p-ISSN : 2548-1932
e-ISSN : 2549-7758

IMPLEMENTASI METODE *FORWARD CHAINING* DALAM SISTEM PENDETEKSI KERUSAKAN *HARDWARE* PADA KOMPUTER DAN *LAPTOP* BERBASIS *ANDROID*

Peti Savitri[1)], Trisna Hadi[2)]
Program Studi Teknik Informatika[1)2)]
Universitas Sangga Buana YPKP[1)], Sekolah Tinggi Sains dan Teknologi Indonesia[2)]
petisavitri@gmail.com[1)], trisn.hadi@gmail.com [2)]

ABSTRAK

Artikel ini menyajikan hasil penelitian mengenai pembangunan sistem berbasis *android* yang menerapkan salah satu metode dalam mesin inferensi sistem pakar yaitu *forward chaining*. Sistem yang dibangun adalah sistem pakar pendeteksi kerusakan *hardware* pada komputer maupun *laptop* dengan tujuan membantu para user pemula mengetahui letak kerusakan *hardware* pada komputer dan *laptop*, serta membantu user untuk dalam menangani kerusakan tersebut. Basis pengetahuan sebagai komponen utama sistem pakar ini selain bersumber dari pakar yang sudah terbiasa menghadapi kerusakan komputer atau *laptop* dan menanganinya, juga melalui beberapa buku yang mengupas hal-hal yang berkaitan dengan dasar-dasar penanganan kerusakan komputer dan *laptop*. Metode yang digunakan untuk membangun sistem adalah metode prototipe (*prototyping*), sedangkan *tools* yang digunakan dalam melakukan analisa dan desain adalah *unified modeling system* (UML). Penelitian ini menghasilkan sistem pakar pendeteksi kerusakan *hardware* pada komputer dan *laptop* berbasis *android* yang sudah menerapkan metode *forward chaining* dalam proses kerjanya.

Kata Kunci *:* sistem, pakar, android, *forwad chaining*, kerusakan, komputer**.**

I. PENDAHULUAN

Meskipun tingkat penggunaan gadget sebagai pengganti *personal computer* dan *laptop* semakin tinggi, namun sampai saat ini komputer dan *laptop* masih menjadi sebuah kebutuhan tersendiri bagi sebagian orang, terutama para pelajar, mahasiswa, guru, dosen ataupun pegawai. Ada kelebihan tersendiri bagi pemakai komputer atau laptop terhadap penggunaan komputer atau *laptop* yang tidak bisa begitu mudah diganti dengan perangkat sejenis gadget.

Hanya saja ketika mereka menghadapi komputer atau *laptop* yang rusak, banyak dari pengguna tidak dapat memperbaikinya karena tidak adanya pengetahuan atau keterampilan. Bagi sebagian kalangan yang tidak memiliki masalah finansial, cara yang praktis adalah dengan membawa komputer atau *laptop* mereka ke penyedia jasa perbaikan komputer atau *laptop*, tapi bagi sebagian orang biaya *service* dirasa cukup mahal. Atas latar belakang inilah penelitian dibuat, sehingga mereka yang ingin mencoba memperbaiki sendiri atau belajar memperbaiki komputer dapat melakukannya melalui sistem yang dihasilkan dari penelitian ini.

Adapun permasalahan dalam penelitian ini dibatasi pada beberapa hal berikut ini, yaitu: (1) sistem hanya menerima 21 (dua puluh satu) gejala/jenis kerusakan yang sering muncul pada *PC*, (2) sistem hanya menerima 6 (enam) gejala/jenis kerusakan yang sering muncul pada *laptop*, (3) sistem mengelola data gejala kerusakan *hardware*, (4) sistem menghasilkan kesimpulan letak kerusakan, dan (5) sistem memberikan cara penanganan kerusakan/solusi.

Sedangkan tujuan dari penelitian adalah: (1) untuk membantu *user* pemula mengetahui letak kerusakan pada komputer dan *laptop*, (2) untuk membantu *user* untuk segera mengambil tindakan dalam penanganan *error* pada komputer dan *laptop*.

II. METODOLOGI PENELITIAN

Dalam proses pembangunannya, sistem pendeteksi kerusakan *hardware* pada komputer maupun *laptop* ini menggunakan metode *prototype* (*prototyping*), sedangkan *tools* yang digunakan dalam melakukan analisis dan desainnya mengacu pada metode berorientasi objek dengan menggunakan bahasa pemodelan *Unified Modeling Language* (UML).

Prototyping[1] adalah kombinasi metode yang memungkinkan bentuk fisik atau visual untuk diberikan ke sebuah ide (Kelley &

Littman, 2006; Schrage, 2013) dan memiliki peran penting dalam proses pengembangan produk, memungkinkan desainer untuk menentukan masalah desain, memenuhi kebutuhan pengguna dan persyaratan teknik, dan verifikasi solusi desain (De Beer, Campbell, Truscott, Barnard, & Booysen, 2009; Moe, Jensen, & Wood, 2004; Viswanathan & Linsey, 2009; Yang & Epstein, 2005).

