Conversational Recommender System

Berbasis pada Kebutuhan Fungsional Produk

Dr ZK Abdurahman Baizal, MKom
Dr Nur Ulfa Maulidevi, MSc
Prof Dwi Hendratmo Widyantoro, MSc, PhD

Penerbit **INFORMATIKA**

Conversational Recommender System
Berbasis pada Kebutuhan Fungsional Produk

Penyusun	:	Dr ZK Abdurahman Baizal, MKom Dr Nur Ulfa Maulidevi, MSc Prof Dwi Hendratmo Widyantoro, MSc, PhD
Penerbit	:	Informatika Bandung
Pemasaran	:	**BI-Obses** Pasar Buku Palasari No. 82 Bandung 40264 Telp.(022)7317812 Fax. (022)7317896
Cetakan Pertama	:	Januari 2019
ISBN	:	xxx–xxx–xxxx–xx–x

Copyright **©** 2019 pada **Penerbit INFORMATIKA** Bandung

RINGKASAN

Menyatakan kebutuhan berdasarkan fitur teknis produk sering menyulitkan banyak calon pembeli, khususnya untuk produk multi fungsi dan mempunyai banyak fitur, seperti mobil, *notebook*, *smartphone*, *server*, kamera, dan sebagainya, dsb-dan sebagainya. Hal ini dikarenakan tidak semua orang familiar terhadap fitur teknis dari produk-produk tersebut. Menanyakan kebutuhan pengguna aspek kegunaan (kebutuhan fungsional) dari produk yang akan dibeli, adalah cara yang lebih natural dalam menggali kebutuhan pengguna. Oleh karena itu, buku ini menyajikan bagaimana membangun sebuah *conversational recommender system* (CRS) yang memperhatikan aspek kebutuhan fungsional produk.

Ontologi dipilih sebagai pengetahuan dari sistem, karena *nature* dari struktur ontologi, memungkinkan untuk membuat pemetaan yang lebih fleksibel antara kebutuhan fungsional produk, spesifikasi, dan produk. Selain itu, dalam ontologi, memungkinkan untuk penyusunan masing-masing konsep (entitas) secara hirarkis, dan struktur seperti ini sangat menguntungkan, terutama untuk mendukung pengembangan model pembangkitan pertanyaan. Struktur ontologi ini mempunyai 3 kelas utama, yaitu *FuncReq* (merepresentassikan kebutuhan fungsional), *Specification* (merepresentasikan gradasi kualitas fitur teknis) dan *Product* (merepresentasikan klasifikasi produk). Ontologi merupakan basis pengetahuan dari sistem. Mekanisme interaksi dilakukan melalui dialog tanya jawab, rekomendasi produk dan penjelasan mengapa suatu produk direkomendasikan, seperti layaknya interaksi antara calon pembeli dengan *professional sales support*. Model komputasional untuk membangkitkan interaksi dikembangkan dengan memanfaatkan eksplorasi relasi semantik dalam ontologi. Dengan model dan struktur ontologi ini, diharapkan pengembangan CRS yang disajikan dalam buku ini, dapat juga diterapkan untuk berbagai domain yang berbeda, khususnya untuk domain produk yang bersifat multi fungsi dan mempunyai banyak fitur (*notebook*, *server*, PC, mobil, kamera, *smartphone*, dan sebagainya, dsbdan sebagainya).

Evaluasi terhadap *CRS* yang dibangun meliputi evaluasi dari sisi efisiensi maupun efektifitas. Hasil evaluasi menunjukkan bahwa model interaksi dalam CRS berbasis kebutuhan fungsional mampu melakukan mekanisme *query requirement* dengan efisien, berdasarkan pengurangan jumlah sisa *record* secara signifikan dalam 4 interaksi. Dalam 4 interaksi, jumlah produk yang direkomendasikan kurang dari 20 dari 288 produk yang ada (< 0.6.9%). Dari sisi efektifitas, dilakukan *user study* yang melibatkan pengguna yang familiar (*expert user*) maupun tidak familiar (*novice user*) dengan fitur teknis produk. Hasil pengujian menunjukkan, CRS berbasis kebutuhan fungsional cukup efektif dalam memandu pengguna. Hal ini ditunjukkan dengan, baik *expert* maupun *novice user* lebih menyukai model interaksi CRS berbasis kebutuhan fungsional daripada model interaksi pada aplikasi pencarian produk berbasis pada fitur teknis produk (*expert user*: 86.67%, *novice user*: 90%). *User study* selanjutnya menunjukkan, interaksi dalam CRS berbasis kebutuhan fungsional mampu meningkatkan persepsi positif pengguna, dibandingkan dengan interaksi yang berbasis pada fitur teknis produk, dilihat dari *perceived ease of use*, *perceived enjoyment*, *trust* dan *perceived usefulness*. Selain itu, model interaksi juga efektif dalam mempengaruhi pengguna untuk tertarik mengadopsi sistem, namun terdapat perbedaan dalam faktor-faktor yang mempengaruhi hal tersebut. Untuk *expert user*, *perceived enjoyment* merupakan faktor yang mempengaruhi secara langsung untuk adopsi sistem, sedangkan *perceived usefulness* merupakan faktor yang secara langsung mempengaruhi adopsi sistem, bagi *novice user*.

Kata Kunci —*conversational recommender system*, kebutuhan fungsional produk, *query refinement*, ontologi

DAFTAR ISI

RINGKASAN ..... **iii**
DAFTAR GAMBAR DAN ILUSTRASI ..... **vii**
DAFTAR TABEL ..... **ix**
DAFTAR ISTILAH ..... **xi**
DAFTAR SINGKATAN DAN LAMBANG ..... **xiii**

Bab 1. Pendahuluan **1**
1.1 Latar Belakang 1
1.2 *Overview* dari CRS Berbasis pada Kebutuhan Fungsional Produk 4

Bab 2. Penelitian-Penelitian Terkait *Conversational Recommender System* **9**
2.1 Pendahuluan 9
2.2 *Recommender System* 10
2.3 *Conversational Recommender System* 14
2.4 *Constraint Satisfaction Problem* 16
2.5 *Query Refinement* dalam CRS 19
2.6 *Recommender System* Berbasis pada Ontologi 20
2.7 *Resume* 23

Bab 3. Formalisasi Ontologi **27**
3.1 Pendahuluan 27
3.2 Struktur Ontologi 27
3.2.1 Relasi 29
3.2.2 Hirarki *class Specification* 32
3.2.3 Hirarki FuncReq 33
3.2.4 Hirarki *Class Product* 34
3.3 Definisi Formal dari Ontologi 37
3.4 Pembangunan instance dari Ontologi 38
3.5 *Resume* 41

Bab 4. Pemodelan Kebutuhan Pengguna **43**
4.1 Pendahuluan 43
4.2 *User Profile* Model 44

4.3 Mekanisme Interaksi 45
4.4 Perumusan Kasus-kasus *Feedback* (Kasus Interaksi) Pengguna 48
4.5 *Resume* 50

Bab 5. Model Komputasional untuk Rekomendasi Produk ... 51
5.1 Pendahuluan 51
5.2 Fungsi-fungsi Dasar dan Algoritma Rekomendasi Produk 52
5.3 Kajian Kompeksitas dari Model Rekomendasi. 62
5.4 *Resume* 65

Bab 6. Model Komputasional untuk Pembangkitan Pertanyaan dan Penjelasan 67
6.1 Pendahuluan 67
6.2 Model untuk Pembangkitan Pertanyaan 69
6.2.1 Kasus empty_*profile* – *User* Profile Model Masih Kosong 70
6.2.2 Kasus *multi_chos_prod* – Terdapat Lebih dari Satu Produk yang Dipilih Pengguna..... 71
6.2.3 Kasus *less_specific* – Kebutuhan Pengguna Masih Belum Cukup untuk Rekomendasi Produk 75
6.2.4 Kasus *no_chos_prod* – Tidak Ada Produk yang Dipilih Pengguna. 81
6.2.5 Kasus *contra_pref*– Tidak Ada Produk yang Sesuai dengan User Profile Model 85
6.3 Model untuk Pembangkitan Penjelasan 87
6.4 *Resume* 89

Bab 7. Evaluasi..... 91
7.1 Pendahuluan 91
7.2 Efisiensi dari Model Interaksi 92
7.3 Efektifitas Model Interaksi dalam Memandu Pengguna 94
7.4 Efektifitas Model Interaksi dalam Mempengaruhi Persepsi Pengguna 99
7.5 *Resume* 111

LAMPIRAN 113
DAFTAR PUSTAKA 121

DAFTAR GAMBAR DAN ILUSTRASI

Gambar 1.1	Komponen-komponen CRS yang dibangun.....	5
Gambar 1.2	Proses pembangkitan interaksi.....	6
Gambar 2.1	Keterkaitan beberapa teknik rekomendasi dengan *recommender system* maupun *conversational recommender system*.....	13
Gambar 2.2	Contoh percakapan untuk query refinement dalam expertclerk (Shimazu, 2002)	20
Gambar 2.3	Contoh potongan ontologi program TV dengan Struktur .multi hirarki (Blanco-Fernadez dkk., 2008a; 2008b).....	21
Gambar 2.4	Contoh hubungan antar *instance* (individu) dalam ontologi (Blanco-Fernadez dkk., 2008a; 2008b)	22
Gambar 2.5	Ruang lingkup dari CRS yang dikembangkan	24
Gambar 3.1	Struktur ontologi	28
Gambar 3.2	Contoh hirarki *Specification* dalam domain smartphone..	32
Gambar 3.3	Contoh hirarki *FuncReq* dalam domain smartphone. ..	33
Gambar 3.4	Contoh struktur hirarki *Product*	34
Gambar 3.5	Contoh potongan ontologi	36
Gambar 3.6	Sumber data untuk pembangunan ontologi	39
Gambar 4.1	*User profile* model dan fungsi correspond	45
Gambar 4.2	Skema interaksi antara pengguna dan sistem	47
Gambar 5.1	Contoh potongan ontologi untuk kasus rekomendasi...	54
Gambar 5.2	Algoritma untuk merekomendasikan produ.....	57
Gambar 5.3	Ilustrasi langkah-langkah pada algoritma *recommend*58	
Gambar 5.4	Contoh potongan ontologi untuk hirarki FuncReq	60
Gambar 5.5	*Graf bipartite* antara Ifunc dan Ispec	63
Gambar 5.6	Graf bipartite antara Ispec dan Iprod	63
Gambar 6.1	Algoritma untuk membangkitkan pertanyaan saat *User profile* model kosong.....	71
Gambar 6.2	Ilustrasi untuk *distinctive functional requirements* dari product p1.....	72
Gambar 6.3	Algoritma untuk membangkitkan saat pengguna memilih lebih dari satu produk	72

Gambar 6.4 Contoh potongan ontologi untuk kasus *multi_chos_prod* 73
Gambar 6.5 Algoritma untuk membangkitkan pertanyaan saat user profile model belum cukup untuk rekomendasi.... 75
Gambar 6.6 Contoh kasus *less_specific* dengan *candidate* $\neq \phi$ 76
Gambar 6.7 Contoh kasus *less_specific* dengan *candidate* = ϕ 78
Gambar 6.8 Algoritma untuk membangkitkan pertanyaan saat pengguna tidak memilih satupun produk yang direkomendasikan 81
Gambar 6.9 Algoritma untuk mencari unexplored *candidate nodes* 82
Gambar 6.10 Contoh untuk kasus *no_chos_prod* 84
Gambar 6.11 Algoritma untuk membangkitkan pertanyaan saat tidak ada produk yang sesuai user *profile model* 86
Gambar 6.12 *Template* dari fasilitas penjelasan 87
Gambar 6.13 Contoh penjelasan 89
Gambar 7.1 Gambar rata-rata jumlah sisa *record* dari masing-masing tahap *refinement* 93
Gambar 7.2 Preferensi pengguna terhadap *func-based model* vs *tech-based model* untuk keseluruhan interaksi..... 97
Gambar 7.3 Preferensi *expert user* terhadap *func-based model* vs *tech-based model* 98
Gambar 7.4 Preferensi *Novice User* Terhadap Func-based Model vs Tech-based Model 99
Gambar 7.5 Model hipotesis..... 103
Gambar 7.6 Paths dari Hipotesis yang diterima untuk Keseluruhan pengguna..... 108
Gambar 7.7 Paths dari hipotesis yang diterima untuk *expert users*. 110
Gambar 7.8 Paths dari hipotesis yang diterima untuk *novice users*. 110

DAFTAR TABEL

Tabel 2.1	Daftar fitur produk kamera (Felfernig dkk., 2009)	17
Tabel 2.2	Variabel dan *constraints* untuk *digital camera* (Jannach, 2004).....	17
Tabel 3.1	Contoh penentuan gradasi kualitas untuk masing-masing jenis spesifikasi.....	40
Tabel 3.2	Contoh pemetaan antara kebutuhan fungsional dan spesifikasi	41
Tabel 7.1	Perbedaan strategi dari kedua model interaksi.....	95
Tabel 7.2	Constructs dan Item-itemnya.....	101
Tabel 7.3	*Item-Item construct* yang ter-ekstrak.....	106
Tabel 7.4	Rata-rata ipengguna dan hasil T-Test.....	107

DAFTAR ISTILAH

Istilah	Nama	Pemakaian pertama kali pada halaman
Conversational recommender system	suatu *recommender system* yang mengadopsi mekanisme interaksi secara berulang antara pengguna dengan sistem untuk mendapatkan produk yang diinginkan oleh pengguna	1
Candidat nodes	Himpunan *node* dalam hirarki *FuncReq* yang potensial untuk ditanyakan (untuk selanjutnya, disebut sebagai *candidat nodes*)	69
Kebutuhan fungsional produk (*product functional requirements*)	Kebutuhan dari produk yang diinginkan, dilihat dari sisi fungsionalitasnya/peruntukannya, misal seseorang perlu *smartphone* untuk keperluan *browsing* dan tidak akan dipakai untuk bermain *game*, tetapi masih nyaman dimasukkan saku	2
Mandatory functional requirements	Kebutuhan pengguna yang wajib dipenuhi, atau mutlak harus dipenuhi	44
Navigation by asking	Suatu strategi navigasi, dengan sistem secara berulang memberikan pertanyaan kepada pengguna dan diakhiri dengan rekomendasi produk berdasarkan kebutuhan yang didapatkan dari jawaban pengguna (Bridge dkk., 2006).	1
Navigation by proposing	Suatu strategi navigasi, dengan sistem menunjukkan produk-produk tertentu kepada pengguna, dan	1

Istilah	Nama	Pemakaian pertama kali pada halaman
	mendapatkan kebutuhan pengguna dalam bentuk *feedback* terhadap produk yang direkomendasikan. Interaksi berakhir jika penguna memilih salah satu produk yang direkomendasikan	
Not required functional requirement	Jenis kebutuhan yang tidak dibutuhkan oleh pengguna	44
Ontologi	Sebuah konseptualisasi domain pengetahuan, ke dalam format yang dapat dipahami oleh manusia, dan juga dapat dibaca oleh mesin, yang terdiri dari entitas-entitas, relasi-relasi dan aksioma (Lee, 2009; Guarino dan Giaretta 1995)	2
Optional functional requirements	Kebutuhan pengguna yang tidak harus dipenuhi, namun kalau dipenuhi akan lebih baik	44
Professional sales support	Seorang *sales* yang mempunyai pengetahuan mendalam tentang produk, dan mampu memandu pengguna untuk mengungkapkan kebutuhan dan mengambil kesimpulan (Shimazu, 2002)	2
Query refinement	Proses penggalian kebutuhan penguna, dengan cara menajamkan, meruncingkan, mendetilkan kebutuhan pengguna melalui suatu mekanisme khusus (Hu dan Aufaure, 2013)	3

DAFTAR SINGKATAN DAN LAMBANG

SINGKATAN	Nama	Pemakaian pertama kali pada halaman
BI	*Behavioral Intention*	100
GFI	*Goodness of Fit Index*	103
CP	*contra_pref*	49
CRS	*Conversational Recommender System*	1
EMP	*empty_profile*	48
EOU	*Perceived Ease of Use*	100
LS	*less_specific*	49
MCP	*multi_chos_prod*	71
NBA	*Navigation by Asking*	1
NBP	*Navigation by Proposing*	1
NCP	*no_chos_prod*	95
PE	*Perceived Enjoyment*	102
PU	*Perceived Usefulness*	100
RMSEA	*Root Mean Square Error of Approximate*	103
TAM	*Technology Acceptance Model*	91
TR	*Trust*	101

LAMBANG

C	Himpunan *class* dalam ontologi	31
C_{func}	Himpunan *class* dalam hirarki *FuncReq*,	37
C_{prod}	Himpunan *class* dalam hirarki *Product*	37
C_{spec}	Himpunan *class* dalam hirarki *Specification*	32

F_i	Himpunan kebutuhan fungsional dalam *user profile model* pada level *i*	45
Fm	Himpunan *mandatory functional requirements*	44
Fo	Himpunan *optional functional requirements*	44
Fx	Himpunan *not required functional requirements*	44
H_i	Hipotesis ke - *i*	103
I	Himpunan individu	32
I_{func}	Himpunan individu dalam hirarki *FuncReq*	38
I_{prod}	Himpunan individu dalam hirarki *Product*	52
I_{spec}	Himpunan individu dalam hirarki *Specification*	38
N	Himpunan *node* dalam ontologi	28
N_D	Himpunan batasan untuk N	37
N_{func}	Himpunan *node* dalam hirarki *FuncReq*	37
N_{prod}	Himpunan *node* dalam hirarki *Product*	37
N_{spec},	Himpunan *node* dalam hirarki *Specification*	37
OP	Himpunan fungsi yang berlaku dalam ontologi	37
Pproperty	Himpunan properti produk yang dipilih	44
PType	Himpunan jenis produk yang diinginkan	44
R	Sebuah relasi dalam ontologi	28
Rel	Himpunan relasi dalam ontologi	29
R_H	Himpunan relasi antar *node* dalam suatu hirarki	29
R_O	Himpunan relasi antar *node* dari hirarki-hirarki yang berbeda	30
α	Maksimum jumlah pertanyaan tiap tahap interaksi	77

Istilah	Nama	Pemakaian pertama kali pada halaman
Conversational recommender system	suatu *recommender system* yang mengadopsi mekanisme interaksi secara berulang antara pengguna dengan sistem untuk mendapatkan produk yang diinginkan oleh pengguna	1
Candidat nodes	Himpunan *node* dalam hirarki *FuncReq* yang potensial untuk ditanyakan (untuk selanjutnya, disebut sebagai *candidat nodes*)	78
Kebutuhan fungsional produk (*product functional requirements*)	Kebutuhan dari produk yang diinginkan, dilihat dari sisi fungsionalitasnya/peruntukannya, misal seseorang perlu *smartphone* untuk keperluan *browsing* dan tidak akan dipakai untuk bermain *game*, tetapi masih nyaman dimasukkan saku	2
Mandatory functional requirements	Kebutuhan pengguna yang wajib dipenuhi, atau mutlak harus dipenuhi	79
Navigation by asking	Suatu strategi navigasi, dengan sistem secara berulang memberikan pertanyaan kepada pengguna dan diakhiri dengan rekomendasi produk berdasarkan kebutuhan yang didapatkan dari jawaban pengguna (Bridge dkk., 2006).	1
Navigation by proposing	Suatu strategi navigasi, dengan sistem menunjukkan produk-produk tertentu kepada pengguna, dan mendapatkan kebutuhan pengguna	1

	dalam bentuk *feedback* terhadap produk yang direkomendasikan. Interaksi berakhir jika penguna memilih salah satu produk yang direkomendasikan	
Not required functional requirement	Jenis kebutuhan yang tidak dibutuhkan oleh pengguna	44
Ontologi	Sebuah konseptualisasi domain pengetahuan, ke dalam format yang dapat dipahami oleh manusia, dan juga dapat dibaca oleh mesin, yang terdiri dari entitas-entitas, relasi-relasi dan aksioma (Lee, 2009; Guarino dan Giaretta 1995)	3
Optional functional requirements	Kebutuhan pengguna yang tidak harus dipenuhi, namun kalau dipenuhi akan lebih baik	44
Professional sales support	Seorang *sales* yang mempunyai pengetahuan mendalam tentang produk, dan mampu memandu pengguna untuk mengungkapkan kebutuhan dan mengambil kesimpulan (Shimazu, 2002)	2
Query refinement	Proses penggalian kebutuhan penguna, dengan cara menajamkan, meruncingkan, mendetilkan kebutuhan pengguna melalui suatu mekanisme khusus (Hu dan Aufaure, 2013)	3

1

PENDAHULUAN

1.1 Latar Belakang

Pada saat seorang calon pembeli akan membeli suatu produk, dia sering kesulitan untuk memutuskan produk yang akan dibeli, karena banyaknya produk yang ada. Untuk itu berkembang suatu aplikasi yang disebut *recommender system*, yang dapat membantu pengguna untuk mendapatkan produk yang sesuai dengan kebutuhan dan keinginannya. Dalam kehidupan nyata, suatu proses rekomendasi jarang terjadi dalam sesi yang singkat, karena pengguna jarang dapat menyatakan kebutuhannya secara utuh di awal sesi interaksi, sehingga banyak dikembangkan *conversational recommender system* (CRS), suatu *recommender system* yang mengadopsi mekanisme percakapan antara pengguna dengan sistem untuk mendapatkan produk yang diinginkan oleh pengguna.

Terdapat dua macam strategi untuk membangun interaksi dalam CRS, yaitu *navigation by asking* (NBA) dan *navigation by proposing* (NBA) (Bridge dkk., 2006). Dalam NBA, sistem secara berulang memberikan pertanyaan kepada pengguna dan diakhiri dengan rekomendasi produk berdasarkan kebutuhan yang didapatkan dari jawaban pengguna. Terdapat beberapa metode yang digunakan untuk menghasilkan urutan pertanyaan yang

efisien, seperti konsep *feature selection* dengan menggunakan *information gain* (Smyth dan Cunningham, 1994; Doyle dan Cunningham, 2000), *similarity of variance* dan *utility function* (Kohlmaier dkk., 2001; Schmitt, 2002; Schmitt dkk., 2002), *popularity of features* (Mirzadeh dkk., 2005) dan *weighted majority voted* (Felfernig dan Burke, 2008). Metode-metode ini mampu menghasilkan pertanyaan berbasis pada fitur teknis produk dengan efisien. Sementara itu, pendekatan *rule-based* menunjukkan performa yang bagus untuk membangkitkan interaksi berbasis pada kebutuhan fungsional produk (Jannach, 2004; Felfernig dkk., 2006; Jannach dan Kreutler, 2004; 2007).

Dalam NBP, sistem menunjukkan produk-produk tertentu kepada pengguna dan mendapatkan kebutuhan pengguna dalam bentuk *feedback* terhadap produk yang direkomendasikan. Untuk menghasilkan interaksi yang efisien, beberapa peneliti memodifikasi bentuk *feedback* pengguna dalam bentuk *compound critiques* (Smyth dkk., 2004; McCarthy dkk., 2004; Reilly dkk., 2005). Selain itu juga dikembangkan metode-metode lain seperti *preference-based feedback* (McGinty and Smyth, 2002), *incremental critiquing* (Llorente dan Guerrero, 2012) dan *experience-based unit critiquing* (Mandl dan Felfernig, 2012).

Dalam perkembangan selanjutnya, dikembangkan CRS yang dapat berperan layaknya seorang *professional sales support* dengan cara memadukan NBA dan NBP. Menurut Shimazu (2001; 2002), seorang *professional sales support* mempunyai 3 peran. Pertama, dia sebagai seorang *problem solver*, dalam arti, dia melakukan tanya jawab dengan calon pembeli untuk mendapatkan produk yang memenuhi kebutuhan pengguna. Kedua, dia berperan sebagai seorang *adviser*, dalam arti jika calon pembeli belum dapat mengungkapkan kebutuhannya secara pasti, dia akan mengajukan beberapa contoh produk untuk mendapatkan kebutuhan pengguna. Ketiga, seorang *profesional sales support* harus mampu memberikan *testimony* (penjelasan) kepada calon pembeli, tentang produk yang direkomendasikan, untuk membantu pengguna mengambil keputusan. Ketiga peran ini didapatkan dengan memadukan NBA dan NBP dalam sebuah CRS. Pengembangan CRS yang disajikan dalam buku ini, terinspirasi dengan ide kombinasi NBA dan NBP. Kedua paper ini memanfaatkan konsep *information gain* dan pendekatan statistik untuk membangkitkan interaksi yang memadukan NBA dan NBP.

Hu dan Aufaure (2013) juga memanfaatkan kombinasi NBA dan NBP untuk *melakukan query refinement* melalui percakapan. Metode yang dikembangkan dalam penelitian ini mampu mengakomodasi kebutuhan yang terdiri dari beberapa atribut, termasuk *contextual information*, dengan menggunakan konsep *Multi Attribute Utility Theory* (MAUT). Paper ini juga mengusulkan sebuah *framework* yang terdiri dari model untuk *query refinement* serta arsitektur ontologi untuk merepresentasikan *user preference model*. Pengembangan CRS dalam buku ini, terinspirasi dengan keberadaan *user preference model* yang di-ekstrak dari ontologi basis pengetahuan, untuk merangkum kebutuhan pengguna serta membangkitkan interaksi. Namun, dalam penelitian-penelitian ini, baik pertanyaan, model untuk rekomendasi produk serta pemberian penjelasan mengacu pada fitur produk. Interaksi yang mengacu pada fitur produk secara langsung memang efektif untuk produk yang mempunyai fitur sedikit, seperti wine, pakaian (Shimazu, 2001; 2002), restoran (Hu dan Aufaure 2013),dan sebagainya. Namun, untuk produk yang mempunyai banyak fitur, dan bersifat multi fungsi (misal mobil, *smartphone*, *notebook*, kamera, *server*,dan sebagainya), tentu tidak semua pengguna familiar dengan fitur-fitur teknis produk, sehingga pertanyaan kebutuhan fungsional produk lebih natural untuk mendapatkan kebutuhan pengguna.

Kebutuhan fungsional produk merupakan kebutuhan/peruntukan dari produk yang diinginkan, dilihat dari sisi fungsionalitasnya, misal seseorang perlu *smartphone* untuk keperluan *browsing* dan tidak akan dipakai untuk bermain *game*, tetapi masih nyaman dimasukkan saku. Seorang *sales support* yang baik, juga harus mempunyai kemampuan untuk menggali kebutuhan pengguna dari sisi fungsional dan merekomendasikan produk yang sesuai dengan kebutuhan pengguna. Dengan demikian, buku ini menyajikan bagaimana membangun sebuah CRS yang berbasis pada kebutuhan fungsional produk, dengan memadukan strategi NBA dan NBP, yang diharapkan dapat menirukan percakapan antara seorang calon pembeli dengan *professional sales support* dalam pencarian produk.

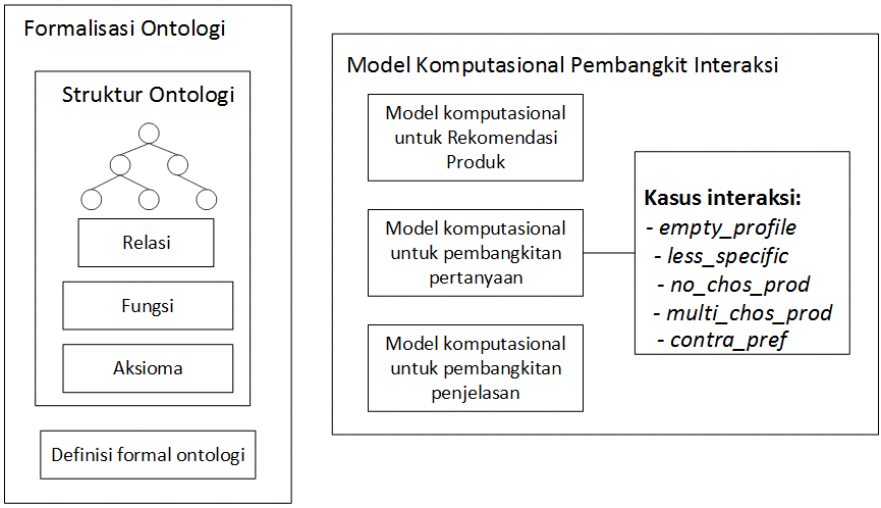
CRS dibangun dengan memanfaatkan konsep *knowledge based recommender system* yang berbasis pada kebutuhan fungsional. Untuk membangkitkan interaksi berbasis pada kebutuhan fungsional, harus dikembangkan sebuah model basis pengetahuan yang mampu memetakan kebutuhan fungsional dengan fitur teknis produk. Ontologi merupakan suatu representasi pengetahuan yang dipandang cukup fleksibel dalam memetakan *high level requirements* (kebutuhan fungsional) dengan konsep-

konsep yang mendukungnya. Pendekatan *rule-based* (Jannach, 2004; Felfernig dkk., 2006; Jannach dan Kreutler, 2004; 2007) maupun perpaduan ontologi dan *rule-based* (Matobuwana dkk, 2009; Kato dkk, 2010; Chen dkk., 2012; Vesin dkk., 2012) dipandang kurang fleksibel untuk membangun sebuah CRS yang dapat diimplementasikan untuk berbagai domain produk. Penelitian ini memanfaatkan ontologi dengan struktur multi-hirarki (Hu dan Aufaure 2013; Blanco-Fernandez dkk., 2008a; 2008b; Gu dan Aamodt, 2008). Struktur multi-hirarki ini memungkinkan untuk memuat informasi yang lebih kaya dalam pembangkitan interaksi berbasis pada kebutuhan fungsional. Model pembangkitan interaksi dikembangkan berdasarkan eksplorasi relasi semantik dalam ontologi.

Buku ini menyajikan formalisasi ontologi (struktur ontologi, relasi-relasi, fungsi-fungsi dasar, aksioma dan definisi formal ontologi) dan model komputasional untuk pembangkitan interaksi dalam CRS berbasis kebutuhan fungsional produk. Interaksi terdiri dari pemberian pertanyaan, rekomendasi produk dan penyajian fasilitas penjelasan mengapa suatu produk direkomendasikan (kombinasi NBA dan NBP). Dengan demikian, CRS yang dibangun diharapkan dapat mempunyai sebuah mekanisme interaksi yang efisien, mampu memandu pengguna untuk mengungkapkan kebutuhannya dengan lebih mudah, serta mampu meningkatkan persepsi positif pengguna, sehingga pengguna tertarik untuk mengadopsi sistem yang dihasilkan.

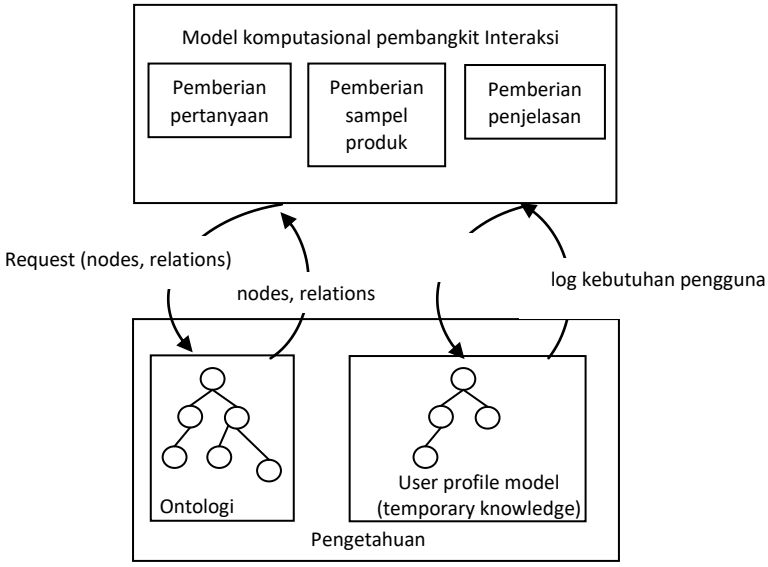
1.2 *Overview* dari CRS Berbasis pada Kebutuhan Fungsional Produk

Buku ini menyajikan suatu cara pengembangan CRS yang berbasis pada kebutuhan fungsional. Untuk dapat berperan layaknya seorang *professional sales support*, sistem harus dapat menciptakan suatu interaksi yang dapat memandu pengguna untuk mengungkapkan kebutuhan serta mengambil keputusan. Dalam memandu pengguna, sistem harus mampu memberikan pertanyaan-pertanyaan, merekomendasikan sampel produk dan menyajikan penjelasan mengapa produk-produk tertentu direkomendasikan. Interaksi ini merupakan representasi dari kombinasi strategi NBA dan NBP.



Gambar 1.1 Komponen-komponen CRS yang dibangun

CRS terdiri dari ontologi dan model komputasional pembangkit interaksi. Gambar I.1 menunjukkan komponen-komponen dalam CRS yang dikembangkan. Struktur ontologi direpresentasikan dengan definisi relasi-relasi (termasuk nodes dalam ontologi), fungsi dan aksioma-aksima yang berlaku dalam ontologi, serta definisi formal ontologi. Definisi ontologi menggunakan pendekatan definisi bahasa formal (Ibanez dkk., 2009; Nguyen, 2008; Aubrech dkk., 2005), dengan menggunakan pendekatan definisi bahasa formal. Ontologi ini bersifat *generic*, sehingga dapat merupakan pedoman dalam penerapan ontologi dalam berbagai domain, khususnya produk yang bersifat multi fungsi dan banyak fitur (notebook, mobil, smartphone, kamera, server, dan sebagainya). Model komputasional pembangkit interaksi merupakan model untuk melakukan *reasoning* dengan tujuan membangkitkan interaksi yang berupa pembangkitan pertanyaan, rekomendasi dan penjelasan mengapa produk direkomendasikan. Dalam membangkitkan pertanyaan, terdapat lima kasus interaksi, yang secara detil akan dibahas dalam Bab VI.



Gambar 1.2 Proses pembangkitan interaksi

Komponen-komponen pada ontologi dan model komputasional ini mendukung terciptanya *personalized dialogue/interaction* antara pengguna dan sistem. Dengan demikian, sistem harus mampu belajar tentang pengetahuan pengguna selama interaksi berjalan, dan memodelkan kebutuhan pengguna dalam *user profile model*. Gambar I.2 menunjukkan proses pembangkitan interaksi berdasarkan ontologi maupun *user profile model*. *User profile model* merepresentasikan *log* kebutuhan pengguna selama interaksi. Informasi ini menjadi dasar dalam pemberian pertanyaan selanjutnya, pemberian sampel produk, maupun dalam memberikan penjelasan. *User profile model* merupakan *subset* dari ontologi, yang dimutakhirkan secara dinamis oleh model pembangkit interaksi selama interaksi berjalan.

Dalam konteks *reasoning*, *user profile model* merupakan fakta, dan ontologi merupakan pengetahuan (*rules* direpresentasikan secara implisit dengan relasi-relasi semantik dan juga struktur ontologi). Untuk pembangkitan interaksi, baik itu rekomendasi produk, pembangkitan pertanyaan maupun fasilitas penjelasan, *reasoning* dilakukan dengan memanfaatkan *semantic reasoning* (Blanco Fernandez dkk., 2008a;2008b). *Reasoning* ini dilakukan dengan menelusuri relasi-relasi dan *nodes* dalam sebuah hirarki maupun

antar hirarki. Rekomendasi produk dapat dipandang sebagai sebuah *constraint satisfaction problem* (CSP), dengan *user profile model* merupakan sekumpulan *constraint* yang termutakhirkan secara dinamis sepanjang interaksi. Dalam konteks *knowledge-based system*, model komputasional pembangkit interaksi yang disajikan dalam buku ini, berperan dalam memicu proses *reasoning* ini (*problem solving component*), baik dalam merekomendasikan produk, membangkitkan pertanyaan maupun penjelasan.

Pada saat membangkitkan pertanyaan, sistem menelusuri *log* kebutuhan dari pengguna pada *user profile model*, dan mencari kebutuhan fungsional yang potensial disukai pengguna, untuk ditanyakan. Model pembangkit pertanyaan bertugas melakukan penelusuran pada *user profile model* (*case specific facts*) dan kemudian dicocokkan dengan pengetahuan (dalam hal ini adalah ontologi). Aksioma-aksioma terkait pembangkit pertanyaan berfungsi sebagai acuan validasi dari model yang dikembangkan.

Proses *reasoning* pada ontologi merupakan *semantic reasoning* (Blanco-Fernandez dkk., 2008a; 2008b). Proses *reasoning* melibatkan penelusuran relasi-relasi semantik antar *node* dalam ontologi. Penelusuran bisa pada relasi-relasi dalam hirarki maupun relasi-relasi antar hirarki. Seperti halnya *semantic reasoning* pada Blanco Fernandez dkk. (2008a; 2008b), penelusuran relasi didukung oleh beberapa fungsi-fungsi yang telah didefinisikan.

2

PENELITIAN-PENELITIAN TERKAIT *CONVERSATIONAL RECOMMENDER SYSTEM*

2.1 Pendahuluan

Bab ini dibagi dalam beberapa topik bahasan, khususnya yang terkait dengan *Conversational Recommender System*. Pertama akan dibahas tentang berbagai penelitian yang telah dikembangkan dalam *recommender system* beserta teknik-teknik populer yang ada beserta keunggulan dan kelemahannya. Selanjutnya dibahas tentang penelitian-penelitian yang telah dikembangkan dalam area *conversational recommender system*, termasuk teknik pembentukan model pengguna dalam *conversational recommender system*, yang dibagi dalam 2 macam strategi, *navigation by asking* (NBA) dan *navigation by proposing* (NBP). Selain itu dibahas juga tentang

beberapa penelitian dalam bidang *recommender syste*m yang memanfaatkan ontologi sebagai basis pengetahuan.

2.2 Recommender System

Recommender system merupakan sistem cerdas yang banyak diaplikasikan dalam *e-commerce,* yang bertujuan untuk mengatasi masalah *information overload*, dengan merekomendasikan konten, produk atau layanan (misal buku, produk, film, musik, program TV, komputer, smartphone, dan sebagainya) kepada pengguna yang paling sesuai dengan kebutuhan dan keinginannya, sesuai dengan situasi dan konteks permasalahan (Resnick & Varian, 1997). Sebagian besar penelitian dalam bidang *recommender system*, terkait dengan bidang film, diikuti oleh bidang *shopping*, kemudian diikuti dengan bidang-bidang lain, seperti bidang *image*, *music*, *TV program*, *travel product*, dan sebagainya. Sampai saat ini, riset dalam bidang *recommender system* masih kurang "*mature*" dibanding riset dalam bidang-bidang yang lain, sehingga masih banyak peluang riset dalam bidang ini (Park dkk., 2012).

Banyak teknik rekomendasi yang telah dikembangkan oleh para peneliti dalam *recommender system*, yang dapat diklasifikasikan dalam 3 kategori (Symeonidis dkk., 2008; Ricci dkk., 2009) : *collaborative filtering*, *content based* dan *hybrid*. Teknik *Collaborative filtering* membangkitkan rekomendasi menggunakan informasi dari rating yang diberikan oleh pengguna terhadap produk. Dalam teknik ini, sistem merekomendasikan produk-produk yang mempunyai rating tinggi dari pengguna lain yang mempunyai selera yang sama. Pendekatan *collaborative recommendation* ini terinspirasi oleh kebiasaaan sehari-hari. Sebagai contoh, dalam mencari informasi atau menentukan pilihan, orang sering berkonsultasi dengan teman-teman yang mempunyai kesukaan atau selera yang sama (Ha, 2002). Namun teknik ini mempunyai dua kelemahan utama: *sparsity problem* dan *scalability problem* (Adomavicius dan Tuzhilin, 2005).

Teknik *content-based filtering* memberikan rekomendasi dengan menggunakan deskripsi pada fitur dari produk. Dalam teknik ini, seorang pengguna dimodelkan dengan sebuah profil yang merepresentasikan kebutuhan pengguna untuk fitur-fitur produk tertentu. Sistem merekomendasikan produk-produk yang mempunyai fitur yang mempunyai kecocokan tinggi dengan profil pengguna (Ricci dkk., 2009). *Content-based*

recommendation ini lebih mirip *task* klasifikasi dalam *machine learning*. Sistem melakukan pembelajaran berdasarkan pada informasi rating pengguna dan atribut-atribut pada masing-masing produk sedemikian hingga produk-produk dapat diklasifikasikan menjadi "menarik" atau "tidak menarik". Tidak ada *social knowledge* yang digunakan. Tetapi, secara umum, beberapa teknik yang telah dikembangkan masih kurang berhasil, dikarenakan masalah *sparsity*. Dalam banyak kasus, dibutuhkan data yang cukup banyak untuk membangun sebuah profil yang bisa diandalkan. *K-NN* dan Naive Bayes terbukti cukup efektif untuk menangani hal ini (Felfernig dan Burke, 2008). Namun karakteristik dari teknik ini yang mendeteksi kesamaan antar produk berdasarkan kesamaan nilai atribut, menyebabkan *overspesialized recommendations*, dalam arti hanya memasukkan produk yang sangat mirip (Ricci dkk., 2003).

Karena kelemahan yang ada pada pendekatan *collaborative* maupun *content based*, beberapa *Recommender System* menggunakan sebuah pendekatan *hybrid recommendation* yang mengkombinasikan dua (atau lebih) teknik rekomendasi untuk mendapatkan kelebihan dari masing-masing teknik, atau untuk mengatasi kelemahan-kelemahan dari masing-masing teknik (Adomavicius dan Tuzhilin, 2005; Hsief dkk., 2004; Kazienko dan Kołodziejski, 2008; Ullah dkk., 2012).

Knowledge-based recommendation menggunakan *domain knowledge* untuk mencari hubungan (kesesuaian) antara kebutuhan pengguna dengan produk tertentu. Dalam *knowledge-based recommender system*, sistem mendapatkan kebutuhan pengguna, kemudian menentukan produk yang paling sesuai berdasarkan basis pengetahuan (Felfernig dan Burke, 2008). Terdapat dua pendekatan yang termasuk dalam *knowledge-based recommendation*, yaitu *case-based recommendation* (Ricci dan Nguyen, 2007; Smyth dkk., 2004) dan *constraint-based recommendation* (Felfernigdan Burke, 2008; Thompson dkk., 2004).

Case-based recommendation memandang rekomendasi sebagai sebuah permasalahan penentuan nilai *similarity*. Sistem mencari produk yang paling sama dengan yang pengguna inginkan dengan melibatkan *domain knowledge*. Sementara itu, *constraint-based recommendation* mempertimbangkan *constraint* yang didefinisikan secara eksplisit. Dalam *constraint-based recommendation*, rekomendasi dipandang sebagai proses pemenuhan *constraint* (Felfernig dan Burke, 2008). *Constraint* bisa berasal

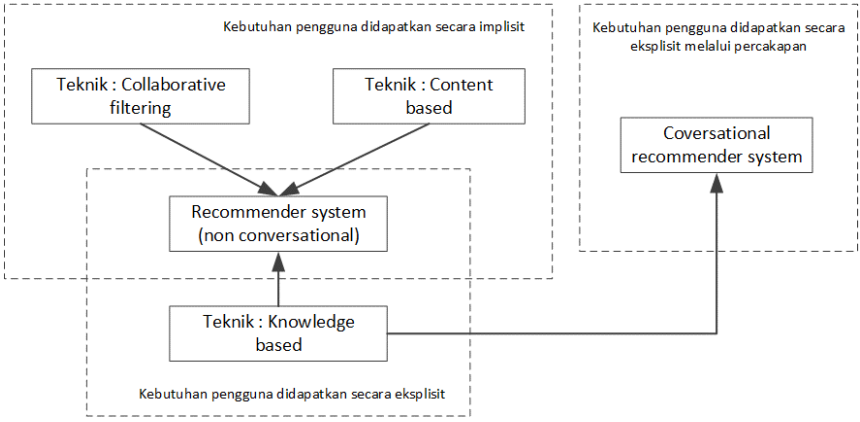
dari pengguna, bisa juga berasal dari produk. Rekomendasi yang baik adalah jika produk yang direkomendasikan dapat memenuhi *constraint*.

Interaksi dengan sebuah aplikasi *knowledge based recommender*, biasanya dimodelkan dalam bentuk sebuah dialog (*conversational recommender*). Pengguna dapat menspesifikasikan kebutuhan mereka dalam bentuk tanya jawab (Felfernig dkk., 2007; Mirzadeh dkk., 2005; Ricci dan Nguyen, 2007; Thompson dkk., 2004; Felfernig dan Burke, 2008). Berdasarkan himpunan kebutuhan pengguna yang diberikan, sistem memberikan solusi-solusi alternatif termasuk penjelasan tentang mengapa solusi tersebut diberikan. Dalam keadaan tertentu, dimungkinkan tidak ada solusi yang direkomendasikan. Dalam keadaan ini sekumpulan alternatif perbaikan diperoleh, yang membantu pengguna untuk menemukan jalan keluar. Aplikasi yang mendukung fungsionalitas rekomendasi tersebut, dalam banyak kasus dibangun dengan memanfaatkan teknik berbasis *constraint*, dengan fitur produk serta hubungan antara kebutuhan pengguna dan produk dimodelkan dalam bentuk *constraint*. Pendekatan berbasis *constraint* sangat membantu untuk merekomendasikan produk yang jarang dibeli. Selain itu, pendekatan ini juga sangat membantu dalam merekomendasikan produk yang lebih kompleks dan banyak pelanggan yang tidak mengetahui detil fitur secara teknis (Felfernig dan Burke, 2008). Felfernig dkk. (2007) menerapkan *constraint-based recommender* dalam domain layanan finansial (*loan recommender*). Jiang dkk. (2005) menyajikan sebuah pendekatan untuk multimedia *recommender* dengan menggunakan teknik *constraint-based*.

Pendekatan *constraint-based* dilengkapi dengan fungsionalitas visualisasi yang memungkinkan pengguna berinteraksi secara langsung dengan produk virtual. Masing-masing dari ketiga pendekatan tersebut mempunyai keterbatasan dan kelemahan (Adomavicius dan Tuzhilin, 2005). Oleh karena itu, beberapa *Recommender System* menggunakan sebuah pendekatan *hybrid recommendation* yang mengkombinasikan dua (atau lebih) teknik rekomendasi untuk mendapatkan kelebihan dari masing-masing teknik, atau untuk mengatasi kelemahan-kelemahan dari masing-masing teknik (Adomavicius dan Tuzhilinm, 2005; (Hsief dkk., 2004; Kazienko dan Kołodziejski, 2008; Ullah dkk., 2012).

Gambar II.1 menunjukkan ilustrasi yang merangkum keterkaitan beberapa teknik rekomendasi dengan *recommender system* maupun *conversational recommender sistem*. Banyak *recommender system* telah dikembangkan

berdasarkan teknik *collaborative filtering, content based*. Dengan kedua macam teknik ini, kebutuhan pengguna tidak perlu didapatkan secara eksplisit dari pengguna. Sistem mendapatkan kebutuhan pengguna dengan cara mempelajari perilaku pengguna berdasarkan *history* yang ada (*content based*) ataupun mempelajari pengguna lain yang mempunyai *taste* yang sama dengan pengguna (*collaborative filtering*). Namun kedua teknik ini terdapat kelemahan, terkait *cold start problem*. Oleh karena itu, teknik *knowledge based* menawarkan cara yang berbeda dalam mendapatkan kebutuhan pengguna.



Gambar 2.1 Keterkaitan beberapa teknik rekomendasi dengan *recommender system* maupun *conversational recommender system*.

Dalam *knowledge based recommender system*, Kebutuhan pengguna didapatkan secara eksplist dari pengguna, untuk mengatasi *cold start problem* ini. Sistem yang menanyakan kebutuhan pengguna hanya di awal interaksi, seperti yang biasa terjadi pada *knowledge based recommender system*, memunculkan kekurangan. Terdapat kecenderungan, pengguna belum dapat mengungkapkan kebutuhannya di awal interaksi (Bridge dkk., 2006), sehingga perlu adanya mekanisme *refinement query*, dan bahkan perlunya *feedback* pengguna dari produk yang direkomendasikan. Oleh karena itu, *knowledge based recommender system* telah banyak dikembangkan dalam bentuk *conversational recommender system* (CRS). Dalam CRS, kebutuhan pengguna didapatkan melalui sebuah mekanisme percakapan.

2.3 Conversational Recommender System

CRS dikembangkan dengan dukungan kemampuan sistem untuk berinteraksi secara intensif dengan pengguna. Dalam banyak riset yang terkait dengan mekanisme interaksi dalam CRS, strategi navigasi terbagi dalam dua macam, *navigation by asking* (NBA) dan *navigation by proposing* (NBP). Dalam strategi *navigation by asking*, penelitian berfokus pada pengembangan pendekatan untuk menghasilkan susunan pertanyaan yang efisien. Pertanyaan efisien dapat disusun menggunakan konsep *feature selection* dengan menggunakan *information gain* (Smyth dan Cunningham, 1994; Doyle dan Cunningham, 2000). Namun, dalam beberapa penelitian selanjutnya, urutan pertanyaan dibangun dengan mempertimbangkan kemungkinan reaksi pengguna terhadap pertanyaan yang diberikan, dengan metode *similarity of variance* (Kohlmaier dkk., 2001; Schmitt, 2002; Schmitt dkk,2002), *popularity of features* (Mirzadeh dkk., 2005), *weighted and majority voted* (Felfernig dan Burke, 2008). Penelitian-penelitian ini menunjukkan performansi yang baik dalam membangkitkan urutan pertanyaan yang efisien.

Dalam NBP, sistem merekomendasikan produk kepada pengguna dan sistem mendapatkan kebutuhan pengguna dalam bentuk *feedback* (kritik) terhadap produk yang direkomendasikan. Untuk membangkitkan interaksi yang efisien, beberapa peneliti memodifikasi *feedback* ini dalam bentuk *compound critiques* (Smyth dkk., 2004; McCarthy dkk., 2004; Reilly dkk., 2005), *preference-based feedback* (McGinty and Smyth, 2002), *incremental critiquing* (Llorente dan Guerrero, 2012) dan *experience-based unit critiquing* (Mandl dan Felfernig, 2012).

Kombinasi NBA dan NBP memungkinkan untuk membangkitkan interaksi yang dapat menirukan percakapan antara calon pembeli dengan *professional sales support* (Shimazu 2001; 2002). Dalam paper ini diungkapkan bahwa, seorang *profesional sales support* harus mampu berperan sebagai *problem solver*, *adviser* dan pemberi *testimony* (penjelasan). Sebagai seorang *problem solver*, dia harus mampu melakukan tanya jawab untuk mendapatkan produk sesuai kebutuhan pengguna (NBA). Sebagai seorang *adviser* dan pemberi *testimony*, dia harus bisa membantu calon pembeli dalam mengambil keputusan dengan memberikan contoh produk dan juga memberikan *testimony* (penjelasan) mengapa produk itu sesuai dengan kebutuhan calon pembeli (NBP). Untuk memilih pertanyaan yang efektif, dimanfaatkan konsep *information gain* (terkait dengan fitur

produk). Sementara untuk NBP, pendekatan statistik dimanfaatkan untuk menentukan produk-produk yang akan ditampilkan pada pengguna.

Hu dan Aufaure (2013) mengusulkan sebuah CRS dengan memadukan NBA dan NBP, untuk domain restoran. Penelitian ini mencoba untuk mengakomodasi kebutuhan pengguna yang bersifat *multi attribute* dan *multi criteria*. Selain itu, model yang dikembangkan dalam penelitian ini juga mengakomodasi *contextual information* sebagai salah satu pertimbangan dalam melakukan *query refinement,* disamping *hard constraint* dan juga *soft constraint*. Model pembangkit interaksi melibatkan konsep *Multi Attribute Utility Theory* (MAUT). Selain model pembangkit interaksi, penelitian ini juga mengusulkan suatu arsitektur ontologi dari *user preference model*. Pengembangan CRS dalam buku ini terinspirasi dari cara pemodelan *user preference model* yang merupakan hasil ekstraksi dari ontologi basis pengetahuan. Struktur ontologi pada Hu dan Aufaure (2013) bertujuan untuk memanfaatkan *contextual information* serta informasi terkait dengan *query refinement* dan bobot-bobot untuk menerapkan pendekatan MAUT. Selain itu, interaksi hanya berbasis pada fitur produk (restoran). Dengan demikian, struktur ontologi dalam penelitian tersebut tidak dapat diterapkan untuk menghasilkan sebuah interaksi CRS berbasis kebutuhan fungsional produk.

Tentunya struktur ontologi interaksi berbasis fitur produk memang efektif untuk produk-produk sederhana yang memiliki sedikit fitur, seperti baju dan *wine* (Shimazu, 2001; 2002), restoran (Hu dan Aufaure, 2013), dan sebagainya. Namun, untuk produk yang mempunyai banyak fitur, dan bersifat multi fungsi, seperti *smartphone*, *notebook*, server, kamera, mobil, dan sebagainya, tentu tidak semua pengguna familiar dengan fitur-fitur teknis produk, sehingga pertanyaan berdasarkan kebutuhan fungsional produk lebih natural untuk mendapatkan kebutuhan pengguna.

Banyak peneliti yang telah mengembangkan CRS yang berbasis pada kebutuhan fungsional produk, seperti FINDME (Hammond dkk., 1996; Burke dkk., 1997; Burke 2002), dan ADVISOR SUITE (Jannach, 2004; Felfernig dkk., 2006; Jannach dan Kreutler, 2004; 2007), dengan melibatkan mekanisme pemetaan antara produk dengan fitur teknis. FINDME memungkinkan pengguna untuk mengungkapkan high level features (misal mencari rumah makan *casual*), dan sistem menerjemahkan arti casual ini melalui pengetahuan yang ada. Namun, sistem menanyakan kebutuhan fungsional hanya di awal interaksi, dan mendapatkan kebutuhan

pengguna selanjutnya dengan strategi NBP. ADVISOR SUITE memberikan *personalized interaction* menggunakan pendekatan *rule-based*. Namun, interaksi dibangkitkan menggunakan strategy NBA, dan tidak mengakomodasi kritik pengguna terhadap produk yang direkomendasikan.

Buku ini menyajikan sebuah pengembangan CRS dengan memadukan NBA dan NBP, dengan cara membangkitkan pertanyaan, mengajukan sampel produk serta penjelasan mengapa suatu produk direkomendasikan, dengan menggunakan kebutuhan fungsional sebagai panduan utama selama proses interaksi. Dengan demikian mekanisme interaksi yang dihasilkan dapat berperan layaknya seorang *professional sales support* yang dapat menggali kebutuhan pengguna dari kebutuhan fungsional dalam suatu kerangka percakapan yang lebih natural. Untuk mendukung hal ini, buku ini juga menyajikan struktur ontologi yang merupakan basis pengetahuan dari sistem.

2.4 *Constraint Satisfaction Problem*

CRS merupakan salah satu bentuk dari *knowledge based-recommender system*. *Knowledge-based recommendation* menggunakan *domain knowledge* untuk mencari hubungan (kesesuaian) antara kebutuhan pengguna dengan produk tertentu. Terdapat dua jenis *knowledge based recommender system*, yaitu *constraint-based recommender system* (Zanker dkk., 2010; Felfernig dkk, 2006; Felfernig dan Burke, 2008) dan *case-based recommender system* (Bridge dkk., 2005; Burke, 2000; Smyth dan Cunningham, 1994; Doyle dan Cunningham, 2000; Kohlmaier dkk., 2001; Schmitt, 2002; Shimazu, 2002). Dalam *knowledge based recommender system*, pengguna harus mengungkapkan kebutuhan secara eksplisit, dan berdasarkan basis pengetahuan, sistem merekomendasikan produk yang sesuai dengan kebutuhan pengguna. Dalam merekomendasikan produk, terdapat perbedaaan antara *case-based* dan *constraint-based recommender system*. Dalam *case based recommender,* sistem merekomendasikan produk berdasarkan nilai *similarity* tertentu, sedangkan *constraint-based recommender system* merekomendasikan produk berdasarkan himpunan *recommendation rules* yang telah didefinisikan.

Tabel 2.1 Daftar fitur produk kamera (Felfernig dkk., 2009)

id	price	mpix	Opt-zoom	LCD-size	movies	sound	waterproof
p1	148	8.0	4x	2.5	No	No	Yes
p2	182	8.0	5x	2.7	Yes	Yes	No
p3	189	8.0	10x	2.5	Yes	No	No
p4	196	10.0	12x	2.7	Yes	Yes	Yes
p5	151	7.1	3x	3.0	Yes	Yes	No
p6	199	9.0	3x	3.0	Yes	Yes	No
p7	259	10.0	3x	3.0	Yes	Yes	No
p8	278	9.1	10x	3.0	Yes	Yes	Yes

Sebagai salah satu pendekatan dalam *knowledge-based recommender* system, *constraint-based recommender* melibatkan pengetahuan detