesuai rencana dan mengalami kegagalan sistem. Daftar hasil dari pengujian masing-masing fitur perangkat lunak "DigiChip" untuk parameter aspek *functional suitability* disajikan dalam **(Lampiran 4).**

2. Pengujian *Maintainability*

Kualitas perangkat lunak yang telah dibuat selanjutnya diuji pada aspek *maintainability* sesuai kaidah pengujian ISO 25010. Aspek *maintainability* pada *software* "DigiChip" meliputi jumlah *Line of Code (LoC) source code* dan jumlah *duplication source code*. Hasil penghitungan menunjukkan terdapat total jumlah *source code* sebanyak 10.541 baris *source code* atau LoC *(Line of Code)* yang digunakan dalam perancangan *software* pembelajaran "DigiChip". Semua *source code* pada *file script* ditulis dalam bahasa C# *(Csharp)*. Hasil analisis *source code* duplikasi menunjukkan terdapat 2 baris *source code* dari 10.541 baris *source code* yang telah dibuat dalam proses perancangan *software* "DigiChip". Duplikasi *source code* 2 baris berada dalam *file script* bernama RejTouchEv.cs, yang merupakan ekstensi *file* berbahasa C#.

Aspek *maintainability* dinyatakan dengan penghitungan nilai *Maintainability Index (MI)* pada semua *file source code* yang digunakan dalam pembuatan media pembelajaran "DigiChip". Penghitungan nilai *Maintainability Index* menggunakan *software* yang digunakan untuk menyusun *script* kode bahasa C# pada proses pembuatan media pembelajaran "DigiChip", yang memiliki fitur *analyzer*. Hasil penghitungan nilai MI diperoleh spesifikasi sebagai berikut;

Tabel 42. Total Nilai *Maintainability Index (MI)*



Tabel hasil penghitungan *Maintainability Index* secara spesifik pada semua *source code* dapat dilihat pada **(Lampiran 9)**.

3. Pengujian *Portability*

Aspek pengujian ISO 25010 selanjutnya adalah aspek *portability*, merupakan aspek uji instalasi yang pada dasarnya lebih kepada analisa operasional *software* "DigiChip" yang telah dibuat pada berbagai varian kernel atau versi *operating system* (OS) android. Sasaran analisa yang dilakukan dititikberatkan pada kemampuan *hardware* grafis dari perangkat *smartphone* yang akan ditanami *software* "DigiChip". Karakteristik *software* "DigiChip" yang dibuat merupakan jenis *software* simulasi *3D realtime*, yang tentu membutuhkan kemampuan perangkat *smartphone* yang memiliki kemampuan grafis yang mumpuni. Produk *software* "DigiChip" yang telah dibuat memiliki karakter memacu kinerja *hardware* grafis secara terus menerus selama aplikasi *"on"*, dan hal ini akan berhenti ketika keluar aplikasi. Karakter *software* yang demikian tentu membutuhkan kinerja *hardware* yang berbeda dibandingkan dengan produk *software* lain yang memiliki berkarakter kinerja yang statis. Tahap pengujian instalasi ini benar-benar harus dapat mengungkap kemampuan grafis perangkat *smartphone* untuk menjalankan *software* "DigiChip" secara *graphics run-test*.

Metode yang digunakan pada pengujian aspek *portability* adalah dengan melalui teknik *running test* yang dimiliki oleh perangkat *emulator* dan *simulator device*, seperti berupa AVD *(Android Virtual Device)*.

Jenis konfigurasi perangkat *smartphone* serta varian kernel dan OS android yang diuji berdasarkan pada data yang bersumber dari responden (siswa). Hasil pengujian yang telah dilakukan pada berbagai konfigurasi *smartphone*, tidak ditemukan adanya *error* pada proses *install* maupun pada saat pengoperasian *software* "DigiChip" secara terus menerus. Pengujian pada berbagai *smartphone* yang memiliki konfigurasi *hardware* CPU *(central processing unit)* dan GPU *(graphics processing unit)* paling minim sekalipun tidak ditemukan adanya kemacetan pada sistem maupun *error* pada saat *running* berlangsung.

Tabel 43. Pengujian *Portability* "DigiChip"

No	Versi OS	Tipe	*Install*	*Running*	*Nilai Hasil*	*Nilai Maks*
1	4.2 Jelly Bean	Samsung Galaxy G- Neo	1	1	2	2
2	4.4 Kit Kat	Lenovo A536	1	1	2	2
3	5.0 Lolipop	Asus ZenFone 2 Laser	1	1	2	2
4	6.0 Marshmallow	Asus ZenFone 3	1	1	2	2
5	7.0 Nougat	Xiaomi Redmi Note 4 Pro	1	1	2	2
6	8.0 Oreo	Vivo Y71	1	1	2	2
Total Nilai					**12**	**12**

4. Pengujian *Usability*

Pengujian pada aspek *usability* merupakan pengguna *(user)* dari *software* media pembelajaran "DigiChip". Pengguna merupakan siswa kelas XI SMK Teknik Mekatronika SMTI Yogyakarta, berjumlah 59 siswa.

Pengujian aspek *usability* dilakukan dengan membagikan *software* pembelajaran "DigiChip" yang telah dibuat, disertai angket kuesioner *usability* berjumlah 28 butir pernyataan. Data dari hasil kuesioner *usability* disajikan pada tabel berikut ini :

Tabel 44. Pengujian *Usability* "DigiChip"

Responden	Nilai Total	Responden	Nilai Total
1	108	31	106
2	97	32	113
3	115	33	112
4	128	34	125
5	107	35	124
6	128	36	123
7	130	37	127
8	124	38	123
9	123	39	125
10	126	40	111
11	114	41	129
12	106	42	128
13	126	43	125
14	123	44	125
15	127	45	126
16	111	46	128
17	103	47	126
18	123	48	126
19	127	49	129
20	108	50	127
21	124	51	131
22	115	52	126
23	104	53	130
24	119	54	129
25	117	55	127
26	107	56	130
27	125	57	127
28	113	58	126
29	121	59	124
30	112	Jumlah	7.119

C. Anlisis Data

1. Analisis Data Hasil *Unit Testing*

Pengujian *unit* terdiri dari 47 eksekusi fitur yang disediakan pada *software* media pembelajaran "DigiChip". Item fitur yang beroperasi dengan baik dicatat kedalam daftar uji fitur dengan diberi tanda (✓) *cek list*. Hasil *unit testing* pada *software* pembelajaran yang telah dibuat adalah sebagai berikut;

Tabel 45. Pengujian *Unit* "DigiChip"

No	Deskripsi	Tes			
1	Mulai	✓	24	Keluar TUTOR	✓
2	*Splashscreen loading system*	✓	25	*Touch* Petunjuk Pengoperasian	✓
3	Halaman *Home*	✓	26	Atur Tampilan	✓
4	*Touch* MENU	✓	27	*Next* Atur Tampilan	✓
5	Tampilan MULAI, TUTOR, KELUAR	✓	28	Keluar Petunjuk Pengoperasian	✓
6	*Touch* MULAI	✓	29	Menampilkan *Scematic*	✓
7	Tampilan Pilihan Simulasi	✓	30	*Next* Menampilkan *Scematic*	✓
8	*Touch* Sistem NOT	✓	31	*Prev* Menampilkan *Scematic*	✓
9	Simulasi Gerbang Logika NOT	✓	32	Pasang Kabel Konektor	✓
10	*Touch* Sistem AND	✓	33	*Next* Pasang Kabel Konektor	✓
11	Simulasi Gerbang Logika AND	✓	34	*Prev* Pasang Kabel Konektor	✓
12	*Touch* Sistem NAND	✓	35	Lepas Kabel Konektor	✓
13	Simulasi Gerbang Logika NAND	✓	36	*Next* Lepas Kabel Konektor	✓
14	*Touch* Sistem OR	✓	37	*Prev* Lepas Kabel Konektor	✓
15	Simulasi Gerbang Logika OR	✓	38	Tombol *Input*	✓
16	*Touch* Sistem NOR	✓	39	*Next* Tombol *Input*	✓
17	Simulasi Gerbang Logika NOR	✓	40	*Prev* Tombol *Input*	✓
18	*Touch* Sistem XOR	✓	41	LED *Output*	✓
19	Simulasi Gerbang Logika XOR	✓	42	*Next* LED *Output*	✓
20	*Touch* Tampilan Penuh	✓	43	*Prev* LED *Output*	✓
21	Keluar Pilihan Simulasi	✓	44	*Touch Truth Tabel*	✓
22	*Touch* TUTOR	✓	45	Tampilan *Truth Tabel*	✓
23	Tampilan Pilihan Tutorial	✓	46	*Touch* KELUAR	✓
			47	Aplikasi Berhenti dan Keluar	✓

Data yang telah diperoleh dari tahapan *unit testing software* "DigiChip" selanjutnya dianalisis dengan berdasarkan pada rumus prosentase kelayakan yang telah dijelaskan pada bahasan sebelumnya.

$$Persentase\ kelayakan\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

$$= \frac{47}{47} \text{ x } 100\%$$

$$= 100\%$$

Hasil penghitungan yang telah diperoleh selanjutnya dikonversikan pada kategori kelayakan yang telah dijelaskan pada bab sebelumnya. Pengujian *unit* 100% termasuk dalam kategori *"Sangat Layak"*.

2. Analisis Data Hasil *Integration Testing*

Pengujian *integration* berupa uji pengoperasian berbagai fitur melalui daftar 16 *test case* dengan metode *black-box*. Pengujian *integration* merupakan pengujian untuk mengetahui aspek *functional suitability* pada standar ISO 25010. Analisis penghitungan yang didapat adalah sebagai berikut;

$$Operasional\ test\ case\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

$$= \frac{16}{16} \text{ x } 100\%$$

$$= 100\%$$

Hasil penghitungan yang telah diperoleh selanjutnya dikonversikan pada tabel kategori kelayakan. Pengujian *integration* 100% sudah dipastikan termasuk dalam kategori *"Sangat Layak"*. Pengertian lain dapat dimaknai bahwa nominal 100% menjelaskan bahwa keseluruhan fitur pada *software* pembelajaran "DigiChip" yang telah dibuat, beroperasi dengan baik.

3. Analisis Data Hasil *System Testing*

Pengujian *system* merupakan pengujian yang dilakukan pada *software* media pembelajaran "DigiChip" untuk mengetahui aspek *maintainability* dan aspek *portability* pada standar ISO 25010.

a. Analisis Data Hasil *Maintainability Testing*

Nilai *Maintainability Index (MI)* yang telah diketahui yaitu 84 dengan tanda simbol kotak berwana hijau, jika diinterpretasikan merujuk pada **(Tabel 21)** maka artinya seluruh *script code* pada *software* "DigiChip" memiliki kategori perawatan yang mudah. Secara teori seluruh *script code* C# "DigiChip" kedepannya akan mudah untuk dilakukan pengembangan dan modifikasi ke tahapan inovasi yang diinginkan.

1). *Cyclomatic Complexity (CC)*

Hasil penghitungan dari *Cyclomatic Complexity (CC)* adalah 16. Berdasarkan tabel hubungan antara nilai *Cyclomatic Complexity (CC)* dengan faktor resiko *(risk)*, apabila nilai *Cyclomatic Complexity* 16 maka tingkat resiko dalam memahami, menguji dan memelihara

software termasuk dalam kategori sedang *(moderate risk)*. Berikut ini merupakan tabel hubungan antara *Cyclomatic Complexity (CC)* dengan evaluasi resiko pada pemeliharaan *(maintaining)* sebuah *software* (Heitlager, 2007:6).

Tabel 46. Evaluasi Resiko *Cyclomatic Complexity (CC)*

CC	Risk evaluation
1-10	simple, without much risk
11-20	more complex, moderate risk
21-50	complex, high risk
> 50	untestable, very high risk

Hasil penghitungan CC dan evaluasi resiko pada pengujian *software* media pembelajaran "DigiChip", jika diinterpretasikan kedalam klasifikasi kelayakan, maka diperoleh hasil kategori *"layak"*.

Tabel 47. Interpretasi Evaluasi Resiko *Cyclomatic Complexity (CC)*

CC	Risk evaluation	Interpretasi
1-10	*simple, without much risk*	Sangat Layak
11-20	*more complex, moderate risk*	Layak
20-50	*complex, high risk*	Tidak Layak
> 50	*untesable, very high risk*	Sangat Tidak Layak

2). *Line of Code (LoC)*

Hasil penghitungan dari *Line of Code (LoC)* diketahui terdapat 10.541 baris *source code* yang tertanam pada 116 *file script code* yang digunakan untuk membuat *software* pembelajaran "DigiChip". Nilai total dari hasil penghitungan *Line of Code* digunakan sebagai acuan

dalam menghitung dan menginterpretasikan persentase *Duplication Source Code.* Sehingga dapat diketahui tingkat efektivitas *script* dalam proses *running program* maupun opeasional *software* pembelajaran "DigiChip".

3). *Duplication Source Code*

Hasil penghitungan dari *Duplication Source Code* diketahui terdapat 2 baris *source code* yang tertanam pada *software* pembelajaran "DigiChip". Nilai total dari hasil penghitungan *Line of Code (LoC)* adalah sebesar 10.541 baris *source code*. Nilai persentase *Duplication Source Code* maka sebagai berikut;

$$\begin{aligned} &Duplication\ \ Source\ Code\ (\%) \\ &= \frac{\text{Jumlah Baris } Duplication\ Source\ Code}{\text{Jumlah Total } Line\ of\ Code\ (LoC)} \text{ x } 100\% \\ &= \frac{2}{10.541} \text{ x } 100\% \\ &= 0{,}018\% \end{aligned}$$

Persentase hasil penghitungan *Duplication Source Code* selanjutnya dikonversikan ke tabel klasifikasi *duplication* **(Tabel 12)**. Berdasarkan tabel *duplication* dapat dimaknai bahwa aspek *Duplication Source Code* pada *software* pembelajaran "DigiChip" masuk dalam kategori sangat kecil (*"very small"*).

b. Analisis Data Hasil *Portability Testing*

Pengujian aspek *portability* dilakukan menggunakan proses instalasi pada berbagai konfigurasi perangkat *smartphone*. Uji instalasi dilakukan pada 6 varian *operating system* (OS) android. Berdasarkan tabel hasil uji *portability* maka dapat dihitung dengan rumus sebagai berikut;

$$Portability\ Testing\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

$$= \frac{12}{12} \text{ x } 100\%$$

$$= 100\%$$

Hasil penghitungan yang telah diperoleh selanjutnya dikonversikan pada kategori kelayakan yang telah dijelaskan pada bab sebelumnya. Pengujian *portability* 100% termasuk dalam kategori *"Sangat Layak"*.

4. Analisis Data Hasil *Acceptance Testing*

Pengujian *acceptence* merupakan proses pengujian yang dilakukan dengan melibatkan *expert judgement* dan *user*. Tahapan *expert judgement* dilakukan pada ahli media dan ahli materi. Tahapan pengujian *usability* dilakukan pada *user*, yang dalam hal ini adalah siswa.

a. Analisis Data Hasil *Alpha Testing untuk Ahli Media*

Hasil pengujian pada ahli media diperoleh nilai 63. Selanjutnya dihitung menjadi persentase kelayakan;

$$Persentase\ Kelayakan\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

$$= \frac{63}{70} \text{ x } 100\%$$

$$= 90\%$$

Hasil penghitungan persentase kelayakan dari uji ahli media selanjutnya diinterpretasikan berdasarkan tabel kelayakan **(Tabel 20)**, maka diperoleh klasifikasi *"Sangat Layak"*.

b. Analisis Data Hasil *Alpha Testing untuk Ahli Materi*

Hasil pengujian pada ahli materi diperoleh nilai 19. Selanjutnya dihitung menjadi persentase kelayakan;

$$Persentase\ Kelayakan\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

$$= \frac{19}{19} \text{ x } 100\%$$

$$= 100\%$$

Hasil penghitungan persentase kelayakan dari uji ahli materi selanjutnya diinterpretasikan berdasarkan tabel kelayakan **(Tabel 20)**, maka diperoleh klasifikasi *"Sangat Layak"*.

c. Analisis Data Hasil *Usability Testing (Beta Testing)*

Penghitungan persentase kelayakan pada aspek *usability testing* adalah sebagai berikut;

$$Persentase\ Kelayakan\ (\%) = \frac{\text{Jumlah skor yang diperoleh}}{\text{Jumlah skor tertinggi}} \text{ x } 100\%$$

$$= \frac{7119}{8260} \text{ x } 100\%$$

$$= 86{,}18\%$$

Hasil penghitungan persentase kelayakan dari uji *usability (user)* selanjutnya diinterpretasikan dengan merujuk pada tabel kelayakan **(Tabel 20)**, maka diperoleh klasifikasi hasil *"Sangat Layak"*.

Analisis parameter reliabilitas pada aspek pengujian *usability* dilakukan dengan menggunakan *software* statistik penghitung *Alpha Cronbach (α)*. Hasil yang diperoleh adalah sebagai berikut;

Tabel 48. Hasil Penghitungan *Alpha Cronbach*

Case Processing Summary

		N	%
Cases	Valid	59	100.0
	Excluded[a]	0	.0
	Total	59	100.0

a. Listwise deletion based on all variables in the procedure.

Reliability Statistics

Cronbach's Alpha	N of Items
.841	28

Alpha Cronbach 0,841 selanjutnya diinterpretasikan melalui tabel deskripsi berikut ini (George, 2016:240) ;

Tabel 49. *Alpha Value Description* IBM SPSS

$\alpha > .9$—excellent
$\alpha > .8$—good
$\alpha > .7$—acceptable
$\alpha > .6$—questionable
$\alpha > .5$—poor
$\alpha < .5$—unacceptable

Hasil penghitungan *alpha* yang telah diperoleh (0,841) maka dapat diartikan dengan klasifikasi *"good"*.

BAB V
SIMPULAN DAN SARAN

A. Simpulan

Hasil dari serangkaian penelitian, perancangan dan pengujian produk media pembelajaran "DigiChip" yang telah dilakukan. Maka dapat diuraikan kesimpulan sebagai berikut;

1. Perangkat lunak *(software)* "DigiChip" sebagai sarana pembelajaran Mata Pelajaran Teknik Digital dikembangkan sebagai media *virtual learning* dan *mobile learning* telah dibuat untuk perangkat berbasis *mobile* komputer, melalui kaidah pengembangan dan rekayasa perangkat lunak *(software engineering)* meliputi *Requirement modeling, Architectural design, Component design, Code generation, Unit testing, Integration testing, System testing, dan Acceptance testing*. Tahapan pengujian selanjutnya pada *software* yang telah dibuat adalah menggunakan prosedur *Software Quality Assurance (SQA)* dengan menggunakan standar pengujian perangkat lunak yang dikembangkan oleh *International Organization for Standardization and International Electrotechnical Commision*, yaitu ISO/IEC 25010. Tahap *requirement modeling* dilakukan dengan penggalian informasi melalui observasi aktivitas pembelajaran dan wawancara. Tahap *architectural design* melalui pembuatan konsep perangkat lunak seperti apa yang bisa memecahkan masalah yang diteliti. Tahap *component design* merupakan aktivitas pembuatan semua UI *(user interface)*, obyek 2D dan 3D, visual efek, pewarnaan material, sistem pengaturan cahaya dan masih banyak lagi.