Menurut Rifa'atunnisa, Eri Satria,dan Rinda Cahyana[7], metode pengembangan *prototype* yaitu metode yang menggunakan pendekatan untuk membangun suatu program dengan cepat melalui langkah-langkah berikut, yaitu: pengumpulan kebutuhan dan perbaikan, perancangan cepat, membentuk *prototype*, evaluasi pelanggan terhadap *prototype*, perbaikan *prototype* dan produk rekayasa.

Model ini bermanfaat bagi *customer* untuk menyampaikan hal-hal yang bersifat teknis atau menyampaikan spesifikasi kebutuhannya kepada pengembang perangkat lunak [8]. Model prototipe diawali dengan kebutuhan dari sisi calon pengguna terhadap perangkat lunak yang akan dibangun . Kemudian prototipe dibuat agar apa yang sebenarnya diinginkan pelanggan lebih terbayang. Jadi di dalam pengembangan prototipe, aplikasi yang sebenarnya juga dikembangkan sehingga sesuai dengan kebutuhan pelanggan.

Nazarudin mengatakan dalam bukunya yang berjudul Komputer dan *Troubleshooting*, bahwa metode prototipe merupakan metode yang sangat baik dalam menyelesaikan kesalahpahaman yang diakibatkan ketidak-mampuan user mendefinisikan kebutuhannya secara jelas[5]. Sementara menurut Neni Purwati dan Hendra Kurniawan dalam Konferensi Nasional Sistem & informatika STMIK STIKOM Bali, menjelaskan bahwa *prototyping* disebut juga sebagai *rapid application design* (RAD) yang proses kerjanya dalam merancang sistem sederhana dan cepat melalui interaksi yang berulang-ulang[6].

Sedangkan Wibowo Agus bersama rekannya Azimah Ariana mengungkapkan bahwa Tahapan-tahapan model *prototyping* terdiri dari[10]: (1) dikumpulkannya kebutuhan pelanggan sebagai dasar untuk mengidentifikasi sistem secara garis besar oleh pengembang. (2) Membangun *prototyping* dengan cara merancang *layout* sementara misalnya dengan cara menentukan format *input* dan *output*. (3) Mengevaluasi apakah *prototyping* yang sudah dibuat telah sesuai dengan yang diinginkan pelanggan. (4) Mengodekan sistem dalam tahap ini *prototyping* yang sudah disepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai. (5) Melakukan pengujian jika sisten sudah siap digunakan baik melalui pengujian *White Box*, *Black Box*, *Basis Path*, atau emetode pengujian lainnya. (6) Mengevaluasi sistem apakah telah sesuai dengan yang diharapkan *customer*. (7) Mengevaluasi *Protoptype* Perangkat lunak hasil pengujian yang sudah diterima pelanggan agar siap digunakan.

Versi lain proses pembuatan prototipe dapat dilihat sebagai berikut pada gambar 1 berikut ini:

Gambar 1. Langkah-langkah *Prototyping*[6]

Seiring dengan penggunaan metode prototipe, untuk mendapatkan data yang dibutuhkan dalam membangun sistem ini dilakukan beberapa teknik, yaitu: (1) studi pustaka; studi pustaka dilakukan dengan mempelajari tentang teori-teori yang menjadi referensi dan pendukung dalam pembuatan sistem pakar untuk deteksi kerusakan perangkat keras, mempelajari gejala, penyebab, dan solusi penanggulangan terhadap kerusakan perangkat keras komputer, dan pemrograman berbasis *mobile Android*. (2) Wawancara dan observasi; teknik ini digunakan untuk berkonsultasi dengan pakar atau ahli dalam bidang perangkat keras komputer, diantaranya guru komputer, teknisi *service* komputer, dan *service center* guna membangun basis pengetahuan yang menjadi komponen utama sebuah sistem pakar.

a. *Unified Modeling Language* (UML) disinggung oleh Fajarianto dalam jurnalnya[2] yaitu sebuah bahasa pemodelan standar dalam industri perangkat lunak

p-ISSN : 2548-1932
e-ISSN : 2549-7758

yang digunakan untuk memvisualisasikan dan merancang sistem perangkat lunak serta mendokumentasikan-nya. Dikemukakan pula bahwa UML lebih tepat digunakan ketika perangkat lunak dibuat dengan bahasa pemrograman berorientasi objek. Desain yang dihasilkan melalui diagram UML selanjutnya pada tahap implementasi diterjemahkan ke dalam baris kode

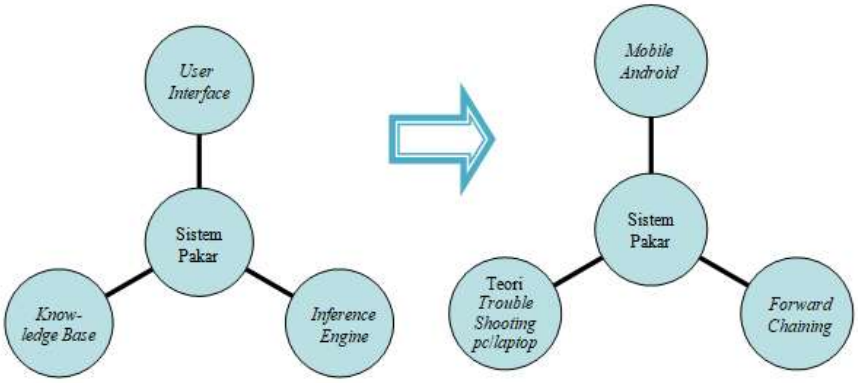
b. program. UML terdiri atas 13 jenis diagram resmi yaitu diagram: *activity*, *class*, *communication*, *component*, *composite structure*, *deployment*, *interactive overview*, *object*, *package*, *sequence*, *state machine timing*, dan *use case*.[2]

