# CASE-BASED REASONING SEBAGAI SOLUSI TROUBLESHOOTING VACUUM CLEANER

**DAVID**
STMIK Pontianak, Jalan Merdeka No. 372 Pontianak, Kalimantan Barat
Telp (0561) 735555, Fax (0561) 737777
Email : DavidLiauw@gmail.com, David_Liauw@yahoo.com

***Abstrak***

*Alat penghisap debu atau Vacuum cleaner telah menjadi salah satu alat kebutuhan yang harus dimiliki untuk membersihkan ruangan dari debu atau kotoran. Kebanyakan pengguna kadang tidak memperhatikan cara penggunaan yang benar sehingga vacuum cleaner menjadi cepat rusak. Penenlitian ini dapat membantu pengguna dalam melakukan troubleshooting sehingga memperpanjang usia vacuum cleaner dan menghemat pengeluaran dan kebersihan. Penelitian ini menggunakan case base reasoning sebagai inference engine dalam mencari solusi kerusakan pada vacuum cleaner. Case base reasoning menggunakan kasus-kasus yang terjadi sebelumnya untuk memecahkan suatu masalah. Pencarian kesamaan antara kasus yang baru dengan kasus-kasus yang lama melalui tahapan 4R yaitu Retrieve, Reuse, Revise, dan Retain. Dalam mewujudkan tahapan retrieve yang cukup kompleks digunakankan lah Java threads. Perhitungan similarity pada tahapan reuse menggunakan metode global similarity dan local similarity. Pengujian aplikasi dilakukan dengan metode test case. Pada saat pengujian, terdapat suatu kesalahan dalam tampilan yang akan diselesaikan pada waktu lain. Hasil dari pengujian sangat baik karena dapat menyesuaikan dengan perhitungan manual. Saran untuk penelitian selanjutnya adalah menggunakan model data yang lebih kompleks agar mendapatkan hasil semaksimal mungkin, serta memperbaiki kesalahan pada penelitian ini.*

***Kata kunci***— Vacuum Cleaner, Case-Based Reasoning, 4R, Threads, Java

***Abstract***

*Vacuum cleaner has become one of the must-have tools to clean the room from dust or dirt. Most users sometimes do not pay attention to how to use it properly so that the vacuum cleaner becomes quickly damaged. This research can help users in troubleshooting to prolong the life of the vacuum cleaner and saves expenses and cleanliness. This research uses case-based reasoning as an inference engine in finding solutions to damage to the vacuum cleaner. Case-based reasoning uses previous cases to solve the problem. Look for similarities between new cases and old cases through the 4R stages, namely Retrieve, Reuse, Revise, and Retain. In realizing a complex phase of the retrieve, Java threads are used. The calculation of similarity in the reuse stage uses global and local similarity methods. Application testing is done by the test case method. During testing, there is an error in the display that will be resolved at another time. The results of the examination are excellent because it can adjust to manual calculations. Suggestions for further research are to use more complex data models to get the maximum results possible, and correct errors in this study.*

***Keywords***— Vacuum Cleaner, Case-Based Reasoning, 4R, Threads, Java

## 1. PENDAHULUAN

Di saat kini, banyak orang mulai menggunakan vacuum cleaner untuk membersihkan rumah. Peralatan tersebut cukup efektif dalam membantu pekerjaan rumah tangga. Tidak perlu mengeluarkan tenaga yang besar juga kebersihan yang terjamin. Alat ini menjadi

pengganti sapu untuk bersih-bersih. Tetapi, alat ini adalah alat elektronik. Alat ini bisa saja rusak atau mengalami masalah yang beragam. Tidak seperti sapu biasa yang dapat dipelajari dalam sekali melihat oleh manusia. Serta tidak memiliki masalah yang aneh-aneh. Vacuum cleaner memiliki kemungkinan untuk mendapatkan masalah yang sulit untuk diketahui.