Tahap *unit testing* dilakukan dengan pembuatan *basis-path*, yang terdiri dari menguraikan *Flowgraph*, *Cyclomatic Complexity* (CC), *Independent Path* dari *software* "DigiChip". Tahap *integration testing* dilakukan dengan menjalankan semua daftar fungsi fitur "DigiChip" (*test case*, *run test*). Tahap *system testing* dilakukan dengan uji instalasi pada berbagai konfigurasi *hardware* perangkat *mobile* komputer dan berbagai versi kernel sistem operasi Android. Tahap *acceptance testing* merupakan pengujian perangkat lunak media pembelajaran "DigiChip" yang dilakukan oleh pihak ahli media, ahli materi dan pihak pengguna.

2. Parameter pengujian *software* "DigiChip" menggunakan ISO/IEC 25010 terdiri dari berbagai aspek pengujian, diantaranya *Functional suitability, Maintainability, Portability,* dan *Usability.* Hasil dari serangkaian proses pengujian dapat disimpulkan bahwa pada aspek *functional suitability* menunjukkan persentase kelayakan 100% semua fitur beroperasi dengan baik, dengan kategori tabel kelayakan dalam klasifikasi *"Sangat Layak"*. Hasil pengujian pada aspek *maintainability* menunjukkan nilai *Maintainability Index (MI)* sebesar 84, yang artinya dalam kategori "*Perawatan yang mudah"*. Hasil pengujian *portability* menunjukkan persentase 100% *software* pembelajaran "DigiChip" mampu beroperasi pada berbagai varian OS android, sesuai tabel kelayakan termasuk dalam kategori *"Sangat Layak"*. Hasil pengujian *usability* menunjukkan persentase 86,18% kategori *"Sangat Layak"*, dengan nilai reliabilitas sebesar 0,841 termasuk dalam kategori *"Baik"*. Tahapan pengujian pada aspek ahli media

menunjukkan persentase 90% dalam kategori *"Sangat Layak"*. Hasil pengujian pada aspek ahli materi menunjukkan persentase 100% dalam kategori *"Sangat Layak"*.

B. Saran

Perancangan *software* pembelajaran "DigiChip" menurut peneliti masih dirasa banyak kekurangan. Konsep rencana awal peneliti adalah merancang *software* pembelajaran "DigiChip" dalam bentuk teknologi VR *(Virtual Reality)*, dengan inovasi fitur algoritma *imaging* (penginderaan) dan *hand-motion catching* menggunakan keilmuan *image processing* yang mengandalkan kamera *onboard* (yang tersedia pada *smartphone*). Serta mengandalkan *hardware* sensor *gyro* bawaan *smartphone* sebagai algoritma *aiming* atau *look viewing* dalam aktivitas penglihatan obyek virtual didalamnya, sehingga memiliki nilai ekonomis karena tidak memerlukan *hardware controller* tambahan. Faktor terbatasnya masa studi peneliti, maka penelitian ini terhenti cukup sampai disini.

DAFTAR PUSTAKA

Acciani, Giuseppe., Fornarelli, Girolamo., & Giaquinto, Antonio. (2010). *A Fuzzy Method for Global Quality Index Evaluation of Solder Joints in Surface Mount Technology*. London: Jurnal IEEE Transactions on Industrial Informatics, Vol. 7, No. 1.

Agarwal, B.B., Tayal, S.P., & Gupta, M. (2010). *Software Engineering & Testing - An Introduction*. London: Jones and Bartlett Publishers.

Ainley, Mary., & Armatas, Christine. (2006). *The International Handbook of Virtual Learning Environments Volume I; Motivational Perspectives on Students' Responses to Learning in Virtual Learning Environments*. *The International Handbook of Virtual Learning Environments*. Dordrecht: Springer.

Ally, Mohamed. (2009). *Mobile Learning - Transforming the Delivery of Education & Training*. Edmonton: AU Press.

Arikunto, Suharsimi. (2013). *Prosedur Penelitian : Suatu Pendekatan Praktik*. Jakarta: Rineka Cipta.

Ayers, John E. (2005). *Digital Integrated Circuits - Analysis and Design*. Florida: CRC Press.

Balaji, S. (2012). *WATEERFALL vs V-MODEL vs AGILE: A Comparative Study On SDLC*. Computer Science Dept., Gulf College Muscat, Sultanate of Oman. International Journal of Information Technology and Business Management. 29th June 2012. Vol.2 No.1 ©JITBM & ARF. ISSN 2304-0777

Bilgin, Can. (2016). *Mastering Cross-Platform Development with Xamarin. Master the skills required to steer cross-platform applications from drawing board to app store(s) using Xamarin*. Birmingham: Packt Publishing.

Bishop, Robert. (2006). *Mechatronics - An Introduction*. Florida: CRC Press - Taylor & Francis Group.

Boehm, Diane., & Aniola-Jedrzejek, Lilianna. (2006). *Teaching and Learning with Virtual Teams; Seven Principles of Good Practice for Virtual International Collaboration. Teaching and Learning with Virtual Teams*. Hershey: Idea Group.

Caladine, Richard. (2008). *Enhancing E-Learning with Media-Rich Content and Interactions*. Pennsylvania: Information Science Publishing.

Cash, Milton. (2012). *Advance Software Engineering.* Delhi: Learning Press.

Cechich, Alejandra., & Piattini, Mario. (2003). *Component-Based Software Quality - Methods and Techniques.* German: Springer.

Chang, Chun-Shi., Bostjancic, Dave V. & Williams, Michael D. (2010). *Systems and Virtualization Management - Standards and the Cloud; Availability Management in a Virtualized World.* Wuhan: Springer.

Chemuturi, Murali. (2011). *Mastering Software Quality Assurance - Best Practices, Tools and Techniques for Software Developers*. Florida: J.Ross Publishing.

Cinar, Onur. (2015). *Android Quick APIs Reference*. California: Apress.

Crussell, Jonathan., Gibler, Clint., & Chen, Hao. (2012). *Attack of the Clones Detecting Cloned Applications on Android Market*. Jurnal Springer-Verlag Berlin Heidelberg.

Darmawan, Deni. (2012). *Inovasi Pendidikan – Pendekatan Praktik Teknologi Multimedia dan Pembelajaran Online*. Bandung: Remaja Rosdakarya Offset.

Daryanto. (2013). *Media Pembelajaran*. Yogyakarta: Gava Media.

David, Matthew. (2011). *Flash Mobile Developing Android and iOS Application*. Oxford: Elsevier.

Deek, Fadi P., McHugh, James A.M., & Eljabiri, Osama M. (2005). *Strategic Software Engineering - An Interdisciplinary Approach*. Florida: Auerbach Publications.

Del Sole, Alessandro. (2017). *Beginning Visual Studio for Mac : Build Cross-Platform Apps with Xamarin and .NET Core*. Cremona: Apress.

Depari, Ganti. (2013). *Teknik Digital - Teori dan Aplikasi*. Bandung: Nuansa Aulia.

Dobrzański, L.A., & Honysz, R. (2010). *The idea of material science virtual Laboratory*. Jurnal of Achievements in Materials and Manufacturing Engineering. Vol. 42 Issues 1-2.

Eberly, David H. (2005). *3D Game Engine Architecture. Engineering Real-Time Applications with Wild Magic*. California: Morgan Kaufmann.

Endres, Albert., & Rombach, Dieter. (2003). *A Handbook of Software and Systems Engineering - Empirical Observations, Laws and Theories*. Harlow: Pearson Education Limited.

Everett, Gerald D., & McLeod, Raymond. (2007). *Software Testing - Testing Across the Entire Software Development Life Cycle*. Hoboken: Wiley.

Evitts, Paul. (2000). *A UML Pattern Language. Software Engineering Series*. Indianapolis: MTP.

Ferdjallah, Mohammed. (2011). *Introduction to Digital Systems - Modeling, Synthesis & Simulation Using VHDL*. Virginia: Wiley.

Festo Didactic. (2014). *Technology for education and science - The current range of Festo Didactic products 2014*. Denkendorf: Festo Didactic Germany GmbH & Co. KG.

Fuller, Mike. (2004). *Virtual Learning and Higher Education*; *Assessment for Real in Virtual Learning Environments - How Far Can We Go?*. Amsterdam: Rodopi B.V.

Galin, Daniel. (2004). *Software Quality Assurance - From theory to implementation*. England: Pearson.

Gao, Jerry Zeyu., Tsao, H.S. Jacob., & Wu, Ye. (2003). *Testing and Quality Assurance for Component-Based Software*. Norwood: Artech House.

Gao, Jerry Zeyu., Shim, Simon., Mei, Hsing., & Su, Xiao. (2006). *Engineering Wireless-Based Software Systems and Applications*. Norwood: Artech House.

Gausemeier, Jürgen., Berssenbrügge, Jan., Grafe, Michael., Kahl, Sascha & Wassmann, Helene. (2011). *Virtual Reality & Augmented Reality in Industry - The 2nd Sino-German Workshop; Design and VR/AR-based Testing of Advanced Mechatronic Systems.* Shanghai: Springer.

George, Darren., & Mallery, Paul. (2016). *IBM SPSS Statistics 23 Step by Step - A Simple Guide and Reference, 4th Edition*. Oxon: Routledge.

Gillespie, Helena., Boulton, Helen., Hramiak, Alison., & Williamson, Richard. (2007). *Learning and Teaching with Virtual Learning Environments*. Exeter: Learning Matters.

Goyal, Vikram. (2006). *Pro Java ME MMAPI - Mobile Media API for Java Micro Edition*. California: Apress.

Gregory, Jason. (2009). *Game Engine Architecture*. Florida: CRC Press.

Gu, Ning., Gul, Leman Figen., Williams, Anthony., & Nakapan, Walaiporn. (2009). *Higher Education in Virtual Worlds – Teaching and Learning in Second Life; Second Life – a Context for Design Learning.* Bingley: Emerald.

Gupta, Aditya. (2014). *Learning Pentesting for Android Devices - A practical guide to learning penetration testing for Android devices and applications*. Mumbai: Packt Publishing.

Harbour, Jonathan S. (2011). *Multi-Threaded Game Engine Design*. Canada: Course Technology.

Hariwibowo, Agus., Muafa, Ahsan., & Sunaryantiningsih, Ina. (2014). *Pengaruh Media Simulasi Komputer Terhadap Aktifitas dan Kemampuan Mahasiswa Prodi PTE*. Jurnal LPPM Vol. 2 No. 1.

Heitlager, Ilja., Kuipers, Tobias., & Visser, Joost. (2007). *A Practical Model for Measuring Maintainability*. Software Improvement Group, 6-7.

Hermes, Dan. (2015). *Xamarin Mobile Application Development : Cross-Platform C# and Xamarin.Forms Fundamentals.* California: Apress.

Hidayani, Noor. (2012). *Bentuk Aljabar*. Jakarta: Balai Pustaka.

Hu, Bin., Moore, Philip., & Jackson, Mike. (2010). *Computational Intelligence for Technology Enhanced Learning; Fuzzy ECA Rules for Pervasive Decision-Centric Personalised Mobile Learning*. Poland: Springer.

Ibrahim, K.F. (1996). *Teknik Digital*. Yogyakarta: Andi.

Ito, Mizuko. (2006). *The International Handbook of Virtual Learning Environments Volume I; Interaction, Collusion, and the Human–Machine Interface*. Dordrecht: Springer.

Indriana, Dina. (2011). *Ragam Alat Bantu Media Pengajaran*. Yogyakarta: Diva Press.

Jalote, Pankaj. (2008). *A Concise Introduction to Software Engineering*. India: Springer.

Jamo Solutions. (2013). *Performance Testing and Monitoring of Mobile Applications*. Jamo Solutions N.V. Part Number MT-PT, Revision 2468, August 2013.

Janet, MacDonald., & Creanor, Linda. (2010). *Learning with Online and Mobile Technologies - A Student Survival Guide.* Farnham: Gower Publishing.

Jiang, Mangqin. (2011). *Virtual Reality & Augmented Reality in Industry - The 2nd Sino-German Workshop; Virtual Reality Boosting Automotive Development*. Shanghai: Springer.

Joni, Dewa Adi Baskara., & Suryani, Erma. (2012). *Analisis Efisiensi Layanan Perbankan untuk Meningkatkan Kinerja Layanan Menggunakan Simulasi Diskrit*. Jurnal Sistem Informasi, Volume 4, No. 2.

Kaeslin, Hubert. (2008). *Digital Integrated Circuit Design - From VLSI Architectures to CMOS Fabrication*. Cambridge: Cambridge University Press.

Kovács, A., & Szabados, K. (2013). *Test software quality issues and connections to international standards*. Acta Univ. Sapientiae, Informatica, 5, 1.2013.77-102.

Koyya, Pardha., Lee, Young., & Yang, Jeong. (2013). *Feedback for Programming Assignments Using Software-Metrics and Reference Code*. International Scholarly Research Network (ISRN) Software Engineering. Department of Electrical Engineering & Computer Science, University-Kingsville.

Kukulska, Agnes., & Traxler, John. (2005). *Mobile Learning - A Handbook for Educators and Trainers*. Oxon: Routledge.

Landriscina, Franco. (2013). *Simulation and Learning - A Model-Centered Approach*. Italy: Springer.

Laplante, Phillip A. (2007). *What Every Engineer Should Know about Software Engineering*. Florida: CRC Press.

Larasati, D.S., & Sukisno, M. (2014). *Penggunaan Media Simulasi Berbasis Teknologi Informasi dalam Pembelajaran Fisika pada Siswa Lintas Minat*. *Unnes Physic Education Journal* 3 (3).

Lee, Wei-Meng. (2012). *Beginning Android™ 4 Application Development*. Indiana: Wiley Publishing

Leblebici, Yusuf., & Kang, Sung-Mo. (2003). *CMOS Digital Integrated Circuits - Analysis and Design, 3rd Edition*. Swiss: McGraw-Hill.

Liao, Xiaofei., Xiong, Xianjie., Jin, Hai., & Hu, Liting. (2008). *Systems and Virtualization Management - Standards and New Technologies; LVD: A Lightweight Virtual Desktop Management Architecture*. Munich: Springer.

Lin, Yu-Ming., Han, Shu-Jen., Farmer, Damon B., Meric, Inanc., Sun, Yanning., Wu, Yanqing., Dimitrakopoulos, Christos., & Jenkins, Keith A. (2011). *Wafer-Scale Graphene Integrated Circuit.* Jurnal Sciencemag.org, DOI: 10.1126/science.1204428.

Lindenmayer, Aristid., & Prusinkiewicz, Przemyslaw. (2004). *The Virtual Laboratory - The Algorithmic Beauty of Plants.* Berlin: Springer.

Louloudi, Athanasia., & Klügl, Franziska. (2013). *Cognitive Agents for Virtual Environments*; *Visualisation on Demand for Agent-Based Simulation.* Valencia: Springer.

Lund, A.M. (2001). *Measuring Usability with the USE Questionnaire*. STC Usability SIG Newsletter. [On-Line] available at https://www.researchgate.net/publication/230786746/. Diakses 12 Mei 2016, Jam 03.00 WIB.

Madjarov, Ivan., & Boucelma, Omar. (2011). *Social Media Tools and Platforms in Learning Environments; Multimodality and Context Adaptation for Mobile Learning*. Berlin: Springer.

Mahendra, I Komang Ari., Darmawiguna, I Gede Mahendra., & Kesiman, Made Windu Antara. (2014). *Pengembangan Media Pembelajaran Berbasis Simulasi untuk Pembelajaran Perakitan Komputer dan Instalasi Sistem Operasi*. Jurnal Mahasiswa Pendidikan Teknik Informatika Universitas Pendidikan Ganesha, Singaraja Bali. ISSN 2252-9063 Vol. 3, No. 3.

Martinez, Jaime. (2011). *A Performatory Approach to Teaching, Learning and Technology*. Rotterdam: Sense Publishers.

Mayer, Richard E. (2009). *Multimedia Learning - Second Edition*. Cambridge: CU Press.

McCabe, Thomas J. (1976). *A Complexity Measure*. IEEE Transactions on Software Engineering, SE-2, No.4.

Microsoft, Developer. (2007). *Maintainability Index Range and Meaning*. Akses tanggal 07.12.2017 dari https://blogs.msdn.microsoft.com/codeanalysis/2007/11/20/maintainability-index-range-and-meaning/

Milano, Diego Torres. (2011). *Android Application Testing Guide - Build intensively tested and bug free Android applications*. Birmingham: Packt Publishing.

Miller, Christopher. (2017). *Cross-platform Localization for Native Mobile Apps with Xamarin*. Slingerlands: Apress.

Millington, Ian. (2007). *Game Physics Engine Development*. Oxford: Morgan Kaufmann.

Moraes, Ronei M., & Machado, Liliane S. (2012). *Assessment Systems for Training Based on Virtual Reality: A Comparison Study*. SBC *Journal on 3D Interactive Systems*, volume 3, number 1.

Morris, Ben., Bortenschlager, Manfred., Lansdell, Jon., Luo, Cheng., & Somerville, Michelle. (2010). *Introduction to bada A Developer's Guide*. Chichester: Wiley.

Morse, Shona., Littleton, Fiona., Macleod, Hamish., & Ewins, Rory. (2009). *Higher Education in Virtual Worlds: Teaching and Learning in Second Life* ; *The Theatre of Performance Appraisal: Role-Play in Second Life.* Bingley: Emerald.

Munadi, Sudji. (2010). *Development Of A Modul For Computer Aided Contextual-Contructivistics Learning In The Subject Of Machining.* Jurnal Vocational Eductaion and Traning *"The Challenges for Vocational Education and Training in Developing Skills for Today's Workforce"*.

Najadat, Hassan., Alsmadi, Izzat., & Shboul, Yazan. (2012). *Predicting Software Projects Cost Estimation Based on Mining Historical Data*. International Scholarly Research Network (ISRN) Software Engineering. Computer Information Systems Department, Jordan University of Science and Technology.

Nakamura, Masumi., & Gargenta, Marko. (2014). *Learning Android – Develop Mobile Apps Using Java and Eclipse, 2nd Edition*. Sebastopol: O'Reilly Media.

Nayrolles, Mathieu. (2015). *Xamarin Studio for Android Programming - A C# Cookbook. Over 50 hands-on recipes to help you get grips with Xamarin Studio and C# programming to develop market-ready Android applications*. Birmingham: Packt Publishing.

Nitin, Khanna., & Gok, Nizamettin. (2013). *Building Hybrid Android Apps with Java and JavaScript*. Sebastopol: O'Reilly Media.

Nolan, Godfrey. (2015). *Agile Android*. California: Apress Media.

Novianti, Ariza., & Fauzijah, Ami. (2009). *Sistem Informasi Sekolah Dasar Berbasis SMS*. Seminar Nasional Aplikasi Teknologi Informasi 2009 (SNATI 2009). ISSN : 1907-5022.

Nutaro, James. (2011). *Building Software for Simulation - Theory and Algorithms with Applications in C++*. Canada: Wiley.

Pachler, Norbert., Bachmair, Ben., & Cook, John. (2010). *Mobile Learning - Structures, Agency, Practices*. London: Springer.