III. ANALISA DAN PERANCANGAN

Kerusakan komputer adalah kondisi dimana komputer tidak bisa menerima input atau intruksi dari user. Kerusakan komputer bisa terjadi kapan saja tanpa bisa kita prediksi sebelumnya. Beberapa faktor bisa menjadi penyebab terjadinya kerusakan pada komputer, bisa karena faktor alat (*hardware*) ataupun karena faktor pengguna (*user*). Jangka waktu atau usia serta cara pemakaian juga bisa menjadi faktor pemicu adanya kerusakan pada komputer. Kerusakan pada komputer bisa terletak pada *hardware* atau *software*. *Error* yang terjadi saling mempengaruhi satu sama lainnya, jika *hardware error* maka kita tidak bisa mengakses *software* dan jika *software error* maka kita tidak bisa mengerjakan pekerjaan kita. *Error* yang terjadi pada *hardware* atau *software* sangat banyak dan bervariatif.

Berdasarkan penelitian melalui wawancara dan melakukan diagnosa langsung pada kasus-kasus kerusakan komputer dan studi pustaka yang bersumber dari buku Komputer dan *Troubleshooting* (Ramdhani Nazarudin) serta buku Mahir Memperbaiki dan Merawat *Netbook* (Yulius Eka Agung Seputera, ST, MSI) serta dilengkapi dengan pengumpulan data *quisioner* terhadap *error* yang sering ditemui dalam kerusakan komputer, maka sistem yang akan dibuat dapat mendiagnosa 22 gejala kerusakan komputer[9] meliputi: (1) tampilan *keyboard error*, (2) tombol *keyboard* tidak berfungsi, (3) beberapa tombol tidak berfungsi, (4) respon *keyboard* terlalu cepat, (5) kursor *mouse* tidak berjalan, (6) kursor *mouse* bergerak *horizontal/vertical*, (7) komputer sering "*hang"*, (8) komputer tidak dapat *booting*, (9) suara *beep* terus menerus, (10) suara *beep* normal tapi tidak ada tampilan, (11) komputer me-*restart* sendiri, (12) pembacaan *disk* tersendat-sendat, (13) susah buka tutup CD/DVD ROOM, (14) CD/DVD tidak terdeteksi, (15) *harddisk* tidak bisa dipartisi, (16) h*arddisk* tidak bisa diformat, (17) *hardisk bad sector*, (18) *fan* atau kipas *power supply* tidak berputar, (19) *fan* atau kipas *power supply* berisik, (20) e*rror display monitor horizontal*, (21) *error display monitor vertical*, (22) *display monitor error* RGB.

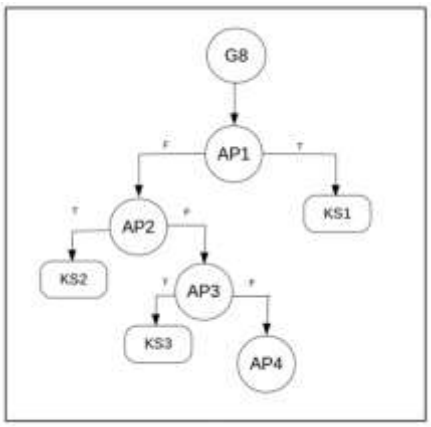
Sistem juga dapat mendiagnosa 6 kerusakan yang sering terjadi pada *laptop*[4], yaitu: (1) l*aptop* tidak menyala, (2) *screen error*/tidak ada tampilan, (3) h*arddisk error*, (4) *CD/DVD* bermasalah, (5) *souncard error*, (6) o*verheating/*panas yang berlebihan. Gejala dan diagnosa yang sudah dirinci di atas di dalam sistem pakar dijadikan sebagai basis pengetahuan, yaitu tempat penyimpanan pengetahuan dalam memori komputer. Sedangkan penelusuran basis pengetahuan digunakan mesin inferensi *forward chaning* yang merupakan otak dari aplikasi sistem pakar[5]. Melalui mesin inferensi inilah fakta yang dimasukan *user* pada akhirnya menuju kepada suatu kesimpulan[4] (lihat gambar 2).



