

**MODUL Manajemen Pengetahuan**

**(KODE CSM410)**

**MODUL SESI 12**

**TOPIK Membangun Chatboat**

**DISUSUN OLEH**

**Riya Widayanti**

**UNIVERSITAS ESA UNGGUL**

**2019**

**SUBTOPIK 1 TOPIK SESI INI**

1. **Kemampuan Akhir Yang Diharapkan**

Setelah mempelajari modul ini, diharapkan mahasiswa mampu :

* 1. Memahami penerapan KM pada beberapa kasus perusahaan
  2. Membuat usulan alternative penerapan dengan mengambil kasus di sekitar lingkungan mahasiswa baik dengan menggunakan CBR atau RULE
  3. Penengembangan chat boat, system pakar

1. **Uraian dan Contoh**

Chatbot Chatbot merupakan program komputer yang dirancang untuk dapat melakukan interaksi dengan manusia melalui pesan teks, maupun suara. Chatbot biasanya juga dibekali dengan kecerdasan buatan dan pemrosesan bahasa alami yang membuatnya menjadi program komputer yang cerdas dan dapat menjawab pertanyaan yang diberikan oleh manusia. Chatbot dibangun sesuai dengan topik yang sudah dimodelkan dalam basis pengetahuan. Banyak chatbot yang sudah ada dibangun sesuai dengan topik dan permasalahan yang ingin dipecahkan oleh seseorang untuk keperluan pribadi ataupun keperluan bisnis, sebagai contoh adalah chatbot yang menangani pemesanan pizza. Di dalam chatbot tersebut telah ditanamkan model pengetahuan untuk menjawab pertanyaan-pertanyaan yang sesuai dengan konteks yang telah disusun. Chatbot terdiri dari tiga kombinasi, di mana ketiga kombinasi inilah yang membentuk sebuah chatbot (Guzman & Ines, 2016), di antaranya adalah: a. User Interface User interface dalam chatbot ini sendiri adalah jembatan antara chatbot dan user saling berinteraksi. Melalui aplikasi pesan berbasis text. User Interface haruslah dapat memberikan pengalaman yang lebih baik kepada user ketika berinteraksi dengan Chatbot. b. Artificial Intelligence (Kecerdasan Buatan) AI atau Artificial Intelligence akan membuat chatbot mengerti dan memahami setiap interaksi yang terjadi dengan user. Chatbot menangani pemecahan masalah melalui aturan yang telah ditentukan sebelumnya di pohon keputusan. c. Integrasi Integrasi dengan sistem lainnya akan menambah kekayaan fitur yang terdapat di dalam suatu chatbot. Dengan mengintegrasikan chatbot ke sistem yang lain dapat menyediakan informasi tambahan. Dengan cara ini chatbot mampu memberikan informasi yang lebih kaya kepada user, seperti pada penelitian kali ini yang akan mengimplementasikan chatbot pada order management system. Di keperluan bisnis chatbot dapat membantu pada pemecahan permasalahan yang berkaitan dengan komunikasi dengan pelanggan untuk meningkatkan pelayanan dan pengalaman dalam hal berkomunikasi, chatbot sangat lah efektif digunakan dalam permasalahan yang fokus dan spesifik serta dapat diprediksi ( Guzman & Ines, 2016). Pada perkembangan dunia bisnis yang sangat cepat, chatbot menjadi salah satu alternatif solusi, ini akan memudahkan pelanggan menjangkau dan melakukan interaksi dengan bisnis