Panhale, Mahesh. (2016). *Beginning Hybrid Mobile Application Development*. California: Apress.

Panigrahy, Parth Sarathi., Konar, Pratyay., & Chattopadhyay, Paramita. (2014). *Broken Bar Fault Detection using Fused DWT-FFT in FPGA Platform*. Jurnal IEEE Indian Institute of Engineering Science and Technology, Shibpur, Howrah, India.

Parisi, Tony. (2015). *Learning Virtual Reality - Developing Immersive Experiences and Applications for Desktop, Web and Mobile*. Sebastopol: O'Reilly Media.

Pelz, Georg. (2003). *Mechatronic Systems - Modelling and Simulation with HDLs. [Modellierung und Simulation Mechatronischer Systeme]*. Munich: Wiley.

Peppers, Jonathan. (2014). *Xamarin Cross-platform Application Development - Develop production-ready applications for iOS and Android using Xamarin*. Birmingham: Packt Publishing.

Peppers, Jonathan. (2016). *Xamarin 4.x Cross-Platform Application Development – 3rd Edition. Develop powerful cross-platform applications with Xamarin*. Birmingham: Packt Publishing.

Petzold, Charles. (2015). *Creating Mobile Apps with Xamarin.Forms : Cross-platform C# programming for iOS, Android, and Windows Phone*. Redmond: Microsoft Press.

Pocatilu, Paul. (2006). *Influencing Factors of Mobile Applications' Quality Metrics*. Economy Informatics Journal, 1-4/2006. Economic Informatics Department, Academy of Economic Studies, Bucharest.

Polosoro, Eko. (2009). *Sistem Digital*. Yogyakarta: Graha Ilmu.

Prastuti, Sulistyorini. (2009). *Pemodelan Visual dengan Menggunakan UML dan Rational Rose.* Jurnal Teknologi Informasi DINAMIK Volume XIV, No.1, Januari 2009 : 23-29. ISSN : 0854-9524

Pressman, Roger S. (2002). *Rekayasa Perangkat Lunak - Pendekatan Praktisi*. Yogyakarta: Andi.

Pressman, Roger S. (2010). *Software Engineering - A Practitioner's Approach, 7th Edition*. NewYork: The McGraw-Hill.

Pressman, Roger S. (2012). *Rekayasa Perangkat Lunak: Pendekatan Praktisi*. (Edisi 7) Buku 1. Yogyakarta: Andi.

Purwoko, Bambang Setiyo Hari. (2009). *Pengembangan Mesin CNC Virtual Sebagai Media Interaktif untuk Melayani Pembelajaran Pemprograman CNC.* Jurnal Seminar Nasional Electrical, Informatics, and It's Educations 2009.

Purwoko, Bambang Setiyo Hari. (2011). *Pengembangan Mesin Bubut Personal Komputer sebagai Media Pembelajaran Praktik Pemrograman CNC*. Jurnal Penelitian Pendidikan. Lembaga Penelitian Universitas Negeri Yogyakarta.

Raj, Gururajan., Danaher, Patrick., & Hafeez-Baig, Abdul. (2009). *Innovative Mobile Learning: Techniques and Technologies; Transforming the Practice of Mobile Learning: Promoting Pedagogical Innovation through Educational Principles and Strategies that Work.* Hershey: IGI Global.

Ramadhani, Muhammad Syari'ati., & Ngadiyono, Yatin. (2016). *Penerapan Modul Inventor Dengan Pendekatan CTL Untuk Meningkatkan Kompetensi Siswa Pada Mata Diklat CAD*. Jurnal Dinamika Vokasional Teknik Mesin Volume 1, No.1 Oktober 2016. Universitas Negeri Yogyakarta.

Reynolds, Mark. (2014). *Xamarin Essentials - Learn how to efficiently develop Android and iOS apps for deployment using the Xamarin platform*. Birmingham: Packt Publishing.

Reynolds, Mark. (2014). *Xamarin Mobile Application Development for Android - Learn to develop full featured Android apps using your existing C# skills with Xamarin.Android*. Birmingham: Packt Publishing.

Riduwan. (2013). *Skala Pengukuran Variabel - Variabel Penelitian*. Bandung: Alfabeta.

Rizky, Soetam. (2011). *Konsep Dasar Rekayasa Perangkat Lunak - Software Reengineering.* Jakarta: Prestasi Pustaka.

Rochayati, Umi., Waluyanti, Sri., & Santoso, Djoko. (2012). *Inovasi Media Pembelajaran Sain Teknologi di SMP Berbasis Mikrokontroler*. Jurnal Kependidikan, Volume 42, No.1, 90.

Rogers, Rick., Lombardo, John., Mednieks, Zigurd., & Meike, Blake. (2009). *Android Application Development*. Sebastopol: O'Reilly Media.

Roland, Sussex. (2012). *Computer-Enhanced and Mobile-Assisted Language Learning: Emerging Issues and Trend; Text Input and Editing as a Bottleneck in Mobile Devices for Language Learning.* Hershey: IGI Global.

Rosa, A.S., & Shalahuddin, M. (2011). *Rekayasa Perangkat Lunak - Terstruktur dan Beorientasi Objek*. Bandung: Modula.

Rosadi, Dadi., & Purnomo. (2012). *Simulasi Kredit Pemasaran Mobil Bekas Berbasis Web Menggunakan Codeigniter Framework*. Jurnal Computech & Bisnis, Vol. 6, No. 1, Juni 2012, 34-39. ISSN 2442-4943.

Ruiz, Antonio Pachón. (2015). *Mastering Android Application Development*. Birmingham: Packt Publishing.

Russell, Nick., Aalst, Wil M.P. van der., Hofstede, Arthur H.M. ter., & Wohed, Petia. (2006). *On the Suitability of UML 2.0 Activity Diagrams for Business Process Modelling (Russell, 2006)*. *In: Conceptual Modelling 2006: Proceedings of APCCM2006*.

Ryu, Hokyoung., & Parsons, David. (2009). *Innovative Mobile Learning: Techniques and Technologies*; *Designing Learning Activities with Mobile Technologies.* Hershey: IGI Global.

Sanaky, Hujair. (2013). *Media Pembelajaran Interaktif-Inovatif*. Yogyakarta: Kaukaba Dipantara.

Schulmeyer, Gordon. (2008). *Handbook of Software Quality Assurance - 4th Edition*. Norwood: Artech House.

Schwab, Brian. (2009). *AI Game Engine Programming*. Boston: Course Technology.

Schwarz, Ronan., Dutson, Phil., Steele, James., & To, Nelson. (2013). *The Android Developer's Cookbook - Building Applications with The Android SDK - 2nd Edition*. Indiana: Pearson Education.

Shetty, Devdas., & Kolk, Richard A. (2011). *Mechatronics System Design, 2nd Edition*. Stamford: Cengage Learning.

Siddiqi, Sameer., Lee, Rebecca E., Layne, Charles S., McFarlin, Brian K., & O'Connor, Daniel. (2010). *Cases on Collaboration in Virtual Learning Environments: Processes and Interactions; Obesity Prevention in Second Life: The International Health Challenge*. Hershey: IGI Global.

Slator, B.M., Clark, J.T., Daniels, L.M., Hill, C., McClean, P., Saini-Eidukat, B., Schwert, D.P., & White, A.R. (2002). *Virtual Environments for Teaching & Learning. Series on Innovative Intelligence – Vol. 1*; *Use of Virtual Worlds to Teach the Sciences.* Singapore: World Scientific Publishing.

Smith, William. (2014). *Learning Xamarin Studio - Learn how to build high-performance native applications using the power of Xamarin Studio*. Birmingham: Packt Publishing.

Sokolowski, John A., & Banks, Catherine M. (2011). *Principles of Modeling and Simulation – A Multidiciplinary Approach*. Canada: Wiley.

Sommerville, Ian. (2011). *Software engineering, 9th Edition*. Boston: Pearson Education.

Song, Yanjie. (2009). *Innovative Mobile Learning: Techniques and Technologies; Handheld Educational Applications: A Review of the Research*. Hershey: IGI Global.

Steele, James., & To, Nelson. (2011). *The Android developer's Cookbook - Building Applications with The Android SDK*. Indiana: Pearson Education.

Suh, Suk-Hwan., Kang, Seong-Kyoon., Chung, Dae-Hyuk., & Stroud, Ian. (2008). *Theory and Design of CNC Systems*. Pohang: Springer.

Surendro, Krisnha Prasetyo. (2010). *Menentukan Optimasi Routing dengan Pengaturan Route Advertisement pada Jaringan Mobile IPV6*. InComTech, Jurnal Telekomunikasi dan Komputer, vol. 1, no. 2.

Surjono, Herman Dwi. (2017). *Multimedia Pembelajaran Interaktif - Konsep dan Pengembangan*. Yogyakarta: UNY Press.

Taguchi, Genichi., Chowdhury, Subir., & Wu, Yuin. (2005). *Taguchi's Quality Engineering Handbook*. Hoboken: Wiley.

Taguchi, Genichi., Taguchi, Shin., & Jugulum, Rajesh. (2005). *Computer-Based Robust Engineering - Essentials for DFSS.* Milwaukee: ASQ Press.

Talekar, Aniruddha., Patil, Saurabh., Thakre, Prashant., & Rajkumar, E. (2017). *Virtual reality and its applications in manufacturing industries.* Journal of Chemical and Pharmaceutical Sciences. JCPS Volume 10 Issue 1. ISSN: 0974-2115.

Thackray, Liz., Good, Judith., & Howland, Katherine. (2010). *Researching Learning in Virtual Worlds; Learning and Teaching in Virtual Worlds: Boundaries, Challenges and Opportunities.* Lancaster: Springer.

Tooley, Mike. (2007). *Aircraft Digital Electronic and Computer Systems: Principles, Operation and Maintenance.* Burlington: Elsevier.

Tullis, Thomas., & Albert, Bill. (2013). *Measuring The User Experience-Collecting, Analyzing, and Presenting Usability Metrics, 2nd Edition.* Waltham: Elsevier.

Vajna, Sandor., Weber, Christian., Zeman, Klaus., Hehenberger, Peter., Gerhard, Detlef., & Wartzack, Sandro. (2018). *CAx für Ingenieure - Eine praxisbezogene Einführung 3 Aulage.* Berlin: Springer.

Versluis, Gerald. (2017). *Xamarin.Forms Essentials - First Steps Toward Cross-Platform Mobile Apps*. Sittard: Apress.

Wagner, Richard. (2011). *Professional Flash® Mobile Development - Creating Android™ and iPhone® Applications*. Indiana: Wiley Publishing.

Wagner, Stefan. (2013). *Software Product Quality Control*. Berlin: Springer.

Wang, Peng., Wu, Peng., Wang, Jun., Chi, Hung-Lin., & Wang, Xiangyu. (2018). *A Critical Review of the Use of Virtual Reality in Construction Engineering Education and Training*. International Journal of Environmental Research and Public Health, 15, 1204.

Watkins, John., & Mills, Simon. (2011). *Testing IT - An Off-the-Shelf Software Testing Process - 2nd Edition*. Cambridge: Cambridge University Press.

Weiss, Joel. (2006). *The International Handbook of Virtual Learning Environments Volume I; Virtual Learning and Learning Virtually*. Dordrecht: Springer.

Widjanarka, Widjaya. (2006). *Teknik Digital*. Jakarta: Erlangga.

Wihlidal, Graham. (2006). *Game Engine Toolset Development*. Boston: Thomson Course Technology.

Wilcox, Mark. (2009). *Porting to the Symbian Platform - Open Mobile Development in C/C++*. Chichester: Wiley.

Williams, L. (2006). *Testing Overview and Black-Box Testing Techniques*. http://agile.csc.ncsu.edu/SEMaterials/BlackBox.pdf. Akses 12 Mei 2016, Jam 02.00 WIB.

Woodill, Gary. (2011). *The Mobile Learning Edge - Tools and Technologies for Developing*. US: The McGraw-Hill.

Xiao, Hong. (2016). *3D IC Devices, Technologies, and Manufacturing*. Bellingham: SPIE Press.

Yabuki, Nobuyoshi. (2011). *Collaborative Design in Virtual Environments; Impact of Collaborative Virtual Environments on Design Process*. Chennai: Springer.

Zerbst, Stefan. (2004). *3D Game Engine Programming*. Boston: Thomson Course Technology.

Žižek, Slavoj. (2006). *The International Handbook of Virtual Learning Environments Volume I; The Matrix, or, the Two Sides of Perversion.* Dordrecht: Springer.

LAMPIRAN

Lampiran 1. Tabel *Test Case* "DigiChip"

Jalur	*Case*
1	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NOT, simulasi Gerbang Logika NOT
2	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem AND, simulasi Gerbang Logika AND
3	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NAND, simulasi Gerbang Logika NAND
4	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem OR, simulasi Gerbang Logika OR
5	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NOR, simulasi Gerbang Logika NOR
6	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem XOR, simulasi Gerbang Logika XOR
7	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Tampilan Penuh, tombol keluar pilihan simulasi
8	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → keluar tutorial
9	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → keluar Petunjuk Pengoperasian
10	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur

	Tampilan → tutorial Menampilkan *Scematic*, tombol *next* Menampilkan *Scematic* → tombol *previous* Menampilkan *Scematic*
11	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Pasang Kabel Konektor, tombol *next* Pasang Kabel Konektor → tombol *previous* Pasang Kabel Konektor
12	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Lepas Kabel Konektor, tombol *next* Lepas Kabel Konektor → tombol *previous* Lepas Kabel Konektor
13	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Tombol *Input*, tombol *next* Tombol *Input* → tombol *previous* Tombol *Input*
14	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial LED *Output*, tombol *next* LED *Output* → tombol *previous* LED *Output*
15	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → tombol *Truth Tabel* Gerbang Logika, menampilkan *Truth Tabel* Gerbang Logika
16	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → tombol KELUAR, aplikasi proses keluar

Lampiran 2. Tabel Hasil *Integration Testing* "DigiChip"

Jalur	Pengujian Sistem	Hasil
1	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NOT, simulasi Gerbang Logika NOT	Berjalan Baik
2	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem AND, simulasi Gerbang Logika AND	Berjalan Baik
3	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NAND, simulasi Gerbang Logika NAND	Berjalan Baik
4	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem OR, simulasi Gerbang Logika OR	Berjalan Baik
5	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NOR, simulasi Gerbang Logika NOR	Berjalan Baik
6	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem XOR, simulasi Gerbang Logika XOR	Berjalan Baik
7	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Tampilan Penuh, tombol keluar pilihan simulasi	Berjalan Baik
8	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → keluar tutorial	Berjalan Baik
9	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU,	

	menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → keluar Petunjuk Pengoperasian	Berjalan Baik
10	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Menampilkan *Scematic*, tombol *next* Menampilkan *Scematic* → tombol *previous* Menampilkan *Scematic*	Berjalan Baik
11	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Pasang Kabel Konektor, tombol *next* Pasang Kabel Konektor → tombol *previous* Pasang Kabel Konektor	Berjalan Baik
12	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Lepas Kabel Konektor, tombol *next* Lepas Kabel Konektor → tombol *previous* Lepas Kabel Konektor	Berjalan Baik
13	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Tombol *Input*, tombol *next* Tombol *Input* → tombol *previous* Tombol *Input*	Berjalan Baik
14	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial LED *Output*, tombol *next* LED *Output* → tombol *previous* LED *Output*	Berjalan Baik
15	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → tombol *Truth Tabel* Gerbang Logika, menampilkan *Truth Tabel* Gerbang Logika	Berjalan Baik
16	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → tombol KELUAR, aplikasi proses keluar	Berjalan Baik

Lampiran 3. Tabel Penghitungan *Source Code* "DigiChip"

File Script	Jumlah Baris	*File Script*	Jumlah Baris
BezierLines.cs	101	ObyekPutarBebas.cs	16
ButtonChecker.cs	38	ObyekReposisi.cs	35
CameraLookSwipe.cs	62	ObyekSlerper.cs	48
CompleteCameraContlr.cs	16	ObyekUpDown.cs	21
Door.cs	27	OnMousedown.cs	13
GeneratePlaneMesh.cs	55	OnMousedownPlayer.cs	15
Gizmo.cs	15	OnPointerdown.cs	50
InputHandler.cs	24	Path.cs	11
KameraDetectorPositn.cs	52	PathAgent.cs	46
KameraMoveKlik1.cs	18	PlayAudio.cs	23
KameraMoving.cs	24	RejCameraZoom.cs	119
KameraOrbit3Ax.cs	27	RejDestroy.cs	27
KameraOrbit3AxManage.cs	21	RejFinger.cs	378
KameraOrbitOtom.cs	28	RejFingerDown.cs	41
KurvaBezier.cs	53	RejFingerDrag.cs	54
MathClamp.cs	19	RejFingerHeld.cs	190
MathLerp.cs	18	RejFingerLine.cs	28
MouseOrbitImproved.cs	69	RejFingerSet.cs	49
MouseOrbitInfRtteZm.cs	75	RejFingerSwipe.cs	62
ObyekArrayControl.cs	31	RejFingerTap.cs	65
ObyekDiJalur.cs	19	RejFingerTrail.cs	147
ObyekDrag.cs	15	RejFingerUp.cs	49
ObyekDragDrop.cs	46	RejGesture.cs	505
ObyekGerakLerp.cs	51	RejMovePinch.cs	76
ObyekGerakTranslate.cs	18	RejMultiTap.cs	117
ObyekGeserPutar.cs	20	RejObjectMove.cs	81
ObyekIndikatorEnable.cs	28	RejPitchYaw.cs	170
ObyekKlikActive.cs	83	RejRotate.cs	80
ObyekKlikAnim.cs	303	RejScale.cs	113
ObyekKlikDelegate.cs	246	RejSelectable.cs	772
ObyekKlikDestroy.cs	47	RejSnapshot.cs	122
ObyekKlikGlow.cs	29	RejSpawn.cs	30
ObyekKlikGUI.cs	78	RejSwipeDirect.cs	144
ObyekKlikIdentify.cs	37	RejSwipeRB2D.cs	146
ObyekKlikInactive.cs	59	RejSwipeRB3D.cs	139
ObyekKlikLampu.cs	18	RejTapSelect.cs	39
ObyekKlikLerp.cs	48	RejTouchEv.cs	720

ObyekKlikPasangLepas.cs	828	ShowHide.cs	467
ObyekKlikPortANDIn.cs	61	SwipeTouch.cs	64
ObyekKlikPortANDOut.cs	101	TmblAnmtrMenu.cs	68
ObyekKlikPortNANDIn.cs	61	TmblGantiKamera.cs	14
ObyekKlikPortNANDOut.cs	110	TmblKameraLerp.cs	50
ObyekKlikPortNORIn.cs	61	TmblKameraMove.cs	65
ObyekKlikPortNOROut.cs	110	TmblMenu.cs	87
ObyekKlikPortNOTIn.cs	70	TmblObyekTrans.cs	19
ObyekKlikPortNOTOut.cs	197	TmblDetector.cs	32
ObyekKlikPortORIn.cs	61	TouchControl.cs	43
ObyekKlikPortOROut.cs	110	TouchDetector.cs	204
ObyekKlikPortXORIn.cs	61	TouchGeser.cs	16
ObyekKlikPortXOROut.cs	110	TouchInput.cs	17
ObyekKlikPortZonk.cs	49	TouchLogic.cs	63
ObyekKlikTrans.cs	13	TouchPinchMove.cs	58
ObyekKlikWarna.cs	145	TouchPinchZoom.cs	42
ObyekLerper.cs	48	TouchSwipeRotate.cs	41
ObyekLook.cs	47	TransformPoint.cs	15
ObyekMouseShade.cs	25	TransTarget.cs	18
ObyekMoveTouch.cs	31	VoidAnalyzer.cs	20
ObyekMuter.cs	82	WayPoints.cs	98
Jumlah Total			**10.541**