Beberapa dari masalah-masalah tersebut sudah ada yang terselesaikan dan memberikan solusi yang sesusai. Hal ini dikarenakan masalah yang serupa dapat saja memiliki solusi yang serupa juga. Seperti pemikiran seorang pakar yang dalam mereparasi peralatannya. Pakar tersebut akan terlebih dahulu mengingat kejadian di masa lampaunya tentang kasus yang dikerjakannya. Kemudian pakar tersebut akan mengaitkan masalah yang dia kerjakan saat ini dengan kasus yang sudah pernah terjadi tersebut. Dengan demikian, pakar tersebut akan mengambil suatu keputusan berupa solusi yang mirip dengan solusi pada kasus sebelumnya yang serupa tersebut.

Pada beberapa waktu dekat ini, peneliti menemukan suatu website perusahaan "Dyson" yang dimana didalamnya terdapat suatu layanan yakni "*Get expert help*". Layanan ini memberikan kesempatan kepada user untuk mencari tahu kerusakan atau masalah pada produk yang mereka beli dari Dyson. Layanan tersebut cukup interaktif yakni pertanyaan-pertanyaan yang berlanjut disertai dengan gambar layaknya berkomunikasi dengan seorang teknisi vacuum cleaner. Akan tetapi, terdapat suatu kelemahan dari layanan ini yang didapati dari peneliti setelah menemukan suatu website lain yang berisikan kumpulan diskusi-diskusi tentang masalah atau kerusakan pada vacuum cleaner. Pada forum diskusi tersebut terdapat beberapa masalah yang tidak dijangkau oleh layanan yang diberikan pada website Dyson (https://www.dyson.com). Jika ditemukan hal demikian, maka pengguna layanan tersebut harus menghubungi *support center* perusahaan Dyson yang dimana membutuhkan waktu untuk menyelesaikan masalah mereka.

Oleh karena itu, peneliti akan mencoba untuk menyelesaikan masalah yang tidak dapat dijangkau oleh layanan dari website Dyson tersebut. Terdapat beberapa penelitian sebelumnya yang dapat dijadikan sebagai referensi dalam mengerjakan penelitian ini. Studi terdahulu mengenai troubleshooting komputer PC menggunakan CBR sebagai knowledge management system [1]. Hasil penelitiannya mengintegrasikan teknik analisis tugas kognitif, pengelompokan hierarki dan ontologi dan mengusulkan sistem manajemen pengetahuan CBR berbasis web untuk troubleshooting PC. Studi lainnya mengenai Case-Based Reasoning untuk menyelesaikan permasalahan diagnosa kerusakan Gas Turbine [2]. Hasil penelitiannya berupa aplikasi berbasis kasus untuk diagnosa kerusakan gas turbin dan memiliki akurasi, rawatan, modularitas, parameterisasi, ketahanan, dan integrasi sistem ke dalam infrastruktur yang ada. Penelitian berikutnya mengenai sistem manajemen pengetahuan maintenance sistem perencanaan menggunakan teknik case-based reasoning [3]. Dari penelitian-penelitian sebelumnya, masing-masing menggunakan cara mereka sendiri dalam melakukan retrieval data. Untuk memperkuat proses retrieval data, peneliti menerapkan beberapa prinsip retrieval yang efektif dengan metode indexing baru [4]. sehingga dalam penelitiannya banyak membahas tentang solusi dalam membuat aplikasi case base reasoning yang baik [4].

Pada permasalahan ini, yang menjadi objek adalah suatu mesin yang berbeda dengan sistem tubuh manusia, gejala yang muncul cukup jelas dan saling berkaitan. Akan tetapi, terdapat juga masalah-masalah yang tidak mudah untuk diselesaikan layaknya pada sistem tubuh makhluk hidup. Sehingga berdasarkan dari 3 penelitian diatas, peneliti akan membuat suatu aplikasi *case base reasoning* untuk *trouble shooting* pada vacuum cleaner merek Dyson yang memfokuskan pada proses retrival data.

## 2. METODE PENELITIAN

### *2.1 Perancangan Inference Engine*

Penyelesaian masalah dalam penelitian ini akan digunakan *inference engine* yakni *case base reasoning*. Alasan menggunakan *case base reasoning* adalah data yang dimiliki berupa

kumpulan kasus-kasus, sehingga lebih memungkinan untuk menyelesaikan masalah ini dengan membuat suatu aplikasi dengan *case base reasoning* dibanding dengan metode lainnya.