**Platform API.AI**

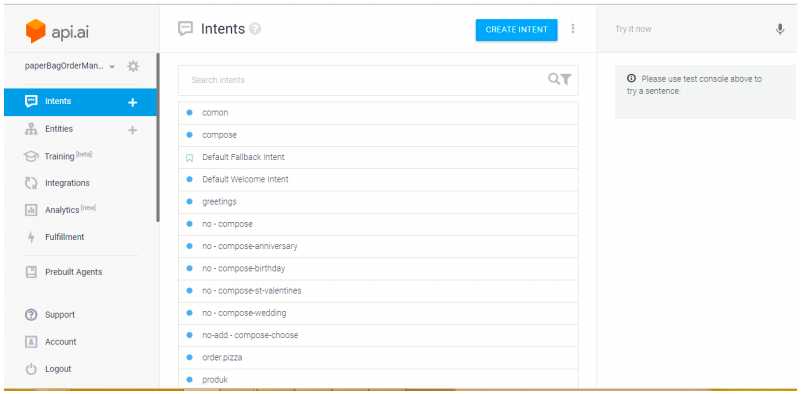
API.AI adalah sebuah platform yang menyediakan layanan NLP (Natural Language Processing) dan NLU (Natural Language Understanding). Layanan ini digunakan untuk membuat chatbot lebih cerdas dan dapat memahami maksud dari apa yang ditanyakan oleh user. Natural language Processing adalah salah satu disiplin ilmu dari Artificial Intelligence yang fokus terhadap interaksi manusia dan komputer melalui bahasa alami yang manusia gunakan. Dalam NLP tujuan yang ingin dicapai adalah kemampuan sebuah sistem NLP memiliki pengetahuan bahasa alami baik dari susunan kalimat, arti dari kata tersebut dan maksud dari sebuah kalimat. Sedangkan NLU sendiri adalah merupakan sub bidang dari NLP, di mana fokus tujuan dari NLU itu sendiri adalah untuk melakukan pemahaman terhadap suatu kalimat dan melakukan analisis semantik. API.AI ini sendiri tidak mendukung domain pengetahuan bahasa Indonesia namun memiliki banyak sekali domain pengetahuan dalam bahasa inggris. Domain adalah koleksi pengetahuan dan struktur data. Di dalam API.AI sendiri telah banyak basis pengetahuan yang sudah tertanam di sistem layanan API.AI ini. Koleksi pengetahuan yang terdapat pada layanan ini di antaranya adalah entitas musisi, bandara, dll. Namun tiadanya dukungan bahasa indonesia di dalam layanan ini bukan berarti layanan ini tidak dapat digunakan oleh peneliti yang notabene berbahasa indonesia. Untuk bagaimana penulis menggunakan layanan ini akan dibahas di Bab 3 Metodologi Penelitian pada bagian perancangan chatbot.

**Sejarah Singkat API.AI**

API.AI yang dulunya adalah Speaktoit adalah pengembang teknologi interaksi komputer dan manusia. Fokus teknologi yang digunakan adalah dalam bidang natural language processing atau NLP. Perusahaan ini telah mengembangkan beberapa produk yang memiliki fitur utama melakukan tugas tanya jawab dengan bahasa alami antara manusia dan komputer. Pada bulan September 2014 Speaktoit memperkenalkan sebuah platform yang diberi nama API.AI. Platform inilah yang banyak digunakan oleh pengembang pihak ketiga untuk membuat sebuah chatbot. Selain itu API.AI menyediakan kepada pengembang fitur-fitur lainnya yaitu sebuak SDK (Software Development Kit) yang memiliki fungsi dapat menambahkan pengenalan suara, pemahaman bahasa alami dan konversi text-to-speach untuk aplikasi Android, dan IOS. API.AI juga memiliki halaman website yang digunakan untuk melakukan pengujian skenario yang dapat digunakan para developer untuk menguji skenario 9 yang telah dirancang sebelumnya. Pada bulan September tahun 2016 Google membeli perusahaan ini. Google menggunakan layanan ini untuk asisten virtual Google.

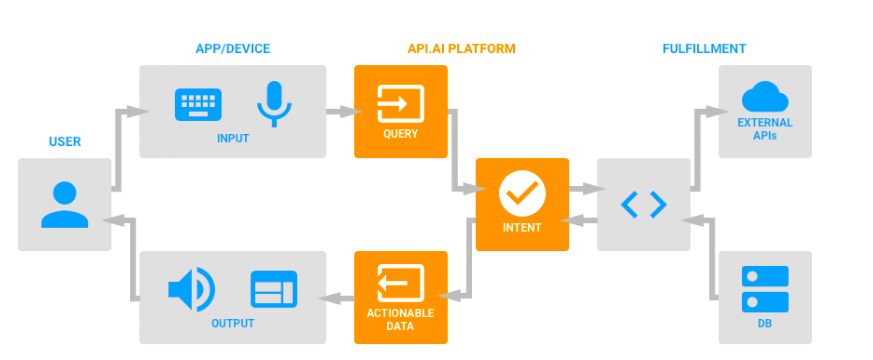
**Lingkungan Pengembangan**

Untuk dapat menggunakan layanan ini, maka penulis perlu melakukan observasi dan belajar untuk mengetahui lingkungan pengembangan dari platform API.AI. Penulis juga melihat beberapa tutorial untuk dapat membangun sebuah chatbot yang terintegrasi dengan platform ini. Ada beberapa hal yang perlu penulis ketahui untuk dapat menggunakan layanan ini. Langkah pertama yang penulis lakukan adalah untuk mengetahui bagaimana cara kerja dasar untuk menggunakan layanan ini. Gambar 1 menunjukkan halaman dashboard dari platform API.AI



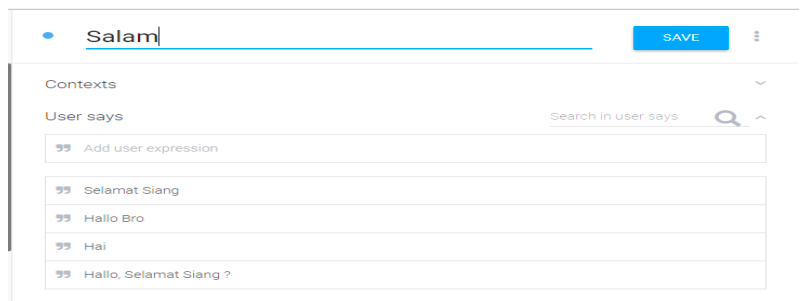
Gambar 1. GAmbar Dashboard API.AI

Gambar 1 di atas merupakan halaman dashboard utama dari platform API.AI ini, di dashboard tersebut developer dapat merancang Conversational Flow dari permasalahan yang ingin dipecahkan Proses awal yang dilakukan adalah dengan adanya request yang datangnya dari user dengan melakukan permohonan ke platform, lalu sistem didalam platform ini akan memberikan response ke user yang terkait bahwa permohonan untuk menggunakan layanan diterima, user disini dapat dikatakan sebagai developer yang merancang sebuah chatbot. Gambar 2 menunjukkan illustrasi mengenai platform ini. 10



Gambar 2 Alur Kerja Platform API.AI

Gambar 2 di atas merupakan gambaran umum mengenai cara kerja di dalam platform API.AI ini. Pertama-tama dimulai dari user yang mengirimkan request berupa teks ataupun suara ke dalam platform API.AI ini, kemudian query atau permintaan tadi akan diproses di dalam intents untuk memetakan permintaan tersebut dan tindakan apa yang harus dilakukan. Selanjutnya apabila terdapat informasi tambahan yang akan diberikan sebagai response, platform ini akan mengirimkan fullfillment, ini akan mendapatkan informasi dari sumber daya luar. Lalu user akan mendapatkan jawaban dari pertanyaan tersebut. Pada langkah awal developer akan membuat sebuah Agent yang akan digunakan sebagai modul. End-user atau user yang akan memanfaatkan layanan chatbot yang sudah terintegrasi dengan platform API.AI akan menanyakan sesuatu. Permintaan dari end-user ini akan diteruskan ke dalam Agent yang telah dibuat oleh developer. Di dalam Agent atau modul inilah permintaan yang datang akan diproses. Agar Agent dapat memahami maksud dari permintaan yang dikirim, Agent perlu memiliki beberapa sampel yang berkaitan dengan pertanyaan tersebut. Oleh karena itu developer perlu mendefinisikan terlebih dahulu persamaan pertanyaan yang akan memiliki kesamaan dengan pertanyaan yang dikirim end-user ke dalam sebuah Intent. Semakin banyak variasi pertanyaan yang didefinisikan maka akan sangat membantu sistem untuk menentukan jawaban yang tepat untuk dikirimkan sebagai jawaban pertanyaan. Gambar 3 menunjukkan cara pendefinisian pertanyaan didalam sebuah intent.



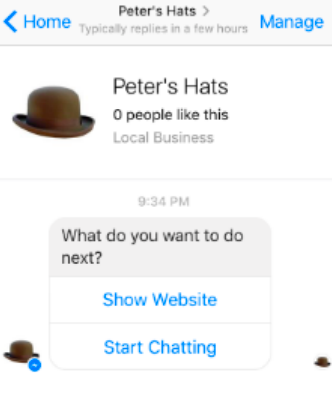
Gambar 3. Pembuatan Intens

Gambar 3 di atas merupakan contoh dalam pembuatan sebuah intent yang diberi isian “Salam” , lalu setelah itu penulis membuat sebuah kata-kata yaitu “Selamat Siang”, ”Hallo Bro”, ”Hai”, “Hallo, Selamat Siang?”. Kata-kata tersebut merupakan sebuah perkiraan pertanyaan yang memiliki kemungkinan besar end-user tanyakan. Intent ini dibuat dengan tujuan untuk menampung segala pertanyaan yang berkaitan dengan salam atau sapaan, apabila end-user mengirim pertanyaan salam atau sapaan, dan apabila sesuai dengan pertanyaan yang sudah ada di basis data seperti di atas maka Agent akan dengan mudah untuk menentukan maksud dari pertanyaan tersebut dan dapat mengirim jawaban dengan tepat. Apabila pertanyaan sapaan yang dikirim belum ada di basis data tersebut developer dapat mendefinisikan ke dalam daftar tersebut. Hal yang telah dijelaskan sebelumnya belum cukup untuk membuat Agent dapat memberikan jawaban yang tepat, namun untuk lebih memahami lebih lanjut tentang platform ini maka dari itu penulis akan menjabarkan istilah-istilah yang penulis pilih dan penting di dalam platform ini.

1. Agents Agents dapat juga disamakan dengan sebuah modul. Agents inilah yang akan mengelola seluruh conversation flow. Di dalam Agents ini terdapat sekumpulan intents, dan entities.
2. Intents Merupakan sebuah tempat untuk memetakan pertanyaan apa yang dikirim oleh pengguna dan tindakan apa yang harus dilakukan. Tujuan dari intents adalah mendifinisikan tata bahasa percakapan dan tugas apa yang harus dilakukan saat pengguna menggunakan frasa tertentu.
3. Entities Entities adalah sebuah alat yang sangat kuat untuk mengindentifikasi nilai yang berguna dari sebuah masukan bahasa alami dalam hal ini adalah pertanyaan yang dikirim oleh user. Di dalam platform terdapat 3 jenis entities, yaitu System Entities, Developer Entities, dan User Entities. System Entities merupakan entities yang sudah dibenamkan di dalam sistem, artinya entities tersebut sudah tersedia, Developer Entities merupakan entities yang dibuat dan didefinisikan oleh developer, User Entities adalah yang didefinisikan untuk setiap user pada setiap permintaan.

**Facebook Messenger**

Facebook Messenger adalah layanan pesan instan yang dibuat oleh Facebook. Aplikasi ini memiliki beberapa fitur-fitur yang membuatnya menjadi sangat interaktif dan menarik untuk digunakan oleh user. Aplikasi ini dapat bertukar pesan antar sesama pengguna, bertukar foto atau video, stiker, audio, file, dan juga layanan panggilan suara dan panggilan video. Aplikasi ini telah tersedia dalam bentuk standalone apps untuk berbagai macam sistem operasi di antaranya Windows 10, Android, dan IOS. Tidak hanya itu aplikasi Messenger juga dapat diakses melalui web browser. Messenger memiliki 600 juta pengguna pada tahun 2015. Jumlah tersebut terus meningkat sampai padai bulan April 2017 pengguna aplikasi ini mencapai 1,2 milyar pengguna. Selain digunakan untuk mengirim pesan antar satu user ke user lainnya, Messenger telah menyediakan fasilitas untuk sebuat bot atau robot chat yang dapat digunakan dan dibuat oleh para developer dengan memanfaatkan layanan-layanan yang ada di dalam Mesenger ini. Aplikasi Messenger ini juga menyediakan layanan untuk membuat user interface pesan, sehingga tidak menimbulkan kesan monoton yang hanya diisi dengan teks melainkan berupa grafik yang menarik. Berikut ini akan dijelaskan beberapa fitur-fitur yang dapat digunakan oleh developer untuk membangun sebuah bot menarik dengan menggunakan layanan restfull api yang sudah tersedia dari layanan Messenger ini. Di antaranya adalah fitur template, yang disebut sebagai Generic Message atau pesan terstruktur. Template yang tersedia di dalam layanan ini memiliki beragam jenis, di antaranya adalah template tombol di mana fitur ini akan menampilkan tombol di dalam antarmuka Messenger. Gambar 4 menunjukkan contoh template tombol.

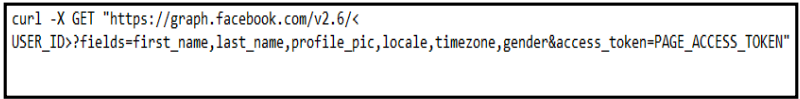


GAmbar. 4Template Control pada Facebook

Gambar 4 di atas merupakan contoh penggunaan template tombol yang dapat digunakan developer untuk membuat user interface yang interaktif dan menarik user.

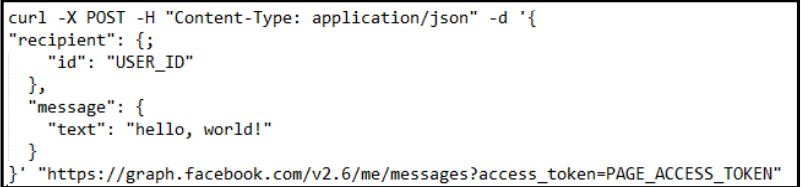
**Penggunaan API**

(Application Program Interface) Messenger Untuk dapat membuat sebuah bot di Messenger terlebih dahulu penulis mempelajari bagaimana menggunakan Application Program Interface yang telah disediakan oleh Messenger. Diantara yang penulis pelajari adalah bagaimana mendapatkan informasi pengguna beserta bagaimana cara mengirimkan pesan kepada pengguna. Kode di bawah ini menunjukan cara untuk mengakses informasi pengguna dengan USER\_ID yang telah ditentukan sebelumnya. Dengan menggunakan HTTP request untuk mengakses API, keluaran dari proses ini akan menampilkan informasi pengguna berupa nama depan, nama belakang, dan foto



*Gambar 5. Request 1*

Untuk mengirim sebuah pesan kepada user menggunakan HTTP Request adalah dengan cara mengirimkan permintaan ke layanan API Messenger. Gambar 6 ini adalah kode untuk mengirimkan pesan. Sebelumnya kode juga harus menyertakan USER ID dan Page Access Token.



*Gambar 6. Request 2*

**Penggunaan saat ini dan Kemajuan dalam Sistem Pakar dan Akuisisi Pengetahuan menggunakan Rule-based Reasoning (RBR**)

Representasi pengetahuan berbasis aturan adalah salah satu jenis yang paling banyak digunakan (Englemore dan Feigenbaum, 1993). Tapi apa sebenarnya Penalaran berbasis Rule? Crossman et al. (1995) menyatakan dengan memecah ini menjadi bagian-bagian individualnya, aturan adalah representasi pengetahuan tentang mana "pola informasi yang digunakan para ahli untuk membuat keputusan dan apa keputusan yang mengikuti." Penalaran berbasis aturan menawarkan seperangkat aturan yang berantai ke diberikan kesimpulan. Cara paling populer untuk mewakili pengetahuan ini adalah dengan menggunakan aturan "jika-maka". Crossman et al. nyatakan bahwa aturan ini dapat direpresentasikan dalam hubungan logis “p → q”. Dalam hubungan ini, "p" mewakili suatu kondisi atau serangkaian kondisi dan "q" mewakili suatu kesimpulan atau serangkaian kesimpulan. Himpunan relasi / kesimpulan ini, menurut Ignizio (1991), himpunan paling umum digunakan oleh sistem pakar, dan terutama dalam sistem pakar berbasis aturan. Strategi inferensi ini lebih dikenal sebagai "modus ponens" yang berarti bahwa jika A menyimpulkan B dan A benar, maka B benar.

Crossman et al. (1995) lebih lanjut menyatakan bahwa banyak algoritma yang berbeda telah dikembangkan untuk mendukung premis dasar penalaran berbasis aturan. Ada berbagai perbedaan dalam pendekatan ini dan semuanya berada dalam domain rekayasa pengetahuan.

Salah satu contoh yang diberikan oleh Crossman et al. adalah bahwa "aturan rantai maju memfasilitasi sintesis pemrograman, sedangkan aturan rantai belakang lebih cocok untuk analisis atau pencarian."

Jika jenis penalaran yang terlibat dalam domain yang menarik melibatkan penggunaan diagram alir atau pohon, maka penggunaan aturan adalah cara terbaik untuk melanjutkan. Namun, peraturan ini tidak; mewakili fakta atau data itu sendiri; mereka lebih mewakili alasan tentang fakta atau data. Sistem berbasis aturan menggunakan mesin inferensi yang merupakan algoritme yang mengatur apa yang dapat dilakukan aturan, kapan aturan itu diaktifkan atau dipicu, dan prioritas apa yang diberikan kepada masing-masing untuk dieksekusi. Aturan dalam sistem berbasis aturan juga dapat menghibur bentuk tertentu dari penalaran tidak pasti, misalnya, menambah atau mengurangi tingkat kepercayaan saat mengevaluasi hipotesis atau menyediakan mekanisme alternatif untuk menangani jalur penalaran lain (Crossman et al., 1995).

Perkakas berbasis aturan dan sistem pakar lainnya lebih dikenal sebagai cangkang. Keuntungan terbesar yang ditawarkan oleh sistem berbasis aturan adalah mereka memungkinkan pengguna untuk melihat aturan dalam format bahasa yang hampir alami dan memberikan penjelasan mengapa penjelasan dibuat.

Luger dan Stubblefield (2002) menyatakan bahwa upaya pertama untuk membangun sistem pakar menggunakan paradigma berbasis aturan, tidak mungkin sangat berhasil. Alasan utama untuk ini adalah karena ahli domain merasa sangat sulit untuk mengekspresikan pengetahuan diam-diam dalam istilah yang dapat digunakan untuk memecahkan masalah spesifik.

Akhirnya, Luger dan Stubblefield (2002) menyimpulkan bahwa skema representasi pengetahuan yang paling banyak digunakan untuk sistem pakar adalah berbasis aturan. Biasanya, aturan itu sendiri tidak akan memiliki kesimpulan tertentu tetapi akan ada beberapa tingkat kepastian bahwa kesimpulan itu akan berlaku jika kondisinya berlaku. Ada teknik statistik yang digunakan untuk menentukan kepastian ini. Sistem berbasis aturan, apakah mereka memiliki kepastian atau tidak, biasanya mudah dimodifikasi. Kepastian ini membuatnya relatif sederhana untuk memberikan jejak bermanfaat dari penalaran sistem. Jejak-jejak ini dapat digunakan dalam memberikan penjelasan tentang apa yang dilakukannya.

Perlu dicatat bahwa alasan berbasis aturan digunakan dalam lingkungan help desk terutama sebagai peningkatan teknologi berbasis kasus (Luger dan Stubblefield, 2002).

**Penggunaan saat ini dan Kemajuan dalam Sistem Pakar dan Akuisisi Pengetahuan menggunakan Case-based Reasoning (CBR)**

Case-based reasoning (CBR) adalah paradigma Artificial Intelligence (AI) untuk pemecahan masalah dan penggunaan kembali pengetahuan yang menggunakan contoh serupa sebelumnya untuk memecahkan masalah saat ini. Selanjutnya, CBR menarik kemampuannya untuk mencari ingatannya untuk solusi dan memperoleh yang baru tanpa harus memahami prinsip-prinsip yang mendasari domainnya (Kolodner, 1993).

Namun, untuk menjelaskan cara kerja CBR, kita harus terlebih dahulu memahami makna kasus. Watson (2002, hlm. 27) mendefinisikan case sebagai:

Kasus adalah catatan pengalaman yang mengandung pengetahuan, yang bisa eksplisit dan diam-diam. Misalnya, mereka dapat menjadi kasus dalam pengertian hukum; mereka dapat berupa riwayat kasus pasien dalam pengertian medis, rincian pinjaman bank, atau deskripsi situasi pemecahan masalah peralatan.

Watson melanjutkan dengan menggambarkan apa yang terdiri dari kasus hukum, riwayat kasus medis, pinjaman bank dan catatan pemecahan masalah. Pertama, ia menyatakan bahwa deskripsi terdiri dari "masalah hukum, gejala pasien, detail pinjaman, dan masalah peralatan". Dia menyimpulkan dengan menyatakan bahwa hasil atau solusi dari masing-masing uraian ini terdiri dari "putusan atau putusan, perlakuan, hasil pinjaman, dan perbaikan teknis".

Kolodner (1993) menyatakan bahwa suatu kasus lebih lanjut dapat digambarkan sebagai laporan dari suatu peristiwa, cerita, atau beberapa catatan yang biasanya terdiri dari masalah yang menggambarkan keadaan dunia ketika kasus itu terjadi dan solusi yang menyatakan solusi yang berasal dari masalah itu. Ini berarti bahwa CBR mendapatkan solusi dari kasus-kasus sebelumnya saja dan memperoleh kasus-kasus baru untuk meningkatkan dan mengembangkan kemampuan pengambilan keputusannya.

Lebih jauh, representasi suatu kasus memiliki berbagai bentuk, seperti contoh atau bahkan cerita, selama itu dapat dikenali oleh seorang pemikir dalam domain tertentu. Secara semantik, sebuah kasus mewakili sepotong pengetahuan tertentu dan konteksnya, di mana kasus tersebut akan diambil untuk membangun solusi untuk masalah baru. Ini berarti bahwa kita dapat melihat penalaran berbasis kasus sebagai proses mengingat serangkaian kasus sebelumnya dan membuat keputusan berdasarkan perbandingan antara mereka dan situasi baru.

Kolodner (1993) melanjutkan dengan membahas kelebihan dan kekurangan menggunakan CBR dalam pengembangan pengetahuan. Kolodner menegaskan bahwa CBR memungkinkan pemikir untuk mengajukan solusi atas masalah dengan cepat, menghindari waktu yang diperlukan untuk memperoleh jawaban-jawaban itu dari awal. Sebagai contoh, seorang dokter yang mengingat diagnosis lama atau perawatan mengalami manfaat ini. Pemikir berbasis kasus, seperti halnya pemikir lainnya, harus mengevaluasi solusi yang diusulkan; memulai upaya memecahkan masalah karena dapat menghasilkan proposal dengan mudah. Hal ini tentu terungkap selama evaluasi aplikasi CBR yang disebut CASEY (aplikasi CBR pertama Kolodner), yang menunjukkan percepatan dua urutan besarnya ketika masalah telah terlihat di masa lalu.

CBR memungkinkan pemikir untuk mengajukan solusi dalam domain yang tidak sepenuhnya dipahami oleh pemikir. Banyak domain yang mustahil untuk dipahami sepenuhnya, sering kali karena banyak yang bergantung pada perilaku manusia yang tidak terduga. CBR memungkinkan asumsi dan prediksi dibuat berdasarkan apa yang berhasil di masa lalu tanpa memiliki pemahaman yang lengkap tentang masalah atau masalah.

CBR memberi pemikir sarana untuk mengevaluasi solusi ketika tidak ada metode algoritmik yang tersedia untuk evaluasi. Menggunakan kasus, dalam hal ini, sangat membantu ketika ada banyak hal yang tidak diketahui, membuat jenis evaluasi lain menjadi tidak mungkin atau paling tidak sulit. Solusi dievaluasi dalam konteks situasi serupa sebelumnya. Sekali lagi, pemikir melakukan evaluasinya berdasarkan apa yang berhasil di masa lalu (Kolodner, 1993).

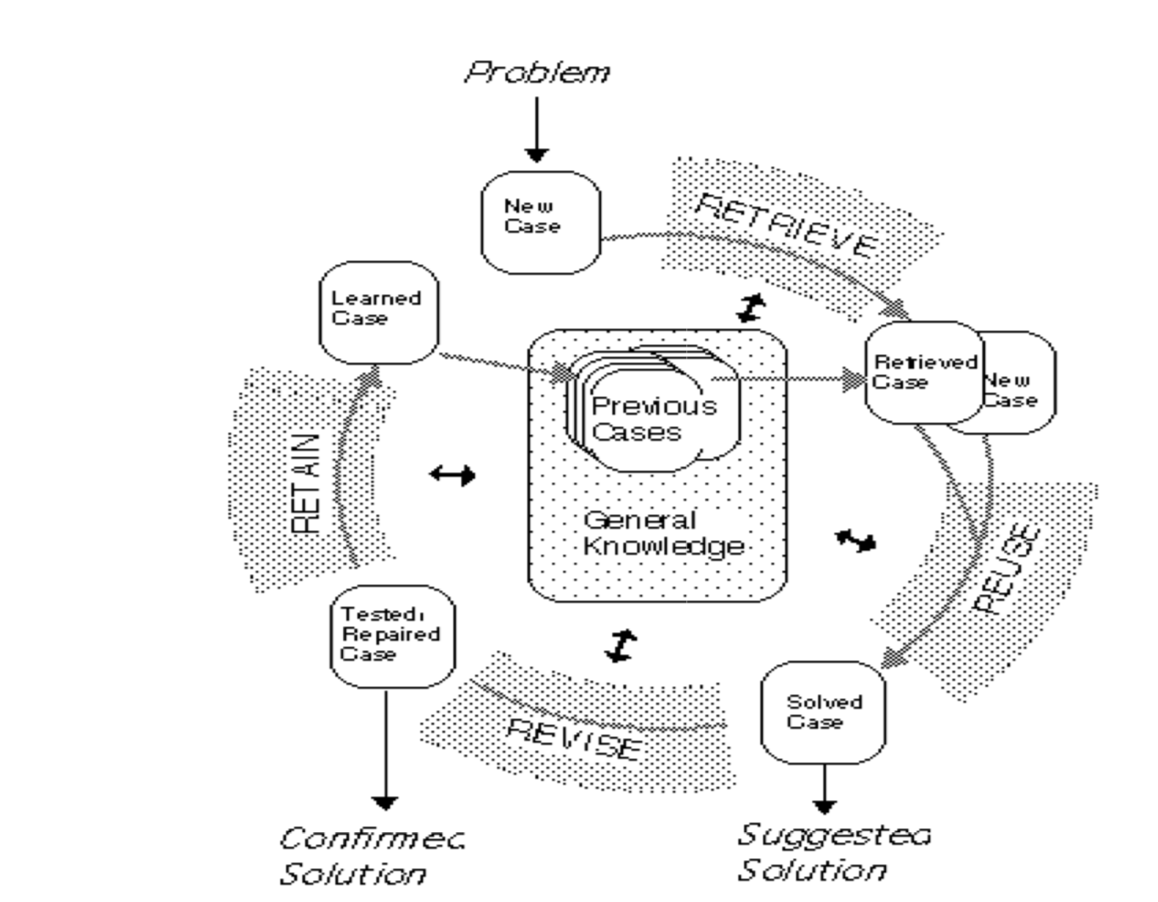
Metodologi penyelesaian masalah CBR sama dengan cara manusia menyelesaikan masalah ketika ada situasi ketika seseorang menghadapi situasi atau masalah baru, orang itu akan sering merujuk pada pengalaman masa lalu dari masalah yang sama.

(Pal dan Shiu, 2004). Pal dan Shiu melanjutkan bahwa konsep CBR sangat menarik karena fakta bahwa itu sangat mirip dengan perilaku pemecahan masalah manusia dan dengan demikian akan menghilangkan tugas analisis mendalam dari domain masalah di mana sejarah tersedia. Akhirnya, Pal dan Shiu menyimpulkan bahwa penggunaan metode ini mengarah pada keuntungan bahwa CBR dapat didasarkan pada "pengetahuan dangkal" dan tidak memerlukan upaya rekayasa pengetahuan yang diperlukan oleh sistem berbasis aturan.

Bergmann et al. (2003, p. 16) menyatakan bahwa "Dibandingkan dengan sistem pakar, sistem pendukung keputusan berbasis kasus tidak bergantung pada aturan yang dipasok oleh spesialis". Bergmann et al. percaya bahwa CBR lebih merupakan "pendekatan alami" di mana teknisi help-desk (atau spesialis lainnya) tidak pernah harus menyediakan aturan diagnostik atau untuk menentukan spesifikasi formal dari setiap proses keputusan yang digunakan untuk menentukan solusi untuk suatu masalah. Sistem pendukung keputusan CBR memiliki kemampuan untuk memperoleh dan mempertahankan pengetahuan karena sistem memiliki kemampuan untuk mempelajari kasus-kasus baru.

Pal dan Shiu (2004) menyatakan bahwa proses CBR dapat diabstraksi sebagai siklus yang terdiri dari empat langkah dasar (Gambar 1, Siklus CBR): (1) Pengambilan Kasus untuk menemukan kasus yang paling mirip yang akan mengatasi masalah baru , (2) Case Reuse untuk memanfaatkan case yang diambil untuk menyelesaikan masalah, (3) Case Revisi atau mengadaptasi untuk memodifikasi case yang diambil dengan harapan akan sesuai dengan masalah baru, dan (4) Retensi Kasus untuk mempertahankan kasus yang direvisi sebagai kasus baru dalam basis kasus setelah dikonfirmasi atau divalidasi.

Penelitian lain telah menyimpulkan bahwa empat langkah dasar dalam siklus CBR sebenarnya harus menjadi proses enam langkah dengan menambahkan fase Restore and Review.



Gambar Siklus CBR Cycle

Göker dan Roth-Berghofer (1999) percaya bahwa langkah-langkah dalam siklus CBR terkandung dalam dua siklus umum; Siklus Aplikasi dan Siklus Pemeliharaan. Siklus Aplikasi, yang berisi langkah Ambil, Penggunaan Kembali, dan Revisi, dilakukan setiap kali pengguna atau teknisi help-desk menyelesaikan masalah dengan sistem dukungan help-desk berbasis kasus. Jika solusi yang dihasilkan selama siklus Reuse tidak benar dan tidak dapat diperbaiki, teknisi help-desk harus menghasilkan solusi baru. Solusi baru ini digunakan selama fase ReCyle. Semua solusi baru yang dihasilkan dengan cara ini disimpan dalam buffer dan tersedia untuk teknisi help desk sebagai kasus yang belum dikonfirmasi. Kasing yang belum dikonfirmasi ini kemudian dikirim ke Siklus Pemeliharaan untuk diproses dan dimasukkan ke pangkalan kasus. Siklus Perawatan, yang berisi langkah Retain dan Perbaiki, jarang dilakukan. Biasanya,

fase pemeliharaan dilakukan pada interval tertentu untuk memperbarui basis kasus dengan case yang belum dikonfirmasi yang terkandung dalam buffer fase ReCycle. Ada dua langkah tambahan yang telah ditambahkan Göker dan Roth-Berghofer ke siklus CBR empat fase yang diterima secara umum. Ini adalah langkah-langkah ReCycle dan Perbaiki. Langkah ReCycle digunakan sebagai persimpangan antara Aplikasi dan Pemeliharaan Siklus dan berisi kasus yang tidak dikonfirmasi dikirim oleh Siklus Aplikasi. Kasing yang belum dikonfirmasi yang diambil dari buffer ReCycle ditempatkan pada langkah Perbaiki di mana kasus tersebut diperbaiki dan ditulis berbasis kasus.

Misi utama dari langkah Perbaiki adalah untuk memastikan bahwa kasus dasar akurat. Ada lima pemeriksaan yang dinyatakan oleh Göker dan Roth-Berghofer (1999, p. 144) yang harus dilakukan sebelum kasing dapat ditambahkan ke pangkalan kasing. Ini adalah:

1. “Apakah itu merupakan alternatif yang layak yang belum ada dalam basis kasus
2. Apakah itu termasuk atau dapat dimasukkan oleh kasus yang ada,
3. Apakah dapat dikombinasikan dengan kasus lain untuk membentuk yang baru,
4. Apakah kasus baru akan menyebabkan ketidakkonsistenan, dan
5. Apakah ada kasing yang lebih baru sudah tersedia di kasing. ”

Roth-Berghofer dan Iglezakis (2001) juga percaya bahwa siklus CBR enam langkah adalah metode yang benar dan bahwa dua fase, Pemeliharaan dan Aplikasi, paling baik menggambarkan proses CBR yang benar untuk mengambil solusi dan memastikan bahwa mereka akurat dan kemudian menyimpannya. dalam basis kasus. Gambar 4 menunjukkan bagaimana fase Perawatan dan Aplikasi berinteraksi.

1. **Daftar Pustaka**
2. https://core.ac.uk/download/pdf/51097894.pdf