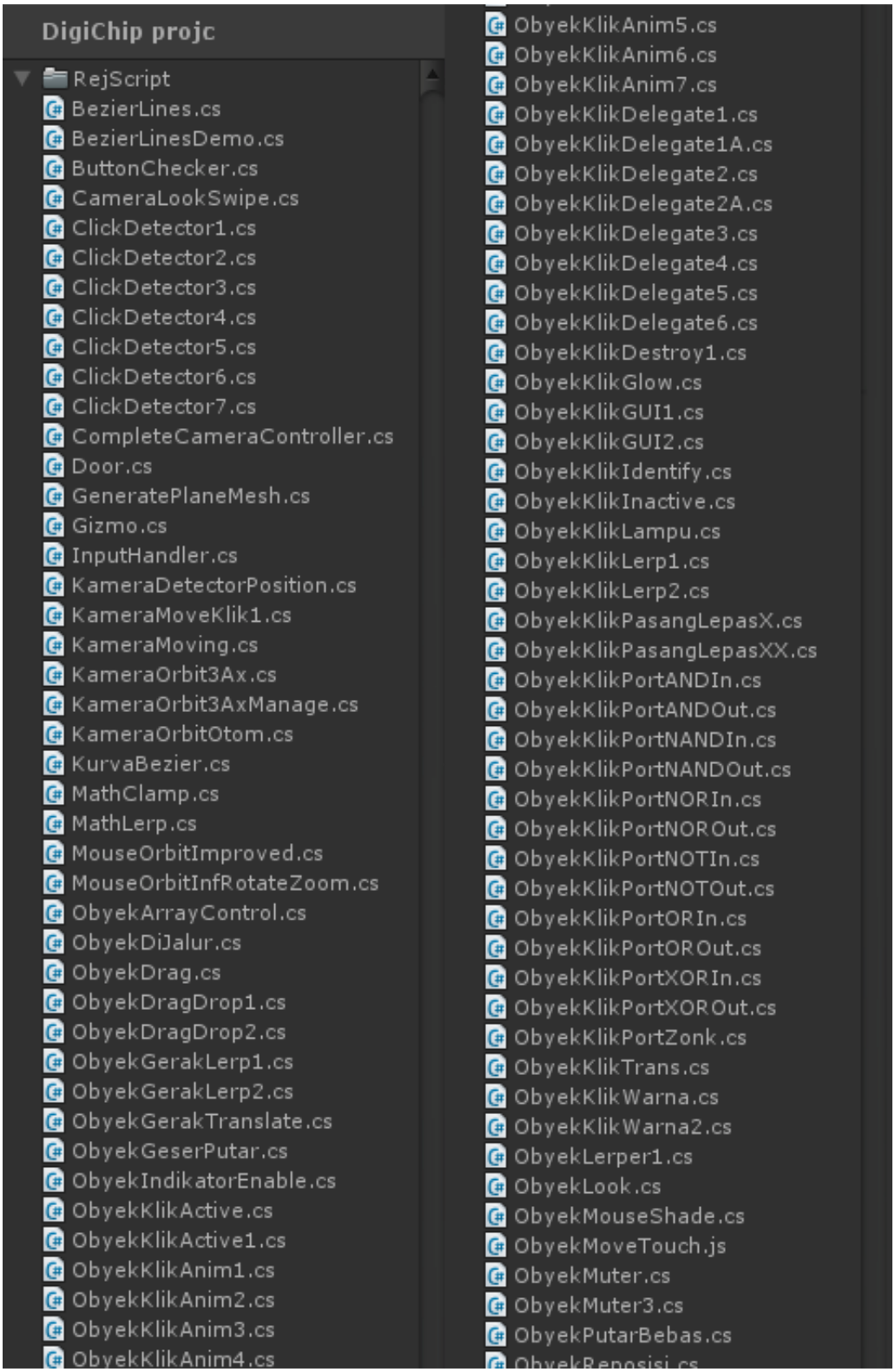
Lampiran 4. Tabel Hasil Pengujian *Functional Suitability* "DigiChip"

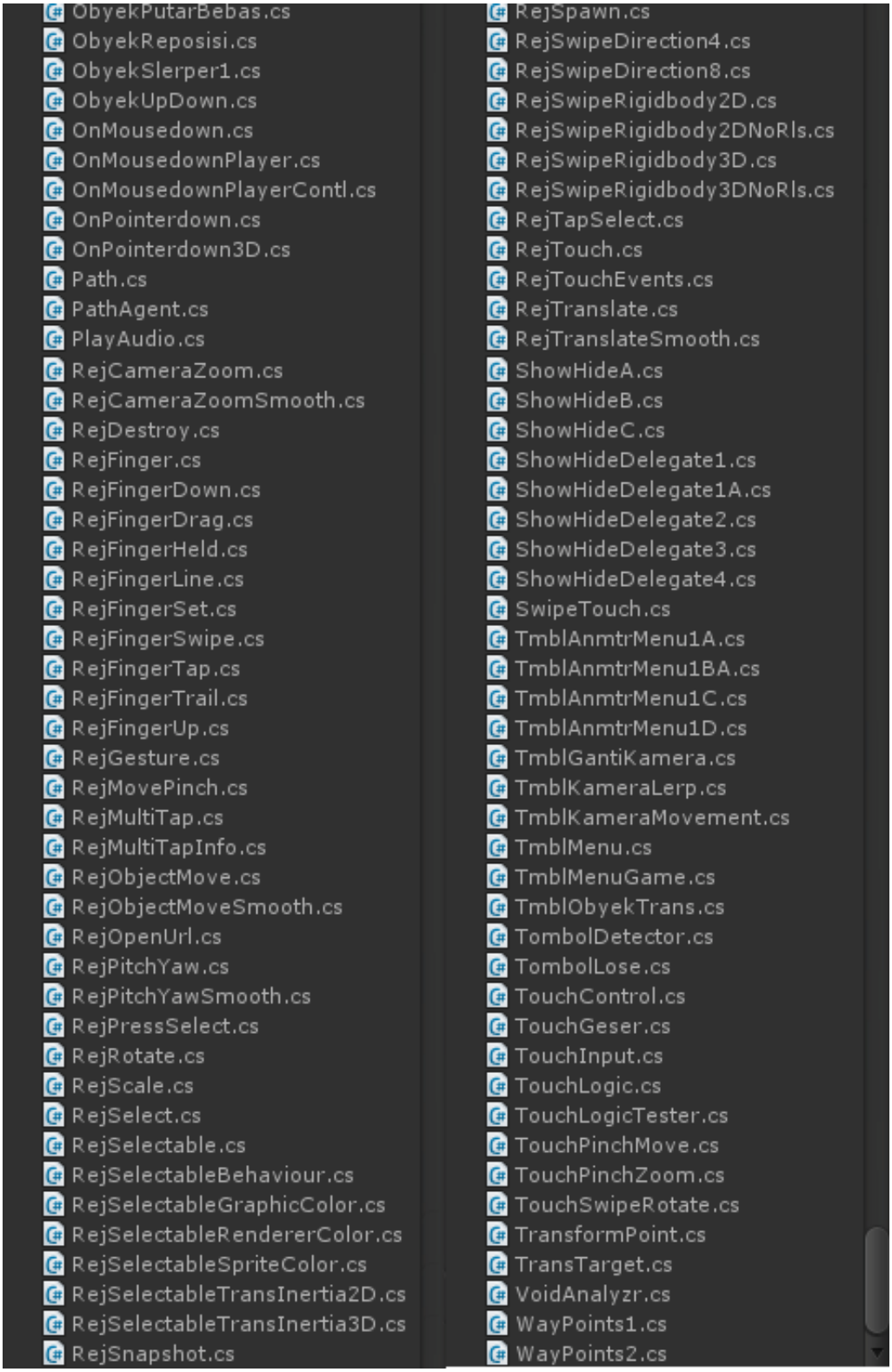
Jalur	Pengujian Sistem	Hasil
1	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NOT, simulasi Gerbang Logika NOT	Berjalan Baik
2	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem AND, simulasi Gerbang Logika AND	Berjalan Baik
3	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NAND, simulasi Gerbang Logika NAND	Berjalan Baik
4	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem OR, simulasi Gerbang Logika OR	Berjalan Baik
5	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem NOR, simulasi Gerbang Logika NOR	Berjalan Baik
6	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Sistem XOR, simulasi Gerbang Logika XOR	Berjalan Baik
7	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol MULAI, menampilkan pilihan simulasi → tombol Tampilan Penuh, tombol keluar pilihan simulasi	Berjalan Baik
8	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → keluar tutorial	Berjalan Baik

9	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → keluar Petunjuk Pengoperasian	Berjalan Baik
10	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Menampilkan *Scematic*, tombol *next* Menampilkan *Scematic* → tombol *previous* Menampilkan *Scematic*	Berjalan Baik
11	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Pasang Kabel Konektor, tombol *next* Pasang Kabel Konektor → tombol *previous* Pasang Kabel Konektor	Berjalan Baik
12	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Lepas Kabel Konektor, tombol *next* Lepas Kabel Konektor → tombol *previous* Lepas Kabel Konektor	Berjalan Baik
13	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial Tombol *Input*, tombol *next* Tombol *Input* → tombol *previous* Tombol *Input*	Berjalan Baik
14	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → *touch* Petunjuk Pengoperasian → tutorial Atur Tampilan, tombol *next* Atur Tampilan → tutorial LED *Output*, tombol *next* LED *Output* → tombol *previous* LED *Output*	Berjalan Baik
15	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → *touch* tombol TUTOR, menampilkan tutorial → tombol *Truth Tabel* Gerbang Logika, menampilkan *Truth Tabel* Gerbang Logika	Berjalan Baik

16	Mulai menjalankan aplikasi, muncul *splashscreen* dan proses *loading* pada sistem → masuk halaman *home*, *touch* tombol MENU, menampilkan tombol MULAI - TUTOR - KELUAR → tombol KELUAR, aplikasi proses keluar	Berjalan Baik
Nilai Hasil		**16**
Nilai Maksimal		**16**

Lampiran 5. *File Script* "DigiChip"




ObyekPutarBebas.cs
ObyekReposisi.cs
ObyekSlerper1.cs
ObyekUpDown.cs
OnMousedown.cs
OnMousedownPlayer.cs
OnMousedownPlayerContl.cs
OnPointerdown.cs
OnPointerdown3D.cs
Path.cs
PathAgent.cs
PlayAudio.cs
RejCameraZoom.cs
RejCameraZoomSmooth.cs
RejDestroy.cs
RejFinger.cs
RejFingerDown.cs
RejFingerDrag.cs
RejFingerHeld.cs
RejFingerLine.cs
RejFingerSet.cs
RejFingerSwipe.cs
RejFingerTap.cs
RejFingerTrail.cs
RejFingerUp.cs
RejGesture.cs
RejMovePinch.cs
RejMultiTap.cs
RejMultiTapInfo.cs
RejObjectMove.cs
RejObjectMoveSmooth.cs
RejOpenUrl.cs
RejPitchYaw.cs
RejPitchYawSmooth.cs
RejPressSelect.cs
RejRotate.cs
RejScale.cs
RejSelect.cs
RejSelectable.cs
RejSelectableBehaviour.cs
RejSelectableGraphicColor.cs
RejSelectableRendererColor.cs
RejSelectableSpriteColor.cs
RejSelectableTransInertia2D.cs
RejSelectableTransInertia3D.cs
RejSnapshot.cs
RejSpawn.cs
RejSwipeDirection4.cs
RejSwipeDirection8.cs
RejSwipeRigidbody2D.cs
RejSwipeRigidbody2DNoRls.cs
RejSwipeRigidbody3D.cs
RejSwipeRigidbody3DNoRls.cs
RejTapSelect.cs
RejTouch.cs
RejTouchEvents.cs
RejTranslate.cs
RejTranslateSmooth.cs
ShowHideA.cs
ShowHideB.cs
ShowHideC.cs
ShowHideDelegate1.cs
ShowHideDelegate1A.cs
ShowHideDelegate2.cs
ShowHideDelegate3.cs
ShowHideDelegate4.cs
SwipeTouch.cs
TmblAnmtrMenu1A.cs
TmblAnmtrMenu1BA.cs
TmblAnmtrMenu1C.cs
TmblAnmtrMenu1D.cs
TmblGantiKamera.cs
TmblKameraLerp.cs
TmblKameraMovement.cs
TmblMenu.cs
TmblMenuGame.cs
TmblObyekTrans.cs
TombolDetector.cs
TombolLose.cs
TouchControl.cs
TouchGeser.cs
TouchInput.cs
TouchLogic.cs
TouchLogicTester.cs
TouchPinchMove.cs
TouchPinchZoom.cs
TouchSwipeRotate.cs
TransformPoint.cs
TransTarget.cs
VoidAnalyzr.cs
WayPoints1.cs
WayPoints2.cs

Lampiran 6. *Script Source-code* "DigiChip" *File Script [RejSelectable.cs]*

```
using Engine;
using Engine.UI;
using System.Collections;

// ___________________________________________________
// [Rejyasmito.Laboratory#TeachTech.project#DigiChip]
// ===================================================

public class RejSelectable : MonoBehaviour
{
    public Object[] portI, portO, kabelIA, kabelIB, kabelO;
    Ray ray;
    RaycastHit hit;
    string label;
//  string label = "IndexArray.";

    public GUISkin skineA, skineB, skineC; //Memanggil GUISkin.
    private Rect luasRect = new Rect ((Screen.width - 300) / 2, 9, 300, 121);
    private bool insAB1,insAB3,insAB5,insOP2,insOP4,insOP6,insT,zonk,zonkY; //popup Box Input A&B, Zonk.
    private bool insA1,insB1,insA3,insB3,insA5,insB5; //popup Button Input A&B.
    private bool insOP21,insOP22,insOP23,insOP41,insOP42,insOP43,insOP61,insOP62,insOP63; //popup Button Output LED.
    private bool insYT1,insYT2,insYT3,insYT4,insYT5,insYT6,insYT7,insYT8,insYT9,insYT10,insYT11,
                 insYT12,insYT13,insYT14,insYT15; //popup Box Lepas Kabel.
    private bool insY1A,insY1B,insY3A,insY3B,insY5A,insY5B,insY21O,insY22O,insY23O,insY41O,insY42O,
                 insY43O,insY61O,insY62O,insY63O; //popup Button Lepas Kabel.

    void Start ()
    {
        portI [0] = Object.Find ("Port.NOT.01");    portI [1] = Object.Find ("Port.NOT.03");
        portI [2] = Object.Find ("Port.NOT.05");    portI [3] = Object.Find ("Port.NOT.GND");
        portI [4] = Object.Find ("Port.NOT.09");    portI [5] = Object.Find ("Port.NOT.11");
        portI [6] = Object.Find ("Port.NOT.13");

        portO [0] = Object.Find ("Port.NOT.02");    portO [1] = Object.Find ("Port.NOT.04");
        portO [2] = Object.Find ("Port.NOT.06");    portO [3] = Object.Find ("Port.NOT.VCC");
        portO [4] = Object.Find ("Port.NOT.08");    portO [5] = Object.Find ("Port.NOT.10");
        portO [6] = Object.Find ("Port.NOT.12");

        //      ============================================================================

        kabelIA[0] = Object.Find ("Kabl.NOT.A1");
        kabelIA[1] = Object.Find ("Kabl.NOT.A3");
        kabelIA[2] = Object.Find ("Kabl.NOT.A5");
        if (kabelIA.Length > 0)
        {   foreach (GameObject xn_IA in kabelIA)
            { xn_IA.SetActive (false); } //Menyembunyikan semua kabel Input A.
        }
        kabelIB[0] = Object.Find ("Kabl.NOT.B1");
        kabelIB[1] = Object.Find ("Kabl.NOT.B3");
        kabelIB[2] = Object.Find ("Kabl.NOT.B5");
        if (kabelIB.Length > 0)
        {   foreach (Object xn_IB in kabelIB)
            { xn_IB.SetActive (false); } //Menyembunyikan semua kabel Input B.
        }
        kabelO[0] = Object.Find ("Kabl.NOT.O2.1");      kabelO[1] = Object.Find ("Kabl.NOT.O2.2");
        kabelO[2] = Object.Find ("Kabl.NOT.O2.3");      kabelO[3] = Object.Find ("Kabl.NOT.O4.1");
        kabelO[4] = Object.Find ("Kabl.NOT.O4.2");      kabelO[5] = Object.Find ("Kabl.NOT.O4.3");
        kabelO[6] = Object.Find ("Kabl.NOT.O6.1");      kabelO[7] = Object.Find ("Kabl.NOT.O6.2");
        kabelO[8] = Object.Find ("Kabl.NOT.O6.3");
        if (kabelO.Length > 0)
        {   foreach (Object xn_O in kabelO)
            { xn_O.SetActive (false); } //Menyembunyikan semua kabel Output.
        }
    }

    void Update ()
    {
        ray = Camera.main.ScreenPointToRay (Input.mousePosition);
        if (Physics.Raycast (ray, out hit) && Input.GetMouseButtonDown (0))
        {   //Pasang Input IC ke Saklar.
```