*2.2 Pemodelan Data*

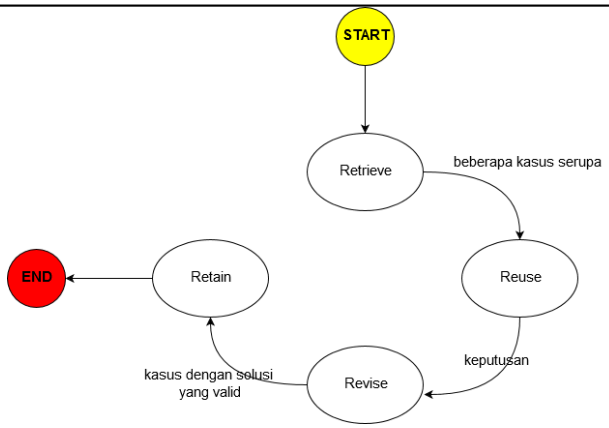
Kumpulan kasus-kasus ini akan disimpan ke dalam suatu database. Untuk itu diperlukan suatu pemodelan terhadap kasus agar dapat disimpan kedalam database [6]. Berdasarkan dari informasi yang didapat dari kumpulan kasus-kasus tersebut, kriteria-kriteria atau atribut-atribut yang penting yakni "Tipe" dari mesin tersebut sebagai contoh DC07, DC32, dan lainnya , "Seri" dari tipe tersebut semisal *Full Gear*, *All Floors*, dan sebagainya, "Usia" dari mesin tersebut seberapa lama mesin itu semenjak dibeli, "Masalah" yang terjadi, "Perbaikan yang dicoba" , serta "Solusi". Untuk masalah, perbaikan yang dicoba, dan solusi dapat memiliki lebih dari satu nilai yang akan dipisahkan dengan tanda (,). Untuk kriteria masalah tidak boleh kosong karena menjadi fitur yang sangat penting tetapi boleh bernilai sama. Sedangkan untuk kriteria "Tipe", "Seri", "Usia", "Perbaikan yang dicoba", dan "Solusi" boleh tidak memiliki nilai atau *null*.

Tabel yang akan digunakan untuk database hanya satu yakni tabel untuk menyimpan semua data kasus. Atribut yang akan digunakan dalam tabel sesuai dengan desain diatas. Pada atribut masalah yang akan diisi dengan satu atau lebih masalah yakni menggunakan tanda (,) sebagai pemisah. Hal ini memang terlihat tidak terlalu efektif, terutama dalam pengambilan data. Oleh karena itu, daripada membuat atribut yang banyak tetapi akan bernilai kosong, maka lebih baik membuat satu atribut saja. Hal ini berlaku juga untuk atribut perbaikan yang dicoba maupun solusi.

*2.3 Case base reasoning*

Sedikit pengetahuan mengenai *case base reasoning* sendiri. *case base reasoning* adalah suatu metode untuk menyelesaikan suatu masalah dengan melihat kembali kumpulan masalah-masalah yang telah diselesaikan pada waktu lampau. Dengan memperhitungkan kesamaan dari setiap kasusnya. *Case base reasoning* terdapat 2 jenis yakni untuk pemecahan masalah dan untuk interpretasi [5]. Dengan anggapan bahwa masalah-masalah yang serupa juga memiliki kesimpulan atau solusi yang serupa pula.

*Case base reasoning* yang akan digunakan pada penelitian ini khususnya adalah metode 4R yakni *Retrieve, Reuse, Revise,* dan *Retain*. *Retrieve* berarti mencari kasus-kasus yang serupa dengan kasus baru yang akan dicari, *Reuse* berarti menentukan apakah solusi dari kasus lama dapat digunakan solusinya untuk kasus yang baru, *Revise* berarti membetulkan solusi jika diperlukan, *Retain* berarti menentukan apakah kasus yang baru tersebut dapat dimasukkan ke kumpulan kasus-kasus[6]. Dengan demikian perlu merancang secara khusus setiap tahapan berdasarkan kerangka dasar dari *case base reasoning* ini. Agar hasil yang didapat bisa maksimal. Gambar 1 berikut adalah gambaran untuk metode 4R.