Gambar 2. Komponen Sistem Pakar Secara Umum dan Penerapannya pada Sistem Pendeteksi Kerusakan *Hardware* Pada Komputer Maupun *Laptop*

p-ISSN : 2548-1932
e-ISSN : 2549-7758

Sistem pakar mempunyai dua komponen utama, yaitu basis pengetahuan atau *knowledge base* sebagai tempat tersimpannya pengetahuan di memori komputer dengan menggunakan kaidah produksi. Dan komponen lainnya yaitu mesin inferensi yang dapat dikatakan sebagai otak dari aplikasi sistem pakar. Pada sistem prototipe yang dibuat, komponen sistem pakar dapat dijelaskan sebagi berikut: 1. Basis pengetahuan berisi tentang teori pakar mengenai penanganan kerusakan komputer/*laptop* beserta solusinya, teori-teori yang berkaitan dengan kerusakan ini disimpan pada media penyimpanan komputer sebagai basis pengetahuan yang nantinya akan dihubungkan dengan mesin inferensi *forward chaining*. 2. Mesin Inferensi; Mesin inferensi yang digunakan adalah *forward chaning* (runut maju) yang merupakan otak dari aplikasi sistem pakar yang akan menuntun user memasukan fakta sehingga diperoleh suatu kesimpulan.. Metode inferensi ini cocok digunakan dalam menangani masalah pengendalian dan peramalan[3]. Berikut adalah perancangan runut maju penanganan kerusakan:



Keterangan gambar:
G = Gejala
KS = Kerusakan dan Solusi
AP = Arahan dan Pertanyaan
T = *True* (Benar)
F = *False* (Salah)

Gambar 3. Salah satu *Graph* Keputusan pada Sistem Pendeteksi Kerusakan *Hardware* Komputer

Mesin inferensi ini mengumpulkan fakta-fakta gejala yang terjadi secara runut maju, kemudian diperolehlah hasil kesimpulan kerusakan (lihat gambar 3). Data fakta bersumber dari gejala kerusakan dan mesin inferensi mengarahkan kesimpulan dengan basis pengetahuan yang bersumber dari pakar.
Berikut contoh beberapa aturan/algoritma yang digunakan pada pembangunan sistem:

Aturan 1 (Laptop)

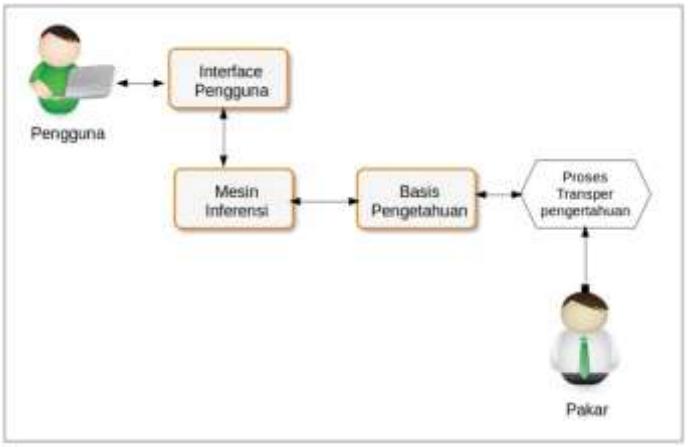
```
IF
Jenis perangkat adalah laptop   AND
(
Sumber listrik menggunakan
charger dan baterai              OR
Sumber listrik menggunakan
charger                          OR
listrik menggunakan charger
dan baterai                      OR
)                               AND
LED Laptop menyala              AND
LED Laptop berkedip jika
kabel digerakan                 AND
Internal Jack Putus             AND
THEN Kerusakan pada Internal
Jack, ganti internal Jack
```

Aturan 1 (Komputer)

```
IF
Jenis perangkat adalah
Komputer                        AND
Keyboard tidak terdeteksi       AND
Connector Keyboard tidak
renggang                        AND
Kabel keyboard bagus            AND
Indikator Capslock tidak
menyala                         AND
THEN Kerusakan IC Keyboard,
ganti dengan Keyboard baru
```

Pengguna melakukan interaksi langsung dengan aplikasi melalui *interface* (gambar 4), kemudian aplikasi memproses data dengan menggunakan mesin inferensi. Mesin inferensi mengambil fakta-fakta berdasarkan basis pengetahuan yang merupakan ilmu pengetahuan yang telah terdokumentasi dan bersumber dari pakar. *Interface* pengguna menggunakan aplikasi *mobile* berbasis *android*, dan untuk mesin inferensi menggunakan metoda *forward chaining* atau runut maju untuk memperoleh kesimpulan.

p-ISSN : 2548-1932
e-ISSN : 2549-7758



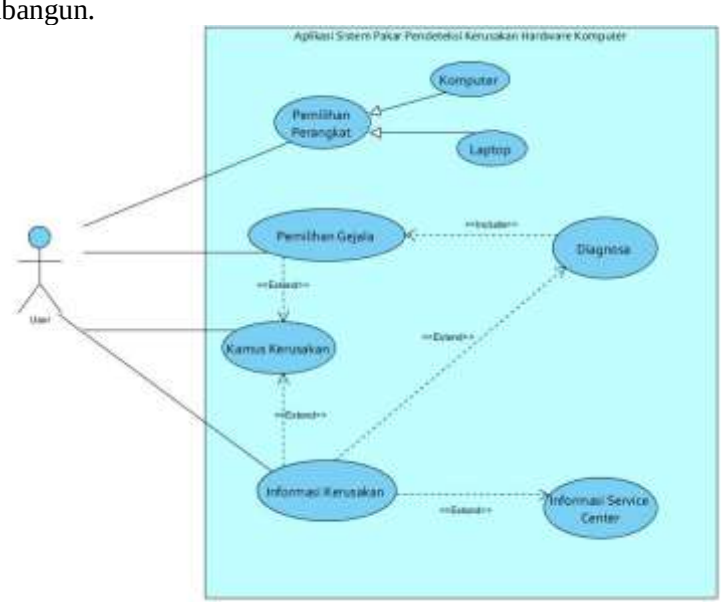
Gambar 4. Alur Sistem Pendeteksi Kerusakan *Hardware* Komputer