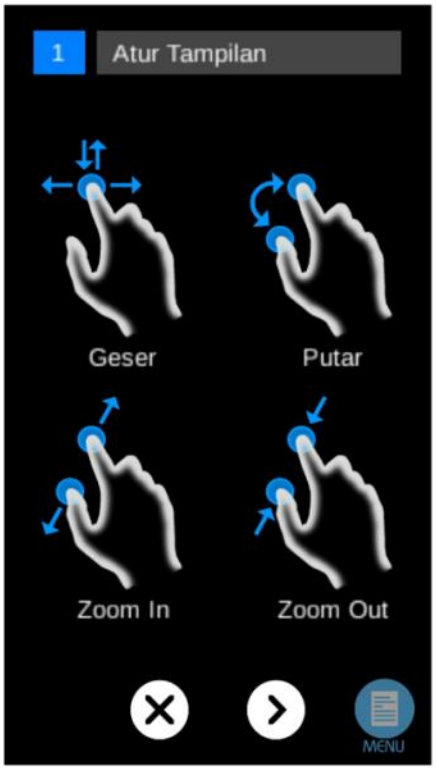
```
        if (hit.collider.gameObject == portI [0].gameObject)
        { insAB1 = true; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portI [1].gameObject)
        { insAB3 = true; insAB1 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portI [2].gameObject)
        { insAB5 = true; insAB1 = false; insAB3 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portI [3].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portI [4].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portI [5].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portI [6].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }

        //Pasang Output IC ke LED.
        if (hit.collider.gameObject == portO [0].gameObject)
        { insOP2 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portO [1].gameObject)
        { insOP4 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP6 = false; }
        if (hit.collider.gameObject == portO [2].gameObject)
        { insOP6 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; }
        if (hit.collider.gameObject == portO [3].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portO [4].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portO [5].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }
        if (hit.collider.gameObject == portO [6].gameObject)
        { zonk = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false; }

        //Lepas Kabel Input IC ke Saklar A.
        if (hit.collider.gameObject == kabelIA [0].gameObject)
        { insYT1 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false; insYT7 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }
        if (hit.collider.gameObject == kabelIA [1].gameObject)
        { insYT2 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false; insYT7 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }
        if (hit.collider.gameObject == kabelIA [2].gameObject)
        { insYT3 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT2 = false; insYT4 = false; insYT5 = false; insYT6 = false; insYT7 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }

        //Lepas Kabel Input IC ke Saklar B.
        if (hit.collider.gameObject == kabelIB [0].gameObject)
        { insYT4 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT2 = false; insYT3 = false; insYT5 = false; insYT6 = false; insYT7 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }
        if (hit.collider.gameObject == kabelIB [1].gameObject)
        { insYT5 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT6 = false; insYT7 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }
        if (hit.collider.gameObject == kabelIB [2].gameObject)
        { insYT6 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT7 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }

        //Lepas Kabel Output IC ke LED.
        if (hit.collider.gameObject == kabelO [0].gameObject)
        { insYT7 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
          insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }
        if (hit.collider.gameObject == kabelO [1].gameObject)
        { insYT8 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
          insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
          insYT7 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
          insYT14 = false; insYT15 = false; }
```

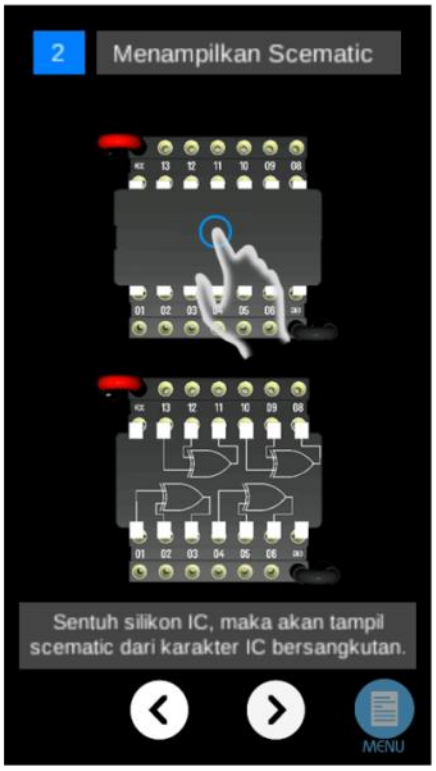
```
            if (hit.collider.gameObject == kabelO [2].gameObject)
            { insYT9 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT10 = false; insYT11 = false; insYT12 = false; insYT13 = false;
              insYT14 = false; insYT15 = false; }
            if (hit.collider.gameObject == kabelO [3].gameObject)
            { insYT10 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT9 = false; insYT11 = false; insYT12 = false; insYT13 = false;
              insYT14 = false; insYT15 = false; }
            if (hit.collider.gameObject == kabelO [4].gameObject)
            { insYT11 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT9 = false; insYT10 = false; insYT12 = false; insYT13 = false;
              insYT14 = false; insYT15 = false; }
            if (hit.collider.gameObject == kabelO [5].gameObject)
            { insYT12 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT13 = false;
              insYT14 = false; insYT15 = false; }
            if (hit.collider.gameObject == kabelO [6].gameObject)
            { insYT13 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false;
              insYT14 = false; insYT15 = false; }
            if (hit.collider.gameObject == kabelO [7].gameObject)
            { insYT14 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false;
              insYT13 = false; insYT15 = false; }
            if (hit.collider.gameObject == kabelO [8].gameObject)
            { insYT15 = true; insAB1 = false; insAB3 = false; insAB5 = false; insOP2 = false; insOP4 = false; insOP6 = false;
              insYT1 = false; insYT2 = false; insYT3 = false; insYT4 = false; insYT5 = false; insYT6 = false;
              insYT7 = false; insYT8 = false; insYT9 = false; insYT10 = false; insYT11 = false; insYT12 = false;
              insYT13 = false; insYT14 = false; }
        }
    }

    void OnGUI ()
    {   //      GUI.Label (new Rect (9, 133, 300, 22), label);
        if (insAB1)
        {   //Display Dialog konfirmasi muncul untuk Input saklar A&B Port1.
            GUI.skin = skineA;  GUI.Box (luasRect, "Pasang kabel konektor ?");
            insA1 = GUI.Button (new Rect ((Screen.width - 200) / 2, 40, 200, 30), "Sambung ke Input Saklar (A)");
            insB1 = GUI.Button (new Rect ((Screen.width - 200) / 2, 80, 200, 30), "Sambung ke Input Saklar (B)");
            if (insA1) { kabelIA [0].SetActive (true); insAB1 = false; }
            if (insB1) { kabelIB [0].SetActive (true); insAB1 = false; }
        }
        if (insAB3)
        {   //Display Dialog konfirmasi muncul untuk Input saklar A&B Port3.
            GUI.skin = skineA;  GUI.Box (luasRect, "Pasang kabel konektor ?");
            insA3 = GUI.Button (new Rect ((Screen.width - 200) / 2, 40, 200, 30), "Sambung ke Input Saklar (A)");
            insB3 = GUI.Button (new Rect ((Screen.width - 200) / 2, 80, 200, 30), "Sambung ke Input Saklar (B)");
            if (insA3) { kabelIA [1].SetActive (true); insAB3 = false; }
            if (insB3) { kabelIB [1].SetActive (true); insAB3 = false; }
        }
        if (insAB5)
        {   //Display Dialog konfirmasi muncul untuk Input saklar A&B Port5.
            GUI.skin = skineA;  GUI.Box (luasRect, "Pasang kabel konektor ?");
            insA5 = GUI.Button (new Rect ((Screen.width - 200) / 2, 40, 200, 30), "Sambung ke Input Saklar (A)");
            insB5 = GUI.Button (new Rect ((Screen.width - 200) / 2, 80, 200, 30), "Sambung ke Input Saklar (B)");
            if (insA5) { kabelIA [2].SetActive (true); insAB5 = false; }
            if (insB5) { kabelIB [2].SetActive (true); insAB5 = false; }
        }
        if (insOP2)
        {   //Display Dialog konfirmasi muncul untuk Output LED Port2.
            GUI.skin = skineA;  GUI.Box (new Rect ((Screen.width - 300) / 2, 9, 300, 160), "Pasang kabel konektor ?");
            insOP21 = GUI.Button (new Rect ((Screen.width - 200) / 2, 40, 200, 30), "Sambung ke Output LED (1)");
            insOP22 = GUI.Button (new Rect ((Screen.width - 200) / 2, 80, 200, 30), "Sambung ke Output LED (2)");
            insOP23 = GUI.Button (new Rect ((Screen.width - 200) / 2, 120, 200, 30), "Sambung ke Output LED (3)");
            if (insOP21) { kabelO [0].SetActive (true); insOP2 = false; }
            if (insOP22) { kabelO [1].SetActive (true); insOP2 = false; }
            if (insOP23) { kabelO [2].SetActive (true); insOP2 = false; }
        }
        if (insOP4)
        {   //Display Dialog konfirmasi muncul untuk Output LED Port4.
```

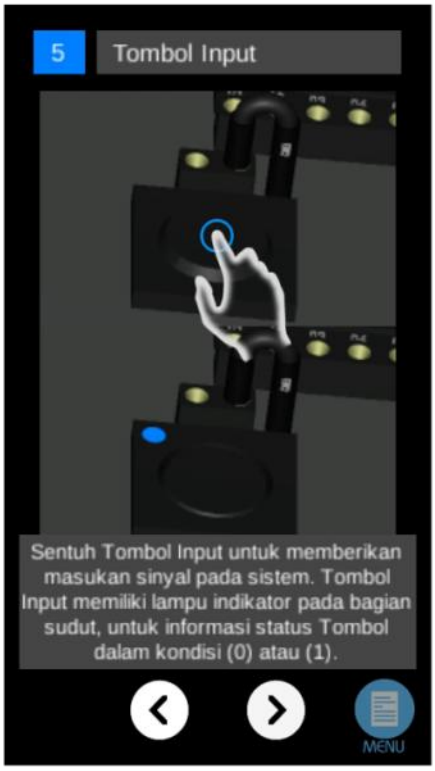
```
            GUI.skin = skineA;
            GUI.Box (new Rect ((Screen.width - 300) / 2, 9, 300, 160), "Pasang kabel konektor ?");
            insOP41 = GUI.Button (new Rect ((Screen.width - 200) / 2, 40, 200, 30), "Sambung ke Output LED (1)");
            insOP42 = GUI.Button (new Rect ((Screen.width - 200) / 2, 80, 200, 30), "Sambung ke Output LED (2)");
            insOP43 = GUI.Button (new Rect ((Screen.width - 200) / 2, 120, 200, 30), "Sambung ke Output LED (3)");
            if (insOP41) { kabelO [0].SetActive (true); insOP4 = false; }
            if (insOP42) { kabelO [1].SetActive (true); insOP4 = false; }
            if (insOP43) { kabelO [2].SetActive (true); insOP4 = false; }
        }
        if (insOP6)
        {   //Display Dialog konfirmasi muncul untuk Output LED Port6.
            GUI.skin = skineA;
            GUI.Box (new Rect ((Screen.width - 300) / 2, 9, 300, 160), "Pasang kabel konektor ?");
            insOP61 = GUI.Button (new Rect ((Screen.width - 200) / 2, 40, 200, 30), "Sambung ke Output LED (1)");
            insOP62 = GUI.Button (new Rect ((Screen.width - 200) / 2, 80, 200, 30), "Sambung ke Output LED (2)");
            insOP63 = GUI.Button (new Rect ((Screen.width - 200) / 2, 120, 200, 30), "Sambung ke Output LED (3)");
            if (insOP61) { kabelO [0].SetActive (true); insOP6 = false; }
            if (insOP62) { kabelO [1].SetActive (true); insOP6 = false; }
            if (insOP63) { kabelO [2].SetActive (true); insOP6 = false; }
        }
        //Display Dialog konfirmasi lepas kabel.
        if (insYT1)
        {   GUI.skin = skineB; //Memanggil skin GUISkin.
            GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY1A = GUI.Button (new Rect (50, 40, 77, 30), "Ya");  insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY1A) { kabelIA [0].SetActive (false); insYT1 = false; }  if (insT) { insYT1 = false; }
        }
        if (insYT2)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY3A = GUI.Button (new Rect (50, 40, 77, 30), "Ya");  insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY3A) { kabelIA [1].SetActive (false); insYT2 = false; }  if (insT) { insYT2 = false; }
        }
        if (insYT3)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY5A = GUI.Button (new Rect (50, 40, 77, 30), "Ya");  insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY5A) { kabelIA [2].SetActive (false); insYT3 = false; }  if (insT) { insYT3 = false; }
        }
        if (insYT4)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY1B = GUI.Button (new Rect (50, 40, 77, 30), "Ya");  insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY1B) { kabelIB [0].SetActive (false); insYT4 = false; }  if (insT) { insYT4 = false; }
        }
        if (insYT5)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY3B = GUI.Button (new Rect (50, 40, 77, 30), "Ya");  insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY3B) { kabelIB [1].SetActive (false); insYT5 = false; }  if (insT) { insYT5 = false; }
        }
        if (insYT6)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY5B = GUI.Button (new Rect (50, 40, 77, 30), "Ya");  insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY5B) { kabelIB [2].SetActive (false); insYT6 = false; }  if (insT) { insYT6 = false; }
        }
        if (insYT7)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY21O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY21O) { kabelO [0].SetActive (false); insYT7 = false; }  if (insT) { insYT7 = false; }
        }
        if (insYT8)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY22O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY22O) { kabelO [1].SetActive (false); insYT8 = false; }  if (insT) { insYT8 = false; }
        }
        if (insYT9)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY23O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY23O) { kabelO [2].SetActive (false); insYT9 = false; }  if (insT) { insYT9 = false; }
        }
        if (insYT10)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY41O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY41O) { kabelO [3].SetActive (false); insYT10 = false; } if (insT) { insYT10 = false; }
        }
        if (insYT11)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY42O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY42O) { kabelO [4].SetActive (false); insYT11 = false; } if (insT) { insYT11 = false; }
```

```
        if (insYT11)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY42O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY42O) { kabelO [4].SetActive (false); insYT11 = false; } if (insT) { insYT11 = false; }
        }
        if (insYT12)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY43O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY43O) { kabelO [5].SetActive (false); insYT12 = false; } if (insT) { insYT12 = false; }
        }
        if (insYT13)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY61O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY61O) { kabelO [6].SetActive (false); insYT13 = false; } if (insT) { insYT13 = false; }
        }
        if (insYT14)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY62O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY1A) { kabelO [7].SetActive (false); insYT14 = false; }  if (insT) { insYT14 = false; }
        }
        if (insYT15)
        {   GUI.skin = skineB;  GUI.Box (luasRect, "Lepas kabel konektor ?");
            insY63O = GUI.Button (new Rect (50, 40, 77, 30), "Ya"); insT = GUI.Button (new Rect (150, 40, 77, 30), "Tidak");
            if (insY63O) { kabelO [8].SetActive (false); insYT15 = false; } if (insT) { insYT15 = false; }
        }
        if (zonk)
        {   //Display Dialog konfirmasi muncul untuk zonk.
            GUI.skin = skineC;
            GUI.Box (luasRect, "Fitur yang Anda pilih\n belum tersedia dalam Versi ini.\n Silahkan pilih Port yang lain.\n");
            zonkY = GUI.Button (new Rect ((Screen.width - 30) / 2, 80, 30, 30), "Ya");  if (zonkY) { zonk = false; }
        }
    }
}
```

Lampiran 7. Fitur Petunjuk Pengoperasian






5
Tombol Input
Sentuh Tombol Input untuk memberikan masukan sinyal pada sistem. Tombol Input memiliki lampu indikator pada bagian sudut, untuk informasi status Tombol dalam kondisi (0) atau (1).
MENU


6
LED Output
Komponen LED Output sebagai keluaran dari sistem Gerbang Logika dasar (Logic Gate) ini. Status Output (1) maka LED akan menyala. Status Output (0) maka LED akan mati.
MENU

Lampiran 8. Uji Instalasi pada Perangkat Siswa

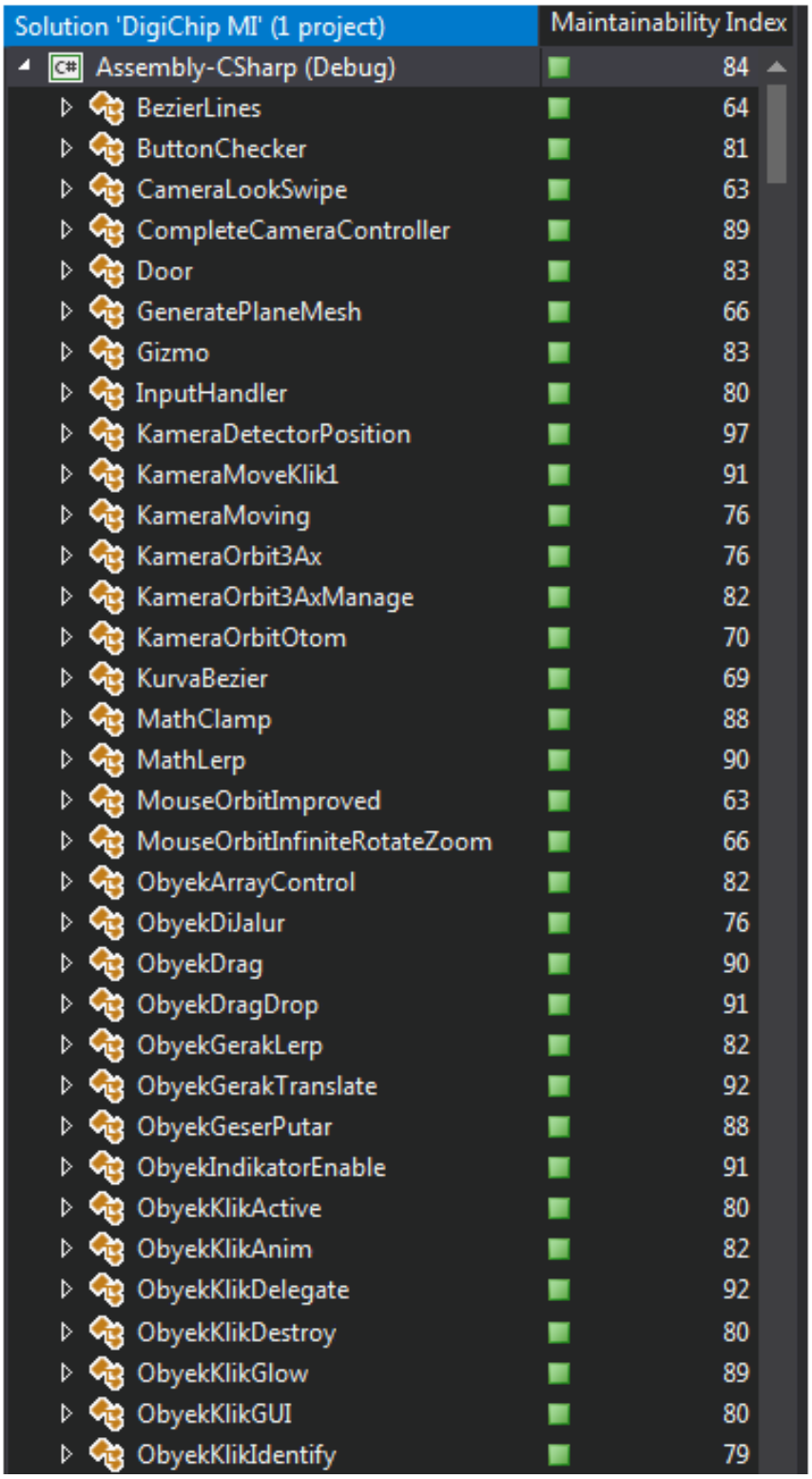
OS Kernel 4.4 Kit Kat

OS Kernel 5.0 Lolipop

OS Kernel 5.1 Lolipop

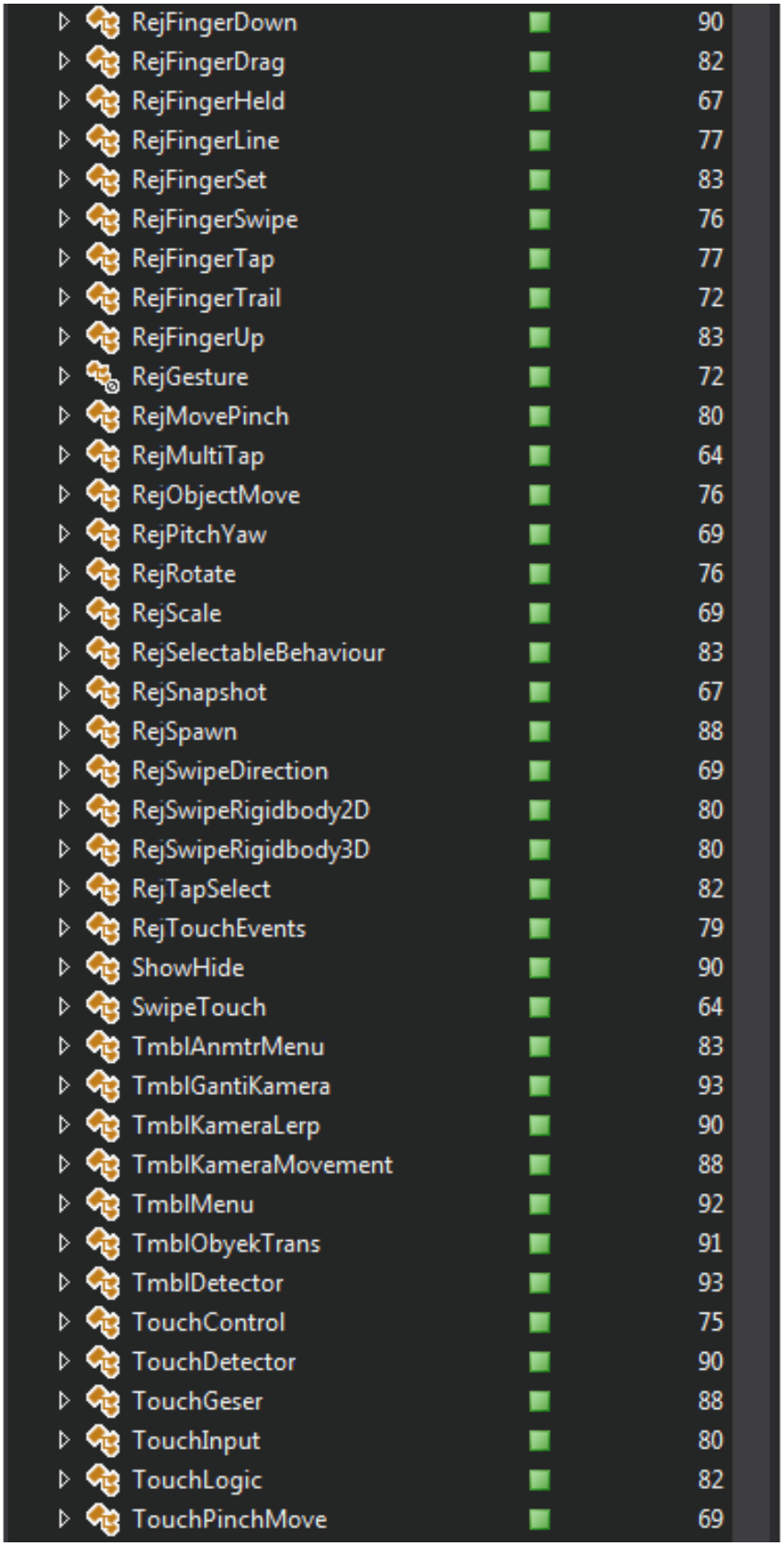
OS Kernel 6.0 Marsmallow

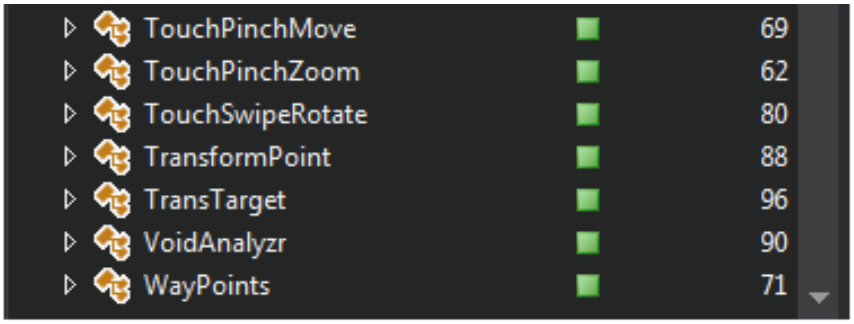
Lampiran 9. Uji *Maintainability* ISO/IEC 25010 - *Maintainability Index (MI)*



Solution 'DigiChip MI' (1 project)	Maintainability Index
Assembly-CSharp (Debug)	84
BezierLines	64
ButtonChecker	81
CameraLookSwipe	63
CompleteCameraController	89
Door	83
GeneratePlaneMesh	66
Gizmo	83
InputHandler	80
KameraDetectorPosition	97
KameraMoveKlik1	91
KameraMoving	76
KameraOrbit3Ax	76
KameraOrbit3AxManage	82
KameraOrbitOtom	70
KurvaBezier	69
MathClamp	88
MathLerp	90
MouseOrbitImproved	63
MouseOrbitInfiniteRotateZoom	66
ObyekArrayControl	82
ObyekDiJalur	76
ObyekDrag	90
ObyekDragDrop	91
ObyekGerakLerp	82
ObyekGerakTranslate	92
ObyekGeserPutar	88
ObyekIndikatorEnable	91
ObyekKlikActive	80
ObyekKlikAnim	82
ObyekKlikDelegate	92
ObyekKlikDestroy	80
ObyekKlikGlow	89
ObyekKlikGUI	80
ObyekKlikIdentify	79


ObyekKlikIdentify 79
ObyekKlikInactive 81
ObyekKlikLampu 89
ObyekKlikLerp 81
ObyekKlikPasangLepas 30
ObyekKlikPortANDIn 81
ObyekKlikPortANDOut 74
ObyekKlikPortNANDIn 81
ObyekKlikPortNANDOut 70
ObyekKlikPortNORIn 81
ObyekKlikPortNOROut 70
ObyekKlikPortNOTIn 81
ObyekKlikPortNOTOut 61
ObyekKlikPortORIn 81
ObyekKlikPortOROut 70
ObyekKlikPortXORIn 81
ObyekKlikPortXOROut 70
ObyekKlikPortZonk 82
ObyekKlikTrans 94
ObyekKlikWarna 72
ObyekLerper 80
ObyekLook 57
ObyekMouseShade 89
ObyekMoveTouch 87
ObyekMuter 57
ObyekPutarBebas 80
ObyekReposisi 82
ObyekSlerper 71
ObyekUpDown 88
OnMousedown 92
OnMousedownPlayer 89
OnPointerdown 90
Path 94
PathAgent 72
PlayAudio 95
RejCameraZoom 70
RejDestroy 91
RejFinger 76
RejFingerDown 90


RejFingerDown 90
RejFingerDrag 82
RejFingerHeld 67
RejFingerLine 77
RejFingerSet 83
RejFingerSwipe 76
RejFingerTap 77
RejFingerTrail 72
RejFingerUp 83
RejGesture 72
RejMovePinch 80
RejMultiTap 64
RejObjectMove 76
RejPitchYaw 69
RejRotate 76
RejScale 69
RejSelectableBehaviour 83
RejSnapshot 67
RejSpawn 88
RejSwipeDirection 69
RejSwipeRigidbody2D 80
RejSwipeRigidbody3D 80
RejTapSelect 82
RejTouchEvents 79
ShowHide 90
SwipeTouch 64
TmblAnmtrMenu 83
TmblGantiKamera 93
TmblKameraLerp 90
TmblKameraMovement 88
TmblMenu 92
TmblObyekTrans 91
TmblDetector 93
TouchControl 75
TouchDetector 90
TouchGeser 88
TouchInput 80
TouchLogic 82
TouchPinchMove 69


TouchPinchMove
69
TouchPinchZoom
62
TouchSwipeRotate
80
TransformPoint
88
TransTarget
96
VoidAnalyzr
90
WayPoints
71

Lampiran 10. Hasil Uji *Usability*

Responden	Butir Soal																												Jumlah
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	
1	4	4	5	3	4	2	2	3	5	5	4	5	3	5	4	3	5	5	4	5	5	3	3	3	2	3	4	5	108
2	4	3	4	3	4	3	4	3	3	3	4	4	3	4	3	3	3	3	4	4	4	3	4	4	3	3	4	3	97
3	5	4	4	5	5	5	4	3	5	5	4	4	3	2	4	4	4	4	5	4	4	5	4	4	3	4	4	4	115
4	5	4	5	4	4	5	4	4	5	5	5	5	4	5	4	5	5	5	5	5	5	5	5	5	4	4	4	3	128
5	3	3	4	4	3	4	4	4	4	4	4	4	4	4	3	4	4	4	4	4	4	4	3	4	4	4	4	4	107
6	4	4	4	4	4	5	5	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	4	4	4	4	4	4	128
7	5	5	5	5	5	4	5	4	5	5	5	5	5	4	4	5	5	4	5	5	5	4	4	5	4	4	5	4	130
8	5	5	4	4	4	4	5	5	5	4	5	5	4	4	4	4	4	4	5	5	4	5	4	4	4	4	4	5	124
9	4	5	4	4	4	4	4	4	5	5	4	4	5	4	4	4	5	4	5	5	5	4	5	5	4	5	4	4	123
10	4	4	5	4	4	5	4	5	5	4	5	5	5	4	4	5	4	4	5	4	5	5	4	5	5	5	4	4	126
11	4	4	5	4	4	2	5	4	4	5	4	4	3	4	3	5	5	4	5	4	4	5	4	4	2	4	5	4	114
12	4	5	4	2	4	5	3	5	4	4	3	4	5	2	4	3	4	3	5	3	4	4	2	5	4	3	4	4	106
13	5	4	4	5	5	5	5	4	4	5	5	4	4	5	5	5	4	4	4	4	4	5	4	5	4	5	4	5	126
14	4	5	5	5	4	4	5	5	4	5	4	5	4	4	5	5	4	4	5	5	4	4	4	4	4	4	4	4	123
15	3	4	5	5	4	4	5	5	4	5	5	5	5	4	4	5	5	5	4	4	5	5	5	5	4	5	4	4	127
16	4	4	5	4	5	4	2	5	3	5	5	4	2	4	2	4	5	4	5	4	4	3	3	4	4	5	4	4	111
17	4	4	5	2	4	3	4	5	3	4	3	4	3	4	4	4	3	5	4	4	2	4	3	3	4	4	3	4	103
18	4	5	5	4	5	4	4	4	4	5	5	5	4	4	5	4	4	4	4	4	4	5	5	5	4	4	4	5	123
19	4	5	5	4	4	5	5	4	5	5	5	5	4	5	4	4	4	4	5	4	5	5	4	4	5	5	4	5	127
20	5	3	5	5	3	3	5	4	4	4	3	4	5	2	5	3	4	4	5	4	5	4	2	4	3	4	2	4	108
21	5	4	5	5	4	4	4	5	5	5	5	4	4	4	4	4	4	5	5	4	4	5	4	4	4	5	5	4	124
22	5	3	5	3	4	4	5	5	4	5	5	5	4	5	4	5	5	4	4	2	4	3	3	5	4	2	4	4	115
23	3	3	5	4	4	5	4	2	5	4	5	3	3	2	4	4	4	2	5	4	4	5	4	2	4	4	3	3	104
24	4	5	4	3	3	4	5	5	4	5	4	5	5	3	4	4	5	4	4	5	5	5	4	4	4	3	4	5	119
25	4	3	4	5	3	4	4	4	5	4	5	4	4	4	4	5	5	3	5	4	5	4	4	4	4	5	3	5	117
26	5	5	5	2	3	3	4	4	2	5	5	3	3	4	4	4	2	5	5	4	4	3	4	2	4	4	4	5	107
27	5	4	5	4	4	4	4	5	5	4	5	5	5	4	4	5	4	4	4	5	5	4	4	5	4	5	4	5	125
28	5	4	5	4	3	3	5	4	5	5	4	5	3	4	4	2	4	5	5	5	4	5	3	4	2	4	3	4	113
29	5	4	5	4	3	3	3	5	5	5	5	5	5	4	3	4	5	5	5	4	5	3	4	4	5	5	4	4	121
30	4	5	2	4	3	5	4	3	3	5	5	5	4	4	3	5	4	4	5	4	4	5	3	2	4	5	3	5	112
31	4	5	2	3	4	4	5	4	4	3	2	4	4	5	4	5	3	4	5	4	5	4	5	2	2	3	4	3	106
32	4	4	3	3	4	2	4	4	5	5	4	5	3	5	4	4	2	4	4	5	5	2	5	5	3	5	5	5	113
33	5	4	5	2	3	3	5	5	3	3	5	5	4	3	4	5	5	4	4	5	4	4	5	2	5	4	2	4	112
34	5	5	4	4	4	4	5	5	5	4	5	5	5	5	4	4	4	5	4	4	5	5	4	4	4	4	5	4	125
35	4	5	5	4	4	4	5	4	5	4	5	5	4	4	5	4	4	5	4	5	4	4	4	4	5	5	4	5	124
36	4	4	4	4	4	4	5	5	4	5	4	4	5	5	5	4	4	5	4	5	5	5	4	4	4	4	5	4	123
37	5	4	5	4	4	4	5	4	5	5	5	4	5	5	4	5	5	4	5	4	4	5	4	4	5	4	5	5	127
38	4	3	3	5	3	5	4	5	5	5	5	4	5	5	4	5	5	4	5	4	4	4	5	4	4	5	4	5	123
39	4	4	4	5	5	4	4	5	5	5	5	5	4	4	4	5	4	4	5	4	5	4	5	5	4	5	4	4	125
40	5	2	3	5	3	4	4	3	3	4	4	5	4	5	3	4	4	5	5	4	5	5	3	3	5	2	5	4	111
41	5	5	4	4	5	4	5	5	4	5	5	4	5	4	5	4	5	5	4	5	5	4	4	5	4	5	5	5	129
42	5	4	5	4	4	5	4	5	4	5	5	5	4	5	5	4	5	4	4	5	4	5	4	5	4	5	5	5	128
43	4	4	5	5	4	5	4	5	5	4	5	4	5	4	4	5	4	5	4	5	4	5	4	4	5	4	5	4	125
44	4	5	5	4	5	4	5	4	5	4	5	4	4	5	4	4	5	4	4	5	5	4	4	5	4	5	4	5	125
45	5	4	5	4	5	4	5	4	5	4	5	5	4	5	4	4	5	4	5	4	5	4	4	5	4	5	4	4	126
46	5	4	5	4	5	4	5	5	4	5	5	5	4	4	5	4	5	5	5	4	5	5	4	5	4	4	4	5	128
47	4	5	5	4	4	5	4	5	4	5	4	5	5	4	5	4	5	4	5	4	5	5	4	4	5	5	4	4	126
48	4	4	5	4	4	5	5	4	5	5	4	5	4	4	5	5	5	4	5	4	5	4	5	4	5	4	4	5	126
49	5	4	5	4	5	5	5	5	5	4	5	5	4	4	5	5	4	5	4	4	5	5	4	5	4	5	5	4	129
50	5	5	5	4	4	5	5	4	4	5	5	5	4	4	4	5	5	4	5	5	4	4	5	5	4	4	5	4	127
51	4	4	5	5	5	4	5	5	5	4	5	5	5	5	4	4	5	5	4	5	5	4	4	5	5	5	5	5	131
52	5	4	5	4	4	4	5	5	4	4	5	5	5	4	4	5	4	4	5	5	4	5	5	4	4	5	4	5	126
53	5	5	4	4	5	4	5	5	5	4	5	5	5	5	5	4	5	5	5	4	5	5	5	5	4	4	4	4	130
54	5	4	5	4	4	4	5	4	5	5	5	5	5	4	4	5	5	5	5	5	4	4	4	5	5	5	5	4	129
55	5	5	5	4	4	5	4	5	5	4	4	5	4	5	5	5	4	4	4	5	5	4	5	5	4	4	4	5	127
56	4	5	5	5	4	4	4	4	5	5	5	5	5	4	5	5	5	5	5	5	4	4	5	4	5	4	5	5	130
57	5	4	5	4	5	4	4	4	5	5	5	4	5	5	5	5	4	4	4	5	5	4	4	5	5	5	4	4	127
58	4	4	5	4	4	5	5	5	5	5	4	4	4	4	5	5	5	5	4	4	5	4	4	4	4	5	5	5	126
59	5	4	5	4	5	4	4	4	4	5	4	5	5	5	4	4	5	4	5	4	4	5	4	4	5	5	4	4	124
JUMLAH																													7,119

Lampiran 11. Hasil Uji *Reliability*

```
RELIABILITY
  /VARIABLES=x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 x15 x16 x17 x18 x19 x20 x21 x22 x23 x24 x25 x26 x27 x28
  /SCALE('ALL VARIABLES') ALL
  /MODEL=ALPHA
  /STATISTICS=DESCRIPTIVE SCALE.
```

Scale: ALL VARIABLES

Case Processing Summary

		N	%
Cases	Valid	59	100.0

a. Listwise deletion based on all variables in the procedure.

Case Processing Summary

		N	%
Cases	Excluded[a]	0	.0
	Total	59	100.0

a. Listwise deletion based on all variables in the procedure.

Reliability Statistics

Cronbach's Alpha	N of Items
.841	28

Item Statistics

	Mean	Std. Deviation	N
x1	4.42	.593	59
x2	4.19	.706	59
x3	4.54	.750	59
x4	4.00	.788	59
x5	4.07	.666	59
x6	4.08	.794	59
x7	4.41	.722	59
x8	4.37	.717	59
x9	4.42	.747	59
x10	4.56	.595	59
x11	4.56	.676	59
x12	4.58	.563	59
x13	4.20	.761	59
x14	4.19	.798	59
x15	4.17	.673	59
x16	4.36	.689	59
x17	4.37	.740	59
x18	4.29	.645	59
x19	4.59	.495	59
x20	4.37	.613	59
x21	4.49	.598	59
x22	4.34	.734	59
x23	4.05	.729	59
x24	4.17	.874	59
x25	4.07	.785	59
x26	4.29	.767	59
x27	4.17	.723	59
x28	4.34	.605	59

Scale Statistics

Mean	Variance	Std. Deviation	N of Items
120.66	73.297	8.561	28

Lampiran 12. Instrumen Ahli Media

Instrumen Pengujian Ahli Media
Pada Software Simulasi "DigiChip"

Nama : _____

Bidang Keahlian : _____

Institusi : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;

(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	STS
1	Penggunaan warna teks terlihat sudah proporsional					
2	Penggunaan warna angka terlihat sudah proporsional					
3	Penggunaan warna background sudah proporsional					
4	Kekontrasan antara foreground dan background					
5	Gambar dapat terlihat dengan jelas					
6	Ukuran gambar sudah proporsional					
7	Penempatan posisi gambar sudah proporsional					
8	Huruf dapat terlihat jelas					
9	Ukuran huruf sudah proporsional					
10	Angka dapat terlihat jelas					
11	Ukuran angka sudah proporsional					
12	Ukuran animasi gambar sudah proporsional					
13	Ukuran animasi tombol sudah proporsional					
14	Efek animasi terlihat artistik					

Saran dan masukan :

Lampiran 13. Instrumen Ahli Materi

Instrumen Pengujian Ahli Materi
Pada Software Simulasi "DigiChip"

Nama : _____

Bidang Keahlian : _____

Institusi : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip" .

No.	Pernyataan	Jawaban	
		Ya	Tidak
1	Hasil simulasi output terhadap input pada *logic gate* NOT sudah benar		
2	Hasil simulasi output terhadap input pada *logic gate* AND sudah benar		
3	Hasil simulasi output terhadap input pada *logic gate* NAND sudah benar		
4	Hasil simulasi output terhadap input pada *logic gate* OR sudah benar		
5	Hasil simulasi output terhadap input pada *logic gate* NOR sudah benar		
6	Hasil simulasi output terhadap input pada *logic gate* XOR sudah benar		
7	Simbol *logic gate* NOT sudah benar		
8	Simbol *logic gate* AND sudah benar		
9	Simbol *logic gate* NAND sudah benar		
10	Simbol *logic gate* OR sudah benar		
11	Simbol *logic gate* NOR sudah benar		
12	Simbol *logic gate* XOR sudah benar		
13	Penulisan rumus Aljabar Boolean sudah benar		
14	Gambar skematik IC NOT sudah benar		
15	Gambar skematik IC AND sudah benar		
16	Gambar skematik IC NAND sudah benar		
17	Gambar skematik IC OR sudah benar		
18	Gambar skematik IC NOR sudah benar		
19	Gambar skematik IC XOR sudah benar		

Saran dan masukan :

_____

_____

_____

_____

_____

_____

_____

_____

Lampiran 14. Instrumen *Usability (USE Questionnaire)*

Instrumen Pengujian Aspek Usability
Pada Software Simulasi "DigiChip"

Nama : _____

Kelas : _____

Tipe Ponsel : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	STS
1	Aplikasi ini membantu saya menjadi lebih efektif dalam pembelajaran teknik digital.					
2	Aplikasi ini membantu saya menjadi lebih produktif dalam pembelajaran teknik digital.					
3	Aplikasi ini bermanfaat dalam pembelajaran teknik digital.					
4	Aplikasi ini memberikan dampak yang lebih dalam pembelajaran teknik digital yang saya lakukan.					
5	Aplikasi ini memberikan saya kemudahan dalam menyelesaikan persoalan dalam pembelajaran teknik digital.					
6	Aplikasi ini mampu menghemat waktu saya dalam memahami pembelajaran teknik digital.					
7	Aplikasi ini sesuai dengan kebutuhan saya.					
8	Aplikasi ini bekerja sesuai dengan harapan saya.					
9	Aplikasi ini mudah digunakan.					
10	Aplikasi ini praktis untuk digunakan.					
11	Aplikasi ini mudah untuk dipahami (user friendly).					
12	Aplikasi ini menggunakan langkah-langkah yang mudah dan sederhana.					
13	Aplikasi ini dapat sesuai dengan kebutuhan saya.					

14	Saya dapat menggunakan aplikasi ini tanpa panduan tertulis.					
15	Saya tidak menemukan ketidakkonsistenan selama saya menggunakan aplikasi ini.					
16	Penggunaan yang jarang ataupun rutin saya menyukai aplikasi ini.					
17	Kapanpun saya melakukan kesalahan menggunakan aplikasi ini saya dapat kembali dengan cepat dan mudah.					
18	Saya dapat menggunakan aplikasi ini dengan baik setiap waktu.					
19	Saya memahami penggunaan aplikasi ini dengan cepat.					
20	Saya dapat dengan mudah mengingat bagaimana cara penggunaan aplikasi ini.					
21	Saya dengan cepat mahir menggunakan aplikasi ini.					
22	Saya merasa puas dengan kinerja aplikasi ini.					
23	Saya akan merekomendasikan aplikasi ini ke teman saya.					
24	Penggunaan aplikasi ini menyenangkan.					
25	Aplikasi ini bekerja seperti apa yang saya inginkan.					
26	Aplikasi ini sangat bagus.					
27	Saya merasa saya harus memiliki aplikasi ini. ini					
28	Aplikasi ini nyaman untuk digunakan.					

Saran dan masukan :

_____

_____

_____

_____

_____

_____

_____

_____

Lampiran 15. Validasi Instrumen Penelitian oleh Ahli

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS NEGERI YOGYAKARTA
PROGRAM PASCASARJANA
Jalan Colombo Nomor 1 Yogyakarta 55281
Telepon (0274) 550836 pesawat 229, Fax (0274) 520326
Laman: pps.uny.ac.id E-mail: pps@uny.ac.id

Nomor : 7078 /UN34.17/LT/2017 9 Agustus 2017
Hal : Izin Validasi

Yth. Prof. Herman Dwi Surjono, Ph.D.
Dosen Universitas Negeri Yogyakarta

Kami mohon dengan hormat, Bapak/Ibu bersedia menjadi validator instrumen penelitian bagi mahasiswa:

Nama : Titih Rejyasmito Hadi
No. Mahasiswa : 14702251086
Prodi : Pendidikan Teknologi dan Kejuruan
Pembimbing : Dr. Eko Marpanaji
Judul : Perancangan dan Analisis Kualitas Software Simulasi DIGICHIP Platform Android Berbasis Mobile Learning untuk Mata Pelajaran Teknik Digital pada Siswa Teknik Mekatronika

Kami sangat mengharapkan Bapak/Ibu dapat mengembalikan hasil validasi paling lama 2 (dua) minggu. Atas kerjasama yang baik dari Bapak kami ucapkan terima kasih.

Asisten Direktur I,

Dr. Sugito, M.A.
NIP 19600410 198503 1 002

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS NEGERI YOGYAKARTA
PROGRAM PASCASARJANA
Jalan Colombo Nomor 1 Yogyakarta 55281
Telepon (0274) 550836 pesawat 229, Fax (0274) 520326
Laman: pps.uny.ac.id E-mail: pps@uny.ac.id

SURAT KETERANGAN VALIDASI

Yang bertanda tangan di bawah ini:

Nama : Prof. Herman Dwi Surjono, PhD.
Jabatan/Pekerjaan : Koprodi S2 TP
Instansi Asal : UNY

Menyatakan bahwa instrumen penelitian dengan judul:

Perancangan dan Analisis Kualitas Software Simulasi DIGICHIP Platform Android Berbasis Mobile Learning untuk Mata Pelajaran Teknik Digital pada Siswa Teknik Mekatronika

dari mahasiswa:

Nama : Titih Rejyasmito Hadi
Program Studi : Pendidikan Teknologi dan Kejuruan
NIM : 14702251086

(sudah siap/~~belum siap~~)* dipergunakan untuk penelitian dengan menambahkan beberapa saran sebagai berikut:

1. lihat catatan di instrument
2. .....

Demikian surat keterangan ini kami buat untuk dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 15 - 8 - 2017
Validator,

Herman Dwi S

*) coret yang tidak perlu

Instrumen Pengujian Ahli Materi
Pada Software Simulasi "DigiChip"

Nama : _____ Ttd :

Bidang Keahlian : _____

Institusi : _____ Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	TST
1	Hasil simulasi input terhadap output pada *logic gate* NOT sudah sesuai					
2	Hasil simulasi input terhadap output pada *logic gate* AND sudah sesuai					
3	Hasil simulasi input terhadap output pada *logic gate* NAND sudah sesuai					
4	Hasil simulasi input terhadap output pada *logic gate* OR sudah sesuai					
5	Hasil simulasi input terhadap output pada *logic gate* NOR sudah sesuai					
6	Hasil simulasi input terhadap output pada *logic gate* XOR sudah sesuai					
7	Simbol *logic gate* NOT sudah sesuai					
8	Simbol *logic gate* AND sudah sesuai					
9	Simbol *logic gate* NAND sudah sesuai					
10	Simbol *logic gate* OR sudah sesuai					
11	Simbol *logic gate* NOR sudah sesuai					
12	Simbol *logic gate* XOR sudah sesuai					
13	Penulisan rumus Aljabar Boolean sudah sesuai					
14	Gambar skematik IC NOT sudah sesuai					

15	Gambar skematik IC AND sudah sesuai					
16	Gambar skematik IC NAND sudah sesuai					
17	Gambar skematik IC OR sudah sesuai					
18	Gambar skematik IC NOR sudah sesuai					
19	Gambar skematik IC XOR sudah sesuai					

Saran dan masukan :

Instrumen Pengujian Ahli Media
Pada Software Simulasi "DigiChip"

Nama : _____

Bidang Keahlian : _____

Institusi : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	TST
1	Penggunaan warna teks terlihat sudah sesuai					
2	Penggunaan warna angka terlihat sudah sesuai					
3	Penggunaan warna background sudah sesuai					
4	Gambar dapat terlihat dengan jelas					
5	Ukuran gambar sudah sesuai					
6	Penempatan posisi gambar sudah sesuai					
7	Huruf dapat terlihat jelas					
8	Ukuran huruf sudah sesuai					
9	Angka dapat terlihat jelas					
10	Ukuran angka sudah sesuai					
11	Ukuran animasi gambar sudah sesuai					
12	Ukuran animasi tombol sudah sesuai					
13	Efek animasi terlihat artistik					

Kekontrasan antara foreground dan background

Saran dan masukan :

Instrumen Pengujian Aspek Usability
Pada Software Simulasi "DigiChip"

Nama : _____

Kelas : _____

Tipe Ponsel : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	TST
1	Aplikasi ini membantu saya menjadi lebih efektif dalam pembelajaran teknik digital.					
2	Aplikasi ini membantu saya menjadi lebih produktif dalam pembelajaran teknik digital.					
3	Aplikasi ini bermanfaat dalam pembelajaran teknik digital.					
4	Aplikasi ini memberikan dampak yang lebih dalam pembelajaran teknik digital yang saya lakukan.					
5	Aplikasi ini memberikan saya kemudahan dalam menyelesaikan persoalan dalam pembelajaran teknik digital.					
6	Aplikasi ini mampu menghemat waktu saya dalam memahami pembelajaran teknik digital.					
7	Aplikasi ini sesuai dengan kebutuhan saya.					
8	Aplikasi ini bekerja sesuai dengan harapan saya.					
9	Aplikasi ini mudah digunakan.					
10	Aplikasi ini praktis untuk digunakan.					
11	Aplikasi ini mudah untuk dipahami (user friendly).					
12	Aplikasi ini menggunakan langkah-langkah yang mudah dan sederhana.					
13	Aplikasi ini dapat sesuai dengan kebutuhan saya.					

14	Saya tidak kesulitan dalam menggunakan aplikasi ini.					
15	Saya dapat menggunakan aplikasi ini tanpa panduan tertulis.					
16	Saya tidak menemukan ketidakkonsistenan selama saya menggunakan aplikasi ini.					
17	Penggunaan yang jarang ataupun rutin saya menyukai aplikasi ini. → rev.					
18	Kapanpun saya melakukan kesalahan menggunakan aplikasi ini saya dapat kembali dengan cepat dan mudah.					
19	Saya dapat menggunakan aplikasi ini dengan baik setiap waktu.					
20	Saya memahami penggunaan aplikasi ini dengan cepat.					
21	Saya dapat dengan mudah mengingat bagaimana cara penggunaan aplikasi ini.					
22	Sangat mudah untuk memahami cara penggunaan aplikasi ini.					
23	Saya dengan cepat mahir menggunakan aplikasi ini.					
24	Saya merasa puas dengan kinerja aplikasi ini.					
25	Saya akan merekomendasikan aplikasi ini ke teman saya.					
26	Penggunaan aplikasi ini menyenangkan.					
27	Aplikasi ini bekerja seperti apa yang saya inginkan.					
28	Aplikasi ini sangat bagus.					
29	Saya merasa saya harus memiliki aplikasi ini. ini					
30	Aplikasi ini nyaman untuk digunakan.					

Saran dan masukan :

no 9 & 14 sama
22

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS NEGERI YOGYAKARTA
PROGRAM PASCASARJANA
Jalan Colombo Nomor 1 Yogyakarta 55281
Telepon (0274) 550836 pesawat 229, Fax (0274) 520326
Laman: pps.uny.ac.id E-mail: pps@uny.ac.id

Nomor : 7047 /UN34.17/LT/2017 8 Agustus 2017
Hal : Izin Validasi

Yth. Dr. Ratna Wardani
Dosen Universitas Negeri Yogyakarta

Kami mohon dengan hormat, Bapak/Ibu bersedia menjadi validator instrumen penelitian bagi mahasiswa:

Nama : Titih Rejyasmito Hadi
No. Mahasiswa : 14702251086
Prodi : Pendidikan Teknologi dan Kejuruan
Pembimbing : Dr. Eko Marpanaji
Judul : Perancangan dan Analisis Kualitas Software Simulasi DIGICHIP Platform Android Berbasis Mobile Learning untuk Mata Pelajaran Teknik Digital pada Siswa Teknik Mekatronika

Kami sangat mengharapkan Bapak/Ibu dapat mengembalikan hasil validasi paling lama 2 (dua) minggu. Atas kerjasama yang baik dari Bapak kami ucapkan terima kasih.

Asisten Direktur I,

Dr. Sugito, M.A.
NIP 19600410 198503 1 002

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS NEGERI YOGYAKARTA
PROGRAM PASCASARJANA
Jalan Colombo Nomor 1 Yogyakarta 55281
Telepon (0274) 550836 pesawat 229, Fax (0274) 520326
Laman: pps.uny.ac.id E-mail: pps@uny.ac.id

SURAT KETERANGAN VALIDASI

Yang bertanda tangan di bawah ini:

Nama : Dr. Ratna Wardani, S.Si. M.T.
Jabatan/Pekerjaan : Dosen.
Instansi Asal : UNY

Menyatakan bahwa instrumen penelitian dengan judul:
Perancangan dan Analisis Kualitas Software Simulasi DIGICHIP Platform Android Berbasis Mobile Learning untuk Mata Pelajaran Teknik Digital pada Siswa Teknik Mekatronika
dari mahasiswa:

Nama : Titih Rejyasmito Hadi
Program Studi : Pendidikan Teknologi dan Kejuruan
NIM : 14702251086

(sudah siap/belum siap)* dipergunakan untuk penelitian dengan menambahkan beberapa saran sebagai berikut:

1. Editing istilah.
2. Yg instrumen Usability, kalau sudah ada yg standar, gunakan yg standar saja.

Demikian surat keterangan ini kami buat untuk dapat dipergunakan sebagaimana mestinya.

Yogyakarta,..... 2017
Validator,

Dr. Ratna Wardani

*) coret yang tidak perlu

Instrumen Pengujian Ahli Materi
Pada Software Simulasi "DigiChip"

Nama : _____

Bidang Keahlian : _____

Institusi : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	TST
1	Hasil simulasi input terhadap output pada *logic gate* NOT sudah sesuai					
2	Hasil simulasi input terhadap output pada *logic gate* AND sudah sesuai					
3	Hasil simulasi input terhadap output pada *logic gate* NAND sudah sesuai					
4	Hasil simulasi input terhadap output pada *logic gate* OR sudah sesuai					
5	Hasil simulasi input terhadap output pada *logic gate* NOR sudah sesuai					
6	Hasil simulasi input terhadap output pada *logic gate* XOR sudah sesuai					
7	Simbol *logic gate* NOT sudah sesuai					
8	Simbol *logic gate* AND sudah sesuai					
9	Simbol *logic gate* NAND sudah sesuai					
10	Simbol *logic gate* OR sudah sesuai					
11	Simbol *logic gate* NOR sudah sesuai					
12	Simbol *logic gate* XOR sudah sesuai					
13	Penulisan rumus Aljabar Boolean sudah sesuai					
14	Gambar skematik IC NOT sudah sesuai					

15	Gambar skematik IC AND sudah sesuai					
16	Gambar skematik IC NAND sudah sesuai					
17	Gambar skematik IC OR sudah sesuai					
18	Gambar skematik IC NOR sudah sesuai					
19	Gambar skematik IC XOR sudah sesuai					

Saran dan masukan :

Instrumen Pengujian Ahli Media
Pada Software Simulasi "DigiChip"

Nama : _____

Bidang Keahlian : _____

Institusi : _____

Ttd :

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	TST
1	Penggunaan warna teks terlihat sudah sesuai					
2	Penggunaan warna angka terlihat sudah sesuai					
3	Penggunaan warna background sudah sesuai					
4	Gambar dapat terlihat dengan jelas					
5	Ukuran gambar sudah sesuai					
6	Penempatan posisi gambar sudah sesuai					
7	Huruf dapat terlihat jelas					
8	Ukuran huruf sudah sesuai					
9	Angka dapat terlihat jelas					
10	Ukuran angka sudah sesuai					
11	Ukuran animasi gambar sudah sesuai					
12	Ukuran animasi tombol sudah sesuai					
13	Efek animasi terlihat artistik					

Kalau kata "sesuai" diganti "proporsional" bagaimana?

Saran dan masukan :

Instrumen Pengujian Aspek Usability
Pada Software Simulasi "DigiChip"

Kalau instrument ini sudah ada yg standar, gunakan saja yg standar. Tdk perlu buat yg baru dan ga perlu divalidasi.

Nama : _____

Kelas : _____

Tipe Ponsel : _____

Tgl : _____

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

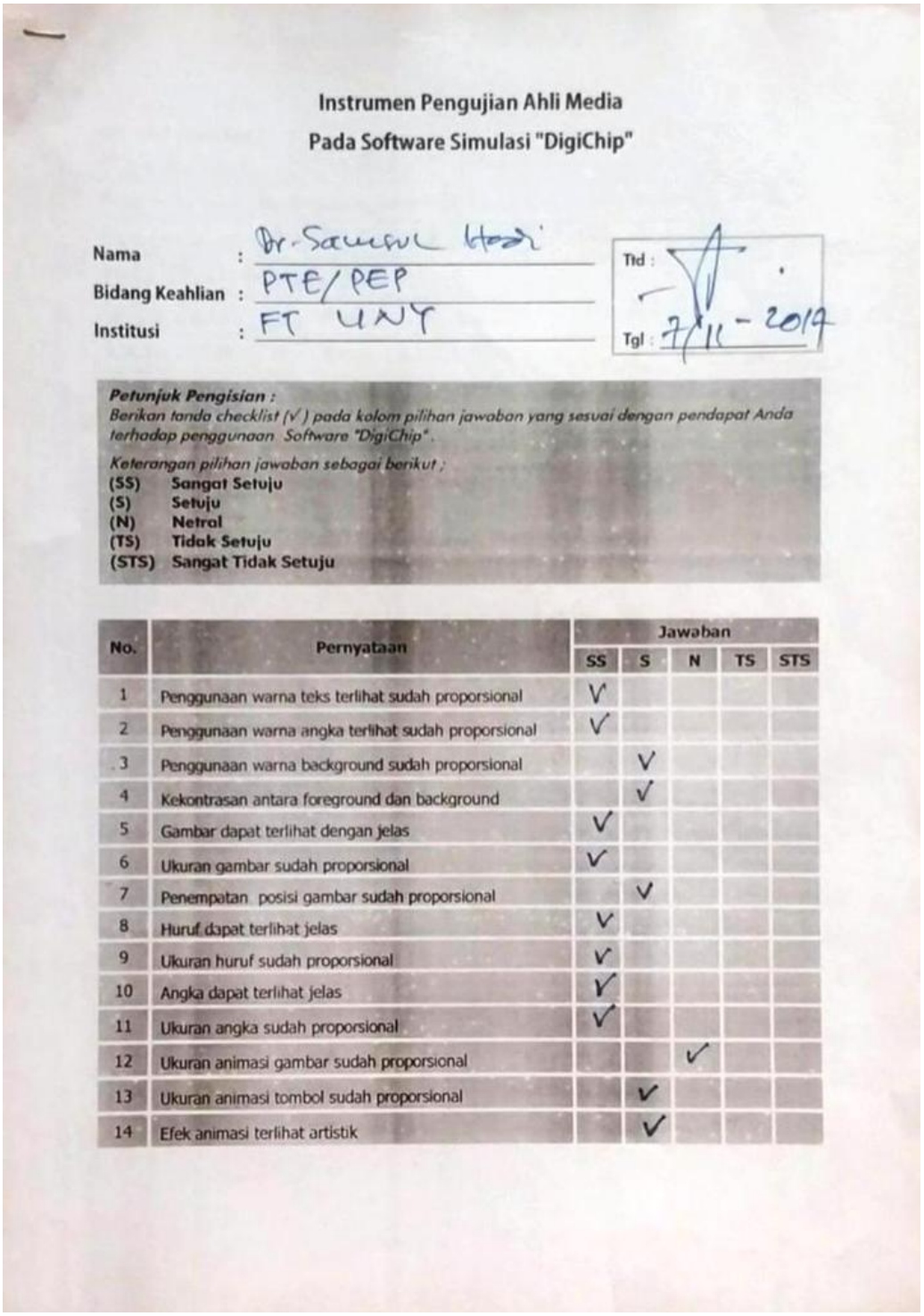
Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

No.	Pernyataan	Jawaban				
		SS	S	N	TS	TST
1	Aplikasi ini membantu saya menjadi lebih efektif dalam pembelajaran teknik digital.					
2	Aplikasi ini membantu saya menjadi lebih produktif dalam pembelajaran teknik digital.					
3	Aplikasi ini bermanfaat dalam pembelajaran teknik digital.					
4	Aplikasi ini memberikan dampak yang lebih dalam pembelajaran teknik digital yang saya lakukan.					
5	Aplikasi ini memberikan saya kemudahan dalam menyelesaikan persoalan dalam pembelajaran teknik digital.					
6	Aplikasi ini mampu menghemat waktu saya dalam memahami pembelajaran teknik digital.					
7	Aplikasi ini sesuai dengan kebutuhan saya.					
8	Aplikasi ini bekerja sesuai dengan harapan saya.					
9	Aplikasi ini mudah digunakan.					
10	Aplikasi ini praktis untuk digunakan.					
11	Aplikasi ini mudah untuk dipahami (user friendly).					
12	Aplikasi ini menggunakan langkah-langkah yang mudah dan sederhana.					
13	Aplikasi ini dapat sesuai dengan kebutuhan saya.					

14	Saya tidak kesulitan dalam menggunakan aplikasi ini.					
15	Saya dapat menggunakan aplikasi ini tanpa panduan tertulis.					
16	Saya tidak menemukan ketidakkonsistenan selama saya menggunakan aplikasi ini.					
17	Penggunaan yang jarang ataupun rutin saya menyukai aplikasi ini.					
18	Kapanpun saya melakukan kesalahan menggunakan aplikasi ini saya dapat kembali dengan cepat dan mudah.					
19	Saya dapat menggunakan aplikasi ini dengan baik setiap waktu.					
20	Saya memahami penggunaan aplikasi ini dengan cepat.					
21	Saya dapat dengan mudah mengingat bagaimana cara penggunaan aplikasi ini.					
22	Sangat mudah untuk memahami cara penggunaan aplikasi ini.					
23	Saya dengan cepat mahir menggunakan aplikasi ini.					
24	Saya merasa puas dengan kinerja aplikasi ini.					
25	Saya akan merekomendasikan aplikasi ini ke teman saya.					
26	Penggunaan aplikasi ini menyenangkan.					
27	Aplikasi ini bekerja seperti apa yang saya inginkan.					
28	Aplikasi ini sangat bagus.					
29	Saya merasa saya harus memiliki aplikasi ini. ini					
30	Aplikasi ini nyaman untuk digunakan.					

Saran dan masukan :

Lampiran 16. Pengujian Ahli Media



Instrumen Pengujian Ahli Media
Pada Software Simulasi "DigiChip"

Nama : Dr. Samsul Hadi
Bidang Keahlian : PTE/PEP
Institusi : FT UNY

Tid :
Tgl : 7/11 - 2019

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

Keterangan pilihan jawaban sebagai berikut ;
(SS) Sangat Setuju
(S) Setuju
(N) Netral
(TS) Tidak Setuju
(STS) Sangat Tidak Setuju

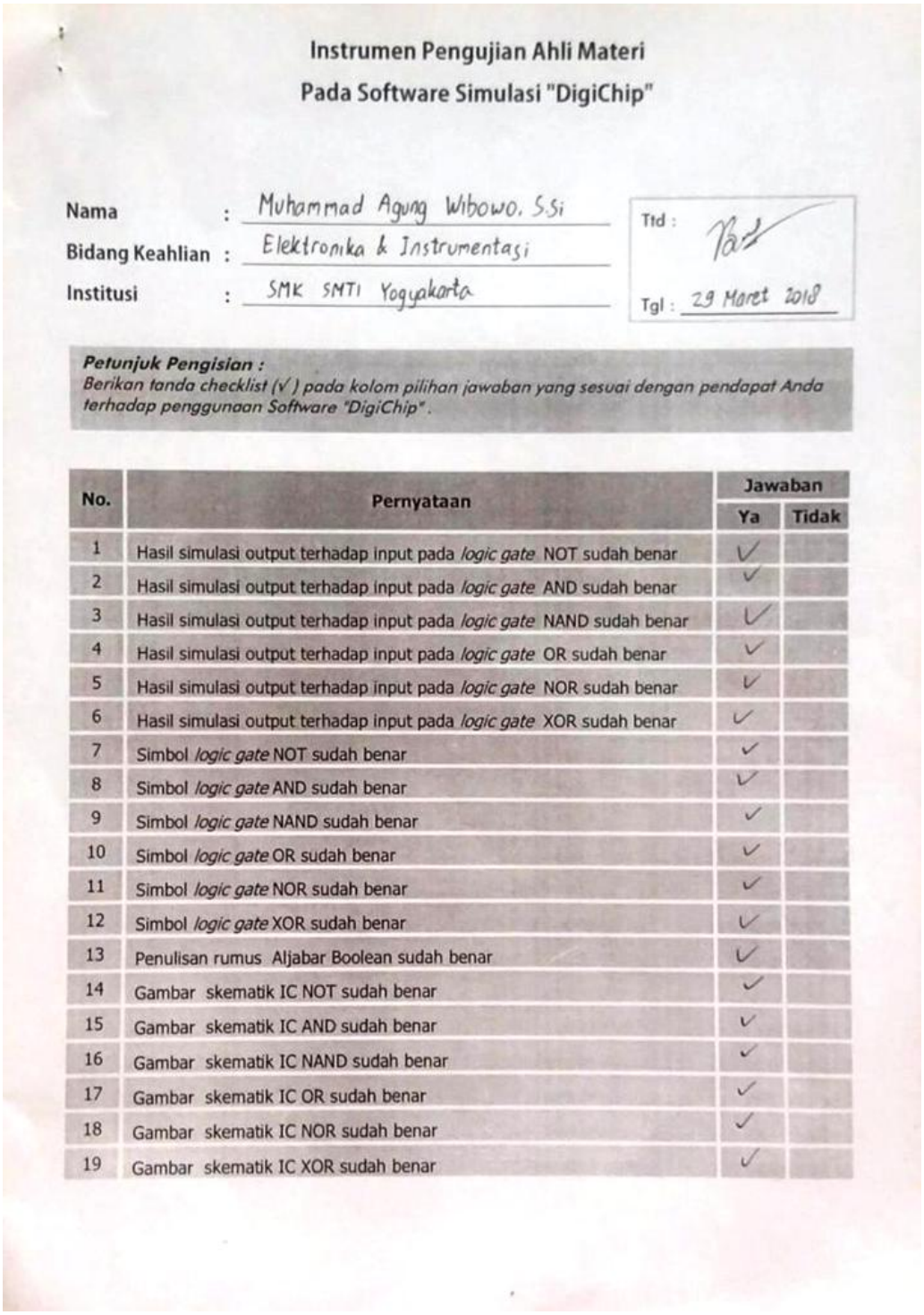
No.	Pernyataan	Jawaban				
		SS	S	N	TS	STS
1	Penggunaan warna teks terlihat sudah proporsional	√				
2	Penggunaan warna angka terlihat sudah proporsional	√				
3	Penggunaan warna background sudah proporsional		√			
4	Kekontrasan antara foreground dan background		√			
5	Gambar dapat terlihat dengan jelas	√				
6	Ukuran gambar sudah proporsional	√				
7	Penempatan posisi gambar sudah proporsional		√			
8	Huruf dapat terlihat jelas	√				
9	Ukuran huruf sudah proporsional	√				
10	Angka dapat terlihat jelas	√				
11	Ukuran angka sudah proporsional	√				
12	Ukuran animasi gambar sudah proporsional			√		
13	Ukuran animasi tombol sudah proporsional		√			
14	Efek animasi terlihat artistik		√			

Saran dan masukan :

1. Sebaiknya ada "pengantar" yg menjelaskan penggunaan media & fungsi dari media (SMK kls berapa, KI/KD apa).
2. Sebaiknya ada quiz untuk mengecek ketercapaian penggunaan media
3. Animasi perlu diperkaya dengan suara & inputan gerbang akan lebih baik bila dibuat variabel (tidak hanya 2 input).
4. Akan lebih baik jika bisa gerbang kombinasi.

Tapi yg lebih penting sesuaikan dengan KI/KD yang disasar

Lampiran 17. Pengujian Ahli Materi



Instrumen Pengujian Ahli Materi
Pada Software Simulasi "DigiChip"

Nama : Muhammad Agung Wibowo, S.Si

Bidang Keahlian : Elektronika & Instrumentasi

Institusi : SMK SMTI Yogyakarta

Ttd :

Tgl : 29 Maret 2018

Petunjuk Pengisian :
Berikan tanda checklist (√) pada kolom pilihan jawaban yang sesuai dengan pendapat Anda terhadap penggunaan Software "DigiChip".

No.	Pernyataan	Jawaban	
		Ya	Tidak
1	Hasil simulasi output terhadap input pada *logic gate* NOT sudah benar	√	
2	Hasil simulasi output terhadap input pada *logic gate* AND sudah benar	√	
3	Hasil simulasi output terhadap input pada *logic gate* NAND sudah benar	√	
4	Hasil simulasi output terhadap input pada *logic gate* OR sudah benar	√	
5	Hasil simulasi output terhadap input pada *logic gate* NOR sudah benar	√	
6	Hasil simulasi output terhadap input pada *logic gate* XOR sudah benar	√	
7	Simbol *logic gate* NOT sudah benar	√	
8	Simbol *logic gate* AND sudah benar	√	
9	Simbol *logic gate* NAND sudah benar	√	
10	Simbol *logic gate* OR sudah benar	√	
11	Simbol *logic gate* NOR sudah benar	√	
12	Simbol *logic gate* XOR sudah benar	√	
13	Penulisan rumus Aljabar Boolean sudah benar	√	
14	Gambar skematik IC NOT sudah benar	√	
15	Gambar skematik IC AND sudah benar	√	
16	Gambar skematik IC NAND sudah benar	√	
17	Gambar skematik IC OR sudah benar	√	
18	Gambar skematik IC NOR sudah benar	√	
19	Gambar skematik IC XOR sudah benar	√	

Saran dan masukan :

Untuk jumlah input dan output, tombol yg dapat digunakan agar diperbanyak, nama IC ditulis untuk tiap logic. Pengembangan selanjutnya dapat dikembangkan

Lampiran 18. Surat Izin Penelitian

KEMENTERIAN RISET, TEKNOLOGI, DAN PENDIDIKAN TINGGI
UNIVERSITAS NEGERI YOGYAKARTA
PROGRAM PASCASARJANA
Jalan Colombo Nomor 1 Yogyakarta 55281
Telp. Direktur (0274) 550835, Asdir/TU (0274) 550836 Fax. (0274)520326
Laman: pps.uny.ac.id Email: pps@uny.ac.id, kerjasama_pasca@yahoo.com

Nomor : 6476 /UN34.17/LT/2017 — 7 Juli 2017
Hal : Izin Penelitian

Yth. Pimpinan Sekolah Menengah Teknologi Industri (SMTI) Yogyakarta
Jl. Kusumanegara No. 3 Yogyakarta

Bersama ini kami mohon dengan hormat, kiranya Bapak/Ibu/Saudara berkenan memberikan izin kepada mahasiswa jenjang S-2 Program Pascasarjana Universitas Negeri Yogyakarta:

Nama	: TITIH REJYASMITO HADI, S.PD.
NIM	: 14702251086
Program Studi	: Pendidikan Teknologi dan Kejuruan
Konsentrasi	: Vokasi Elektro

untuk melaksanakan kegiatan penelitian dalam rangka penulisan tesis yang dilaksanakan pada:

Waktu	: juni 2017 s.d agustus 2017
Lokasi/Objek	: Sekolah Menengah Teknologi Industri (SMTI) Yogyakarta
Judul Penelitian	: Perancangan dan Analisis Kualitas *Software* Simulasi "DigiChip" *Platform Android* Berbasis *Mobile Learning* untuk Mata Pelajaran Teknik Digital
Pembimbing	: Dr. Eko Marpanaji, M.T.

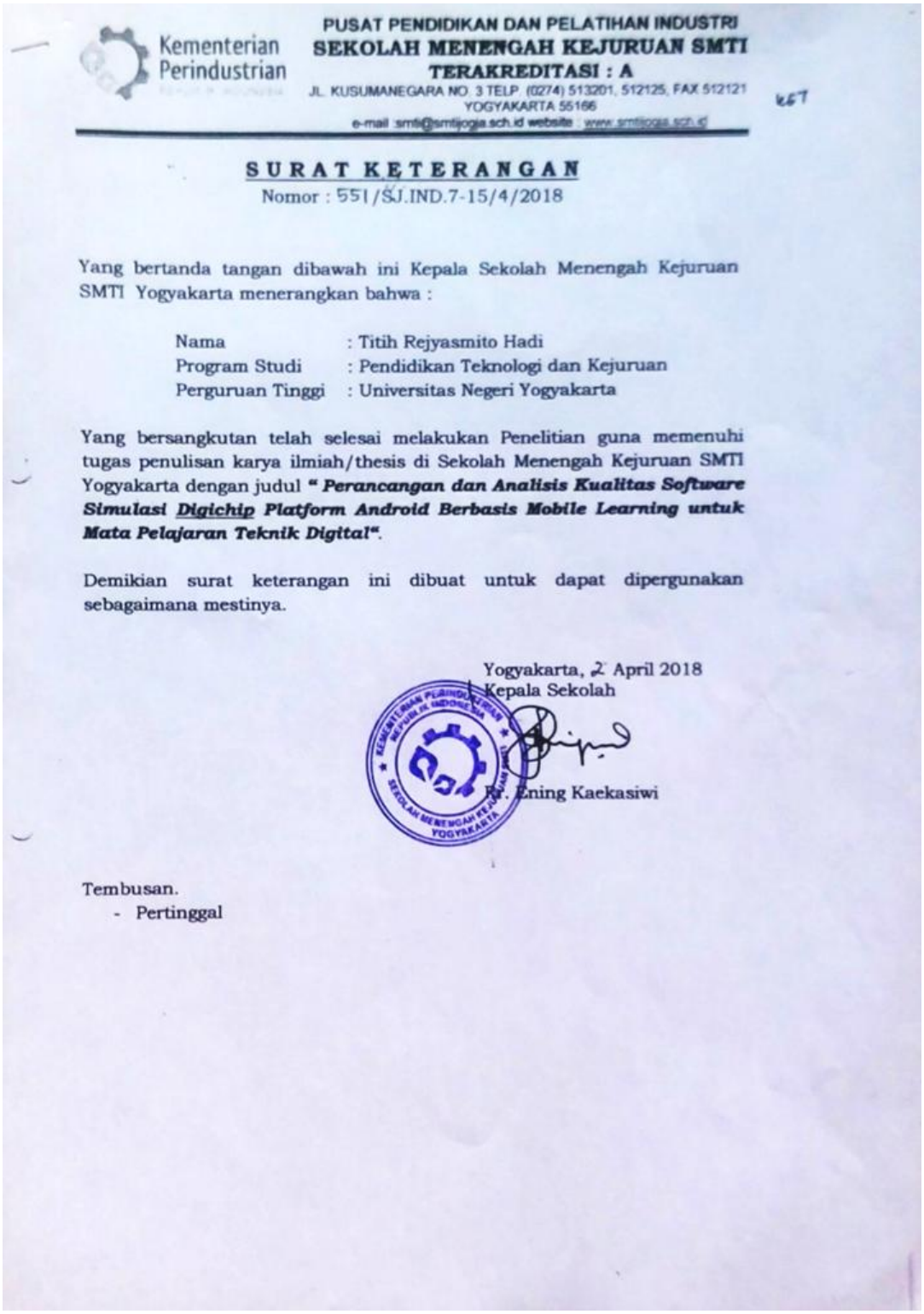
Demikian atas perhatian, bantuan dan izin yang diberikan, kami ucapkan terima kasih

Asisten Direktur I,

Dr. Sugito, MA.
NIP 19600410 198503 1 002

Tembusan:
Mahasiswa Ybs.

Lampiran 19. Surat Penelitian



Kementerian Perindustrian

PUSAT PENDIDIKAN DAN PELATIHAN INDUSTRI
SEKOLAH MENENGAH KEJURUAN SMTI
TERAKREDITASI : A
JL. KUSUMANEGARA NO. 3 TELP. (0274) 513201, 512125, FAX 512121
YOGYAKARTA 55166
e-mail :smti@smtijogja.sch.id website : www.smtijogja.sch.id

SURAT KETERANGAN
Nomor : 551/SJ.IND.7-15/4/2018

Yang bertanda tangan dibawah ini Kepala Sekolah Menengah Kejuruan SMTI Yogyakarta menerangkan bahwa :

Nama	: Titih Rejyasmito Hadi
Program Studi	: Pendidikan Teknologi dan Kejuruan
Perguruan Tinggi	: Universitas Negeri Yogyakarta

Yang bersangkutan telah selesai melakukan Penelitian guna memenuhi tugas penulisan karya ilmiah/thesis di Sekolah Menengah Kejuruan SMTI Yogyakarta dengan judul ***" Perancangan dan Analisis Kualitas Software Simulasi Digichip Platform Android Berbasis Mobile Learning untuk Mata Pelajaran Teknik Digital"***.

Demikian surat keterangan ini dibuat untuk dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 2 April 2018
Kepala Sekolah

Ir. Ening Kaekasiwi

Tembusan.
- Pertinggal

Lampiran 20. Dokumentasi Uji *Usability*