*Gambar 1. Diagram 4R pada CBR*

## 3. HASIL DAN PEMBAHASAN

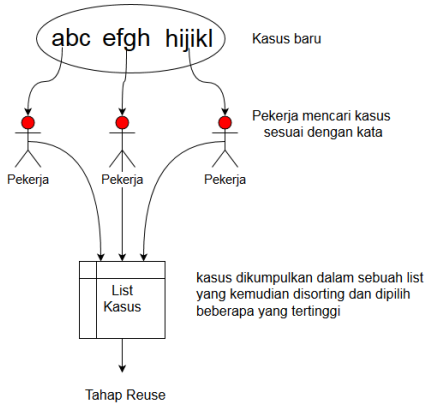
### *3.1. Tahapan Retrieve*

Proses *Retrieve* adalah proses paling penting dan paling pertama dalam *case base reasoning*. Hal ini dikarenakan tanpa adanya proses pengambilan data, maka proses berikutnya tidak dapat dilaksanakan. Oleh karena itu dalam mendasain suatu *case base reasoning*, membuat suatu metode pengaksesan data yang cepat menjadi suatu keunggulan pada aplikasi *case base reasoning* tersebut. Pada tahapan ini, data akan dipilih yang dianggap mirip dengan kasus yang baru dari kumpulan data kasus yang ada pada aplikasi.

Pada umumnya, dan kebanyakan penelitian tentang *case base reasoning*, untuk tahapan ini digunakan metode *Nearest Neighbour* yang dimana mengambil data-data yang terdekat berdasarkan urutan *indexing* pada database. Sangat disayangkan bahwa metode tersebut tidak dapat diterapkan pada model data penelitian ini. Hal ini karena, komponen utama penentuan solusi adalah berdasarkan dari kriteria masalah. Serta atribut masalah dalam tabel hanya berbentuk satu buah nilai. Sehingga *indexing* tidak dapat berguna dalam database ini.

Untuk itu, peneliti menggunakan sedikit prinsip dari algoritma genetika untuk membantu melakukan tahapan *retrieve* ini. Seperti algoritma genetika yang menerapkan prinsip melakukan perhitungan kecil dalam menemukan solusi, hanya saja prinsip yang diambil adalah jumlah pekerja yang digunakan. Sehingga untuk dapat melakukan *retrieve* dengan cepat, maka pekerjaan tersebut dilakukan sekaligus oleh banyak pekerja.

Dalam penelitian ini, untuk mendapatkan masalah yang serupa yakni berdasarkan atribut masalah. Setiap kata yang terdapat pada atribut masalah memiliki kesamaan tersendiri. Sehingga peneliti menetapkan untuk menggunakan atribut masalah sebagai indikator untuk mendapatkan data kasus-kasus yang serupa. Hal ini dapat dilakukan dengan mencari kesamaan antar kata pada masalah yang terdapat pada kasus baru. Semisal jika terdapat suatu kasus baru dengan masalah "Mesin tidak hidup" maka aplikasi akan membagi pekerja (worker) sebanyak 3 (sesuai dengan jumlah kata pada masalah) dan kemudian masing-masing pekerja tersebut akan mencari data kasus-kasus yang memiliki 3 kata tersebut. Setelah itu, pekerja-pekerja akan saling berkumpul untuk mendapatkan sebuah list yang berisi tentang ID dari pada kasus-kasus yang telah didapat oleh pekerja-pekerja, sehingga tidak ada ID kasus yang muncul 2 kali dalam list tersebut. Berikut adalah gambaran proses.




*Gambar 2. Proses Retrieve dengan Konsep Worker*

Setelah mendapatkan list kasus-kasus yang memiliki kata-kata tersebut. Akan dilakukan sistem ranking terhadap list tersebut. Hal ini dilakukan dengan menghitung jumlah kata yang sama antara kasus-kasus pada list dengan kata-kata pada kasus yang baru. Setelah dijumlahkan maka akan dipilih kasus pada list yang memiliki kesamaan paling banyak dengan kasus baru. Akan dipilih sebanyak 7 tertinggi untuk dikirim ke tahapan selanjutnya. Akan tetapi, jika pada urutan selanjutnya memiliki nilai yang sama dengan urutan ke-7, maka data selanjutnya akan diambil juga dan seterusnya.

### *3.2. Tahapan Reuse*

Pada tahapan *reuse*, data-data yang telah dikelompokkan dan dianggap cukup dekat dengan kasus baru akan diperhitungkan lebih dalam lagi untuk mendapatkan solusi terbaik. Hal ini dilakukan dengan memperhitungkan nilai kesamaan atau *similarity* dari atribut-atribut yang ada dalam model data. Tidak berarti bahwa setiap atribut memiliki bobot yang sama, setiap atribut yang menentukan suatu kesamaan atau *similarity* akan berbeda-beda.

Kriteria "Tipe" yang bernilai tentang tipe dari vacuum cleaner Dyson tidak memiliki bobot yang terlalu tinggi karena hanya dari tipe saja tidak dapat menentukan suatu solusi untuk suatu masalah. Hal ini sangat jelas karena suatu tipe tidak dapat menggambarkan apa-apa. Akan tetapi, dapat membantu dalam mengelompokkan masalah pada vacuum cleaner berdasarkan dari tipe vacuum cleaner tersebut karena memiliki kesamaan jenis produk. Sehingga dianggap tipe yang sama memiliki masalah dan kemungkinan solusi yang mirip juga. Bobot untuk kriteria ini adalah 1. Pemberian nilai bobot ini didasarkan pada tingkatan kepetingan yakni dari 1 sampai 5.

Kriteria "Seri" bersifat lebih fokus dari pada kriteria tipe. Kriteria ini adalah turunan dari kriteria tipe, dimana jenisnya akan lebih sama daripada kriteria tipe. Dengan anggapan penggunaan seri yang sama dapat lebih spesifik antara bagian-bagian dalam vacuum cleaner tersebut. Bobot untuk kriteria ini diberikan oleh peneliti adalah 1,5.

Kriteria "Usia" adalah waktu dalam satuan tahun dimana umur dari vacuum cleaner semenjak pertama kali dibeli. Dengan adanya kriteria ini, dapat menghubungkan kesamaan antar kasus oleh umur. Dengan anggapan, jika suatu vacuum cleaner yang berumur 5 tahun biasa akan memiliki masalah yang cukup mirip dengan yang berumur 4 tahun seperti misalnya, berkarat dan sebagainya. Akan tetapi kriteria ini cukup abstrak dan tidak memiliki kepastian tinggi, sehingga Bobot untuk kriteria ini adalah 0,5.

Kriteria "Masalah" adalah kriteria inti yang berisi sumber utama untuk perbandingan. Masalah-masalah yang sama kemungkinan besar akan memiliki solusi yang sama. Terlebih mesin vacuum cleaner memeliki sistem yang cukup mirip karena memiliki satu tujuan yaitu untuk menyedot debu atau kotoran. Sehingga bobot yang diberikan peneliti untuk kriteria ini adalah 5,0.

Kriteria "Perbaikan yang dicoba" merupakan suatu kriteria yang dibuat peneliti karena peneliti melihat kebanyakan orang terkadang cukup kreatif dalam mengotak-atik barang mereka. Dengan demikian, ada kemungkinan bahwa apa yang mereka coba tersebut dapat menjadi suatu

indikator dalam menentukan perbaikan selanjutnya hingga menemukan solusi yang benar-benar dapat memperbaiki mesin vacuum cleaner mereka. Oleh karena itu, peneliti memberikan bobot 3,0.

Setelah menentukan bobot masing-masing kriteria, untuk mengetahui mana yang dapat diambil dari data kasus hasil *retrieve* akan dilakukan perhitungan kesamaan atau *similarity*. Untuk perhitungan ini, peneliti menggunakan prinsip dari Negny Ste´phane yang mengatakan bahwa cara menghitung nilai *similarity* adalah dengan menggunakan *global similarity* dan *local similarity*[3]. Maksud dari pernyataan ini adalah pertama-tama dicari nilai *local similarity* dengan membandingkan kriteria antara kasus yang baru dan kasus-kasus yang lama. Setelah itu barulah dicari *global similarity* (lihat persamaan 1) dengan menjumlahkan setiap *local similarity* setiap kriteria yang ada pada kasus baru. Untuk kriteria perbaikan yang dicoba dan usia, jika kasus yang baru tidak memiliki nilai untuk kriteria tersebut, maka bobotnya tidak akan memberatkan pada jumlah bobot akhir. Kemudian hasil penjumlahan tersebut akan dibagi dengan total bobot (W) hingga didapatlah nilai akhir untuk penentuan.

$$Global\ Similarity = \frac{\sum_{i=1}^{n}(W_i \times localsimilarity_i)}{\sum_{i=1}^{n} W_i} \quad (1)$$

Sedangkan untuk menghitung setiap nilainya pada *local similarity* pada setiap kriteria akan ditentukan berdasarkan jenis kriteria tersebut. Metode yang akan banyak dipakai untuk menghitung nilai *local similarity* ini adalah dengan menggunakan String Matching. Yakni dengan membandingkan setiap kata antara kedua kasus. Cara melakukan String Matching ini adalah dengan memisahkan setiap kata pada kasus dan kemudian membandingkan 1 kata dengan kata yang lain pada kasus yang dibandingkan. Jika kata tersebut menemukan kata yang sama pada kasus yang dibandingkan, maka kata tersebut akan dihilangkan agar tidak terjadi perulangan kata yang banyak. Nilai pembagi untuk menormalisasikan nilai *local similarity* ini adalah dengan jumlah kata yang ada pada kasus yang baru. Sehingga nilai tertinggi adalah 1 jika sama semua dan akan bernilai 0 jika berbeda semua. Adapun kriteria yang menggunakan metode ini adalah kriteria seri, kriteria masalah, dan kriteria perbaikan yang dicoba.

Sedangkan untuk menghitung nilai *local similarity* pada kriteria tipe dilakukan dengan mengecek secara lansung, jika tipe berbeda maka bernilai 0 dan jika tipe sama maka akan bernilai 1. Kemudian untuk kriteria usia akan dibandingkan antara nilai usia kasus baru dan kasus lama. Dimana nilai yang kecil akan dibandingkan dengan nilai yang besar sehingga didapatkan lah perbandingan perbedaannya.

### *3.3. Tahapan Revise*

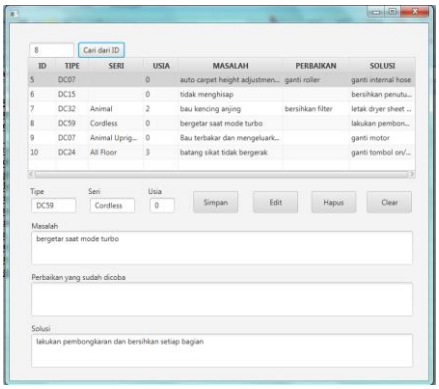
Pada tahapan ini, peneliti memutuskan untuk menggunakan cara manual dalam merevisi masalah. Dengan kata lain, proses revisi ini akan dikerjakan oleh seorang pakar atau ahli, ataupun ketika ada yang menemukan solusi tersebut. Data kasus-kasus yang baru yang tidak dapat menyelesaikan masalah akan tetap disimpan dan menunggu sampai ada perbaikan untuk masalah tersebut.

### *3.4. Tahapan Retain*

Pada tahapan terakhir dari metode 4R yakni *retain* adalah bagaimana kasus baru tersebut akan masuk kedalam kumpulan kasus-kasus untuk dipakai berikutnya. Kasus yang tersebut akan diambil bilamana pengguna mendapatkan hasil penyembuhan dari solusi tersebut. Sehingga kasus tersebut akan disimpan dengan menggisikan solusi seperti yang aplikasi cari. Apabila pengguna tidak mendapatkan manfaat dari solusi tersebut, maka kasus akan disimpan dengan tanpa solusi. Yang dimana akan direvisi oleh ahli atau pakar pada waktu lain.

### *3.5. Perancangan Aplikasi*

Setelah menentukan *inference engine* yakni bagian dalam dan utama dari aplikasi ini. Tahap selanjutnya adalah menentukan rancangan dari aplikasi itu sendiri. Aplkasi *Case base reasoning* ini akan dibuat dengan menggunakan bahasa pemrograman Java khususnya ekspansi JavaFX. Sedangkan untuk databasenya digunakan SQLite untuk menyimpan semua data. Database ini berfungsi seperti otak seorang pakar. Alasan menggunakan bahasa pemrograman Java karena merupakan pemrograman objek yang bagus sehingga mudah untuk mengatur segala keperluan yang menyangkut objek. Serta fitur thread pada Java juga mudah untuk digunakan dalam membantu pembuatan aplikasi ini. Fungsi thread disini adalah untuk membantu mewujudkan tahapan *retrieve* agar dapat menyaring data berdasarkan data model yang telah dirancang. Berikut adalah hasil tampilan user interface (gambar 3).




*Gambar 3. Tampilan aplikasi*

Pengujian Aplikasi

Untuk menguji kesesuaian hasil, maka diperlukan pengujian data kasus (test cases) [10]. Data tersebut dapat dilihat sebagai berikut.

*Tabel 1. Data kasus Uji Coba*

| Tipe | Seri | Usia | Masalah | Perbaikan yang dicoba | Solusi |
|---|---|---|---|---|---|
| DC07 | | | Auto carpet height adjustment tidak bekerja | ganti roller | ganti internal hose |
| DC15 | | | Tidak Menghisap | | bersihkan penutup diatas casing filter motor |
| DC32 | Animal | 2 | Bau kencing anjing | bersihkan filter | letak dryer sheet pada kantong plastik bersih dan biarkan hingga 5 sampai 10 menit |
| DC59 | Cordless | | bergetar saat mode turbo | | lakukan pembongkaran dan bersihkan setiap bagian |
| DC07 | Animal Upright Cleaner | | Bau terbakar dan mengeluarkan suara aneh | | ganti motor |
| DC24 | All Floor | 3 | batang sikat tidak bergerak | | ganti tombol on/off pada batang sikatnya |

Contoh kasus yang diambil adalah sebuah mesin dyson dengan tipe DC07, tidak diketahui seri, tidak diketahui usia, memiliki masalah “Bau Anjing”, dan perbaikan yang telah dicoba adalah “bersihkan filter dan hose”. Secara perhitungan maka sebagai berikut.

*Tabel 2. Hasil retrieve*

| No | Tipe | Seri | Usia | Masalah | Perbaikan yang dicoba | Solusi |
|---|---|---|---|---|---|---|
| 1 | DC32 | Animal | 2 | Bau kencing anjing | bersihkan filter | letak dryer sheet pada kantong plastik bersih dan biarkan hingga 5 sampai 10 menit |
| 2 | DC07 | Animal Upright Cleaner | 0 | Bau terbakar dan mengeluarkan suara aneh | | ganti motor |

Berdasarkan dari kata yang terdapat pada masalah maka terpilihlah 2 buah kasus yang dianggap mirip dengan kasus baru. Setelah itu akan dilakukan perhitungan *local* similariy dan akan dikalikan dengan bobot kriteri, selanjutnya diteruskan ke penjumlahan untuk mendapatkan *global similarity* yang akan digunakan untuk mendapatkat keputusan.

*Tabel 3. Hasil perhitungan local similarity*

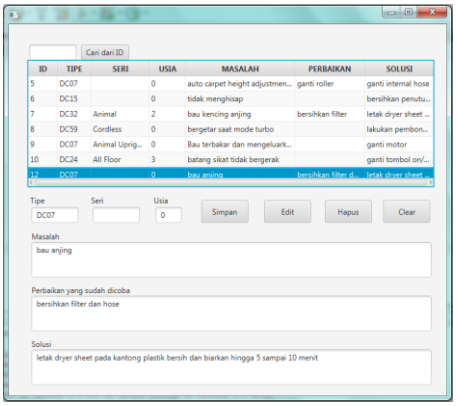
| Kriteria | Jumlah kata yang sama | jumlah kata kasus baru | Nilai local similarity |
|---|---|---|---|

| | Kasus 1 | Kasus 2 | | kasus 1 | kasus 2 |
|---|---|---|---|---|---|
| Masalah | 2 | 1 | 2 | 1 | 0,5 |
| Perbaikan yang dicoba | 2 | 0 | 4 | 0,5 | 0 |
| Tipe | 0 | 1 | 1 | 0 | 1 |

$$global\ similarity\ kasus\ 1 = \frac{(1\times0)+(5\times1)+(3\times0{,}5)}{1+1{,}5+5+3}$$

$$global\ similarity\ kasus\ 2 = \frac{(1\times1)+(5\times0{,}5)+(3\times0)}{1+1{,}5+5+3}$$

Maka dari itu didapatlah *global similarity* kasus 1 sebesar 5,1425 dan *global similarity* kasus 2 sebesar 3,5. Sehingga keputusan yang diperoleh akan memilih kasus 1 sebagai solusi yakni "letak dryer sheet pada kantong plastik bersih dan biarkan hingga 5 sampai 10 menit "karena memiliki nilai terbesar dari antara kasus-kasus lama dan memiliki nilai lebih dari 50% total bobot.




*Gambar 4. Tampilan Hasil test case*

## 4. KESIMPULAN

Kesimpulan pada penelitian ini adalah semua fungsi user interface pada aplikasi bekerja dengan baik serta aplikasi dapat mencari solusi berdasarkan uji coba terhadap kasus baru hanya saja tidak menampilkan hasil perhitungan. Metode case-based reasoning dengan menggunakan konsep worker pada tahapan retrieve dan perhitungan local dan global similarity sudah tepat penerapannya untuk melakukan troubleshooting vacuum cleaner

## 5. SARAN

Saran untuk penelitan berikutnya adalah menggunakan threads lebih maksimal, menambahkan kriteria pada model data agar dapat mendapatkan hasil yang lebih baik, menggunakan data kasus yang lebih banyak dan menambahkan metode perhitungan heuristic untuk hasil yang lebih handal.

## DAFTAR PUSTAKA

[1] Wang, S. L., & Hsu, S. H. (2004). A Web-based CBR knowledge management system for PC troubleshooting. *The International Journal of Advanced Manufacturing Technology*, *23*(7-8), 532-540.

[2] Devaney, M., & Cheetham, W. (2005, May). Case-Based Reasoning for Gas Turbine Diagnostics. In *FLAIRS conference* (pp. 105-110).

[3] Wan, S., Li, D., Gao, J., & Li, J. (2019). A knowledge-based machine tool maintenance planning system Using case-based reasoning techniques. *Robotics and Computer-Integrated Manufacturing*, *58*, 80-96.

[4] Stéphane, N., & Hector, R. (2010). Effective retrieval and new indexing method for case-based reasoning: application in chemical process design. *Engineering Applications of Artificial Intelligence*, *23*(6), 880-894.

[5] Salem, A. B. M. (2007). Case based reasoning technology for medical diagnosis. *World academy of science, engineering and technology*, *31*(2007), 9-13.

[6] Dahouk, A. W., & Abu-Naser, S. S. (2018). A Proposed Knowledge Based System for Desktop PC Troubleshooting.

[7] Mansar, S. L., Marir, F., & Reijers, H. A. (2003). Case-based reasoning as a technique for knowledge management in business process redesign. *Electronic Journal on Knowledge Management*, *1*(2), 113-124.

[8] Rahman, A., Slamet, C., Darmalaksana, W., Gerhana, Y. A., & Ramdhani, M. A. (2018, January). Expert System for Deciding a Solution of Mechanical Failure in a Car using Case-based Reasoning. In *IOP Conference Series: Materials Science and Engineering* (Vol. 288, No. 1, p. 012011). IOP Publishing.

[9] Khosravani, M. R., Nasiri, S., & Weinberg, K. (2019). Application of case-based reasoning in a fault detection system on production of drippers. *Applied Soft Computing*, *75*, 227-232.

[10] Khatib, E. J., Gómez-Andrades, A., Serrano, I., & Barco, R. (2018). Modelling LTE solved troubleshooting cases. *Journal of Network and Systems Management*, *26*(1), 23-50.