Berdasarkan analisis, maka perlu dibuat aplikasi yang dapat memenuhi kebutuhan *user* serta dapat membantu *user* terhadap permasalah *error* atau kerusakan komputer dan *laptop*, aplikasi tersebut memiliki karakteristik sebagai berikut: (1) aplikasi mampu mendeteksi kerusakan komputer dan laptop berdasarkan gejala-gejala yang ada. (2) Aplikasi mampu memberikan letak kerusakan yang terjadi. (3) Aplikasi mampu memberikan solusi/arahan terhadap kerusakan yang terjadi. (4) Aplikasi mampu diakses kapanpun dan dimanapun *(Mobile Aplication)*.

Adanya perancangan sistem bertujuan untuk memudahkan *user* dalam mendapatkan informasi tentang berbagai jenis serta letak kerusakan pada komputer/*laptop* yang digunakan oleh *user*. Dari perancangan sistem ini maka akan terlihat alur dan cara kerja dari aplikasi yang akan dibangun.

Dalam perancangan sistem digambarkan serta dijabarkan setiap proses serta aturan yang ada pada sistem yang akan dibangun sehingga menjadi bahan acuan dalam pembuatan aplikasi agar aplikasi sesuai dengan kebutuhan *user* serta aplikasi dapat bekerja dengan optimal. Perancangan aplikasi sistem pakar pendeteksi kerusakan komputer dan *laptop* ini menggunakan bahasa permodelan *Unfield Modeling Language (UML).*

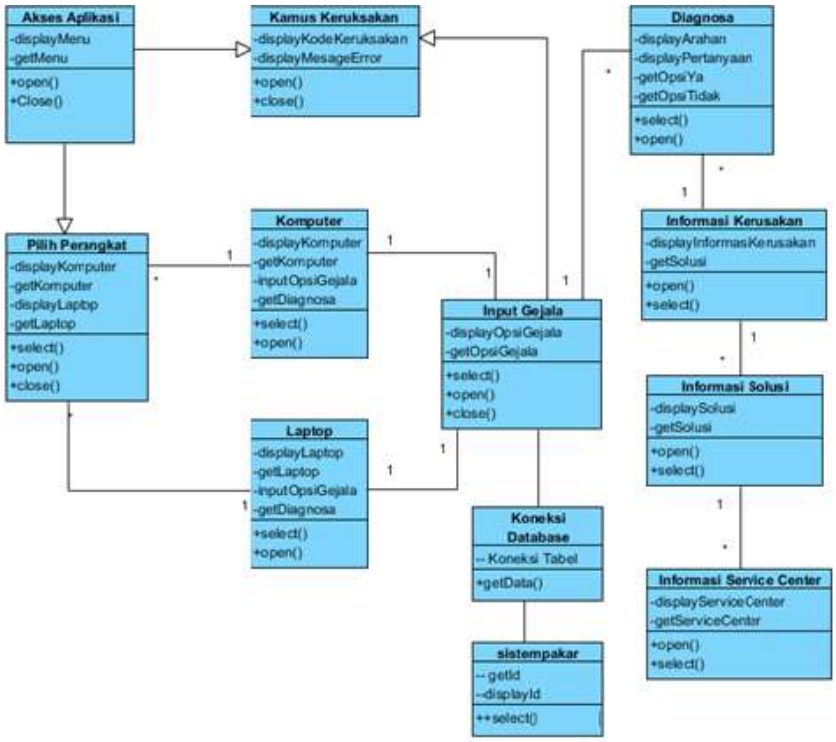
Model *use case* menjelaskan mengenai aktor yang terlibat dengan perangkat lunak yang dibangun beserta proses-proses yang ada di dalamnya. *Use case* yang terbentuk pada diagram di bawah (gambar 5) terdiri dari fungsi pemilihan perangkat di mana terdapat dua opsi pilihan yaitu komputer dan *laptop* , pemilihan gejala, kamus kerusakan, informasi kerusakan dan diagnosa.



Gambar 5. *Use Case Diagram* Sistem Pendeteksi Kerusakan *Hardware* Komputer

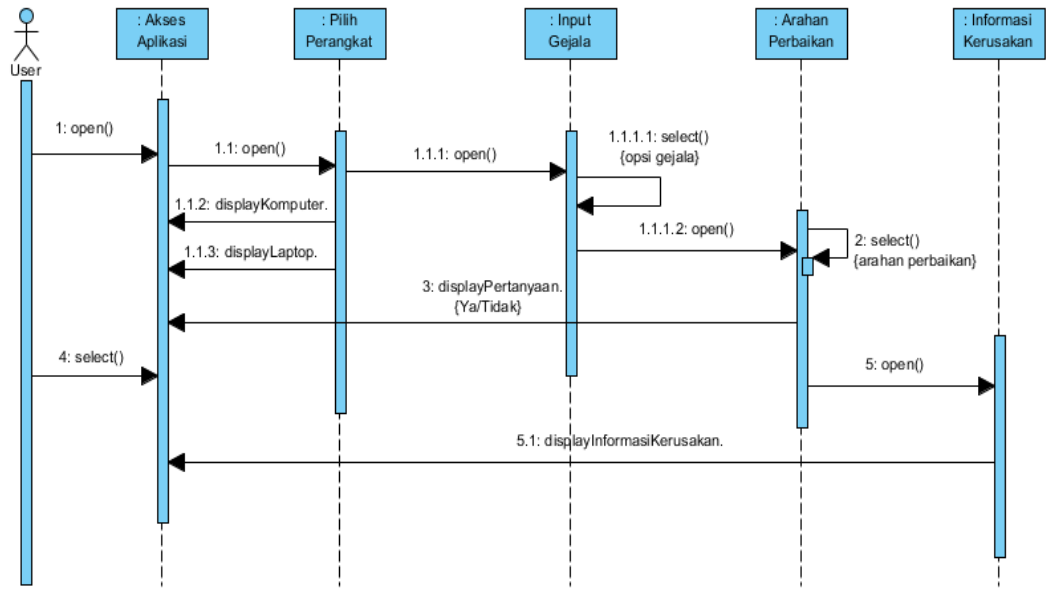
p-ISSN : 2548-1932
e-ISSN : 2549-7758

Adapun fungsi informasi *service center* merupakan relasi *extend* yang tersambung dengan informasi kerusakan, apabila kerusakan perangkat yang dialami oleh pengguna tidak terpecahkan. Berdasarkan *Use Case Diagram* (gambar 5), secara terstruktur *class* yang akan dibangun untuk sistem dapat dilihat dari *class diagram* (gambar 6).



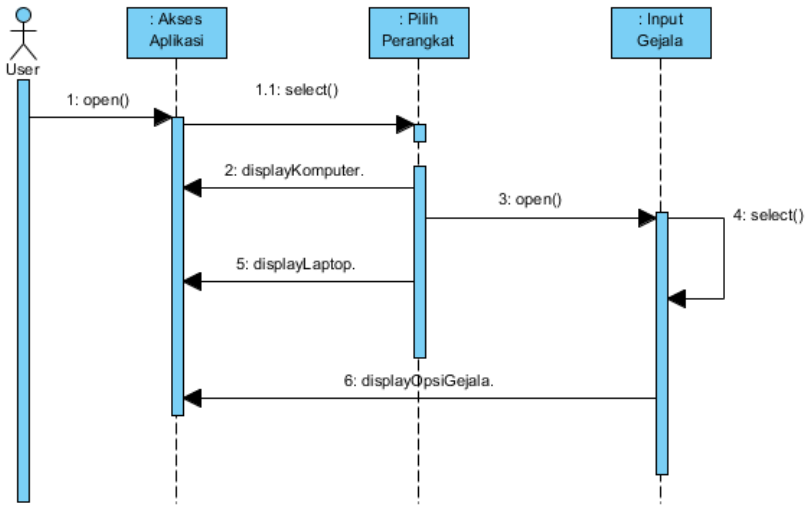
Gambar 6. *Class Diagram* Sistem Pendeteksi Kerusakan *Hardware* Komputer

Selanjutnya interaksi antar objek yang akan dibangun diperlihatkan pada gambar 7 yaitu *Sequence Diagram*. Jumlah *Sequence Diagram* digambarkan sebanyak jumlah *use case* yang sudah dibuat sebelumnya pada diagram *use case*. Berikut ini adalah contoh dari diagram *sequence* yang digambarkan untuk Sistem Pendeteksi Kerusakan *Hardware* Komputer dan *Laptop*. (Lihat Gambar 7, 8, dan 9).

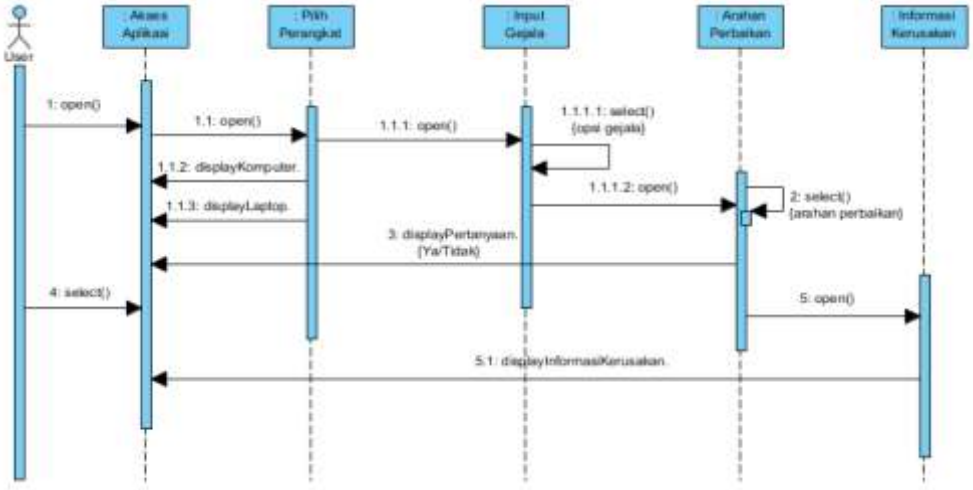


Gambar 7. *Sequence Diagram* Pemilihan Perangkat

p-ISSN : 2548-1932
e-ISSN : 2549-7758



Gambar 8. *Sequence Diagram Input* Gejala



Gambar 9. *Sequence Diagram* Informasi Kerusakan

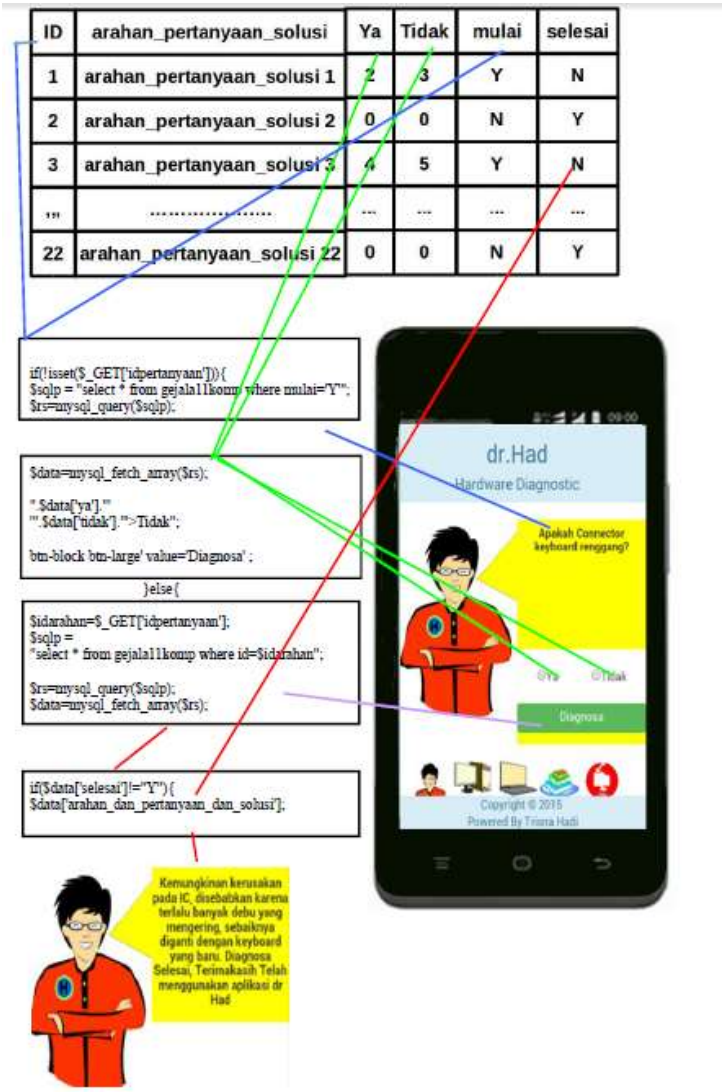
Sesudah perancangan dilakukan sampai pada tahap perancangan *interface*, tahap selanjutnya adalah mengimplementasikan konsep rancangan ke dalam baris *code program*. Perangkat lunak yang diperlukan untuk pembuatan sistem adalah *Sublime* Text 3 dan *Eclipse*. Sedangkan Aplikasi yang diperlukan untuk mengakses aplikasi ini diperlukan perangkat lunak minimal *Android OS*, v2.3.5 (*Gingerbread*). Hasil implementasi (dapat dilihat pada gambar 10) menampilkan tiga menu utama, yaitu: Komputer, *Laptop*, dan Kamus Kerusakan.

Berdasarkan identifikasi kerusakan *hardware* yang sudah dibahas di awal, menu komputer untuk menuju halaman gejala-gejala komputer terdiri dari 22 gejala kerusakan komputer, sedangkan menu *laptop* menuju halaman gejala-gejala kerusakan *laptop* terdiri dari 6 gejala kerusakan *laptop*. Menu Kamus kerusakan menuju halaman kamus kerusakan yang terdiri dari 3 jenis pesan *error*, dalam bentuk *beep*, *mesaage error number*, *blue screen error*.

Sedangkan implementasi *forward chaining* pada sistem pendeteksi kerusakan pada komputer dan *laptop* dapat dilihat pada gambar 11 di bawah ini.

p-ISSN : 2548-1932
e-ISSN : 2549-7758

Gambar 10. Menu-Menu Pada Sistem



Gambar 11. Implementasi *forward chaining* pada sistem

Setelah implementasi dilakukan, tahap selanjutnya adalah pengujian dengan menggunakan metode *blackbox*. Pengujian meliputi pengujian menu utama, pengujian gejala komputer, pengujian gejala *laptop*, dan pengujian kamus kerusakan. Tahap pengujian memperlihatkan keberhasilan.

Hasil akhir sistem yang berbentuk aplikasi *android* diberi nama dr. Had mendapatkan respon yang dianggap posistif. Hal ini dapat kita lihat

p-ISSN : 2548-1932
e-ISSN : 2549-7758

melalui grafik yang terdapat pada gambar 12 mengenai hasil kuisioner yang telah disebar ke 50 orang responden dengan hasil sebagai berikut:

1. Kemudahan penggunaan sistem, memperlihatkan jumlah prosentase responden yang terbesar mengatakan baik (60%) diikuti tanggapan sangat memuaskan sebanyak 30%, sedangkan yang memberi kurang puas dan sedang, masing-masing mendapatkan 5% tanggapan dari 50 responden.

Gambar 12a. Kemudahan Penggunaan

2. Kesesuaian antara gejala kerusakan dengan identifikasi kerusakan pada tampilan aplikasi, yang dalam hal ini merupakan hasil kerja metode *forward chaining*, menurut hasil responden tidak memiliki kekurangan, memiliki nilai sedang 25%, nilai baik 55%, dan nilai sangat baik 20%.

Gambar 12b. Kesesuaian antara gejala kerusakan dengan identifikasi kerusakan pada tampilan aplikasi

3. Pemahaman responden terhadap tata bahasa dan istilah yang digunakan pada aplikasi dirilai responden 65% baik, 15% sedang, 10% sangat baik, kurang dan sangat kurang 5%.

Gambar 12c. Hasil *Questioner* tentang Penilaian Penggunaan Sistem

IV. PENUTUP

Kesimpulan yang didapat dari penelitian ini adalah bahwa aplikasi sistem pakar pendeteksi kerusakan *hardware* komputer ini dapat membantu *user* pemula mengetahui letak kerusakan pada komputer atau *laptop*, sehingga pada akhirnya membantu mereka untuk segera mengambil tindakan dalam penangan *error* yang mereka temukan.

Keakuratan dalam penanganan kerusakan sudah cukup baik, gejala-gejala yang diinputkan bisa menangani permasalahan yang terjadi dan dapat didiagnosa dengan baik dan menghasilkan kesimpulan yang cukup tepat serta pemberian solusi yang baik. Fasilitas informasi *service center* juga memudahkan user untuk mengetahui alamat *service center* yang bisa dikunjungi apabila permasalahan yang dihadapi *user* tidak dapat teratasi oleh sistem. Tampilan aplikasi cukup menarik meskipun terlihat sederhana namun struktur menu yang ada cukup memberikan kemudahan bagi *user*.

DAFTAR PUSTAKA

[1] Deininger, M., et al., 2017. "*Novice designers' use of prototypes in engineering design"*. Elsevier. http://dx/doi.org/10.1016/j.destud.2017.04.002.

[2] Fajarianto, Otto. 2016. "*Prototype Pelayanan Akademik Terhadap Komplain Mahasiswa Berbasis Mobile"*. Jurnal Lentera Ict. ISSN 2338-3143. Vol.3 No.1.

[3] Fajarianto, Otto. 2016. "*Prototype Pelayanan Akademik Terhadap Komplain Mahasiswa Berbasis Mobile"*. Jurnal Lentera Ict. ISSN 2338-3143. Vol.3 No.1.

[4] Kusrini. (2008). *Aplikasi Sistem Pakar*. Yogyakarta: Andi Offset.

[5] Nazarudin, Ramdani. (2006). *Komputer dan Troubleshooting*, Bandung: Informatika

[6] Neni Purwati dan Hendra Kurniawan. 2016. "*Studi Pengembangan Prototype Knowledge Management Pada Pengecekan Judul Tugas Akhir atau Skripsi Fakultas Ilmu Komputer IBI Darmajaya".* Konferensi Nasional Sistem & informatika STMIK STIKOM Bali. ISSN : 2302-3805, *6-7 Februari 2016.*

[7] Rifa'atunnisa, Satria Eri, Cahyana Rinda. 2014. "*Pengembangan Aplikasi Zakat Berbasis Android Menggunakan Metode Prototype"*. Jurnal Algoritma Sekolah Tinggi Teknologi Garut. ISSN : 2302-7339 Vol. 11 No. 1.

[8] Rosa, A. S., dan Salahudin M..(2013). *Rekayasa Perangkat Lunak.* Bandung: Penerbit Informatika, pp. 31 -34.

[9] Saputra, Yulius Eka Agung. (2013). *Mahir Memperbaiki dan Merawat Notebook*. Yogyakata: Gava Media.

[10] Wibowo Agus, Azimah Ariana. (2016). "Rancang Bangun Sistem Informasi Penjaminan Mutu Perguruan Tinggi Menggunakan Metode Throwaway Prototyping Development. Seminar Nasional Teknologi Informasi dan Multimedia". ISSN : 2302-3805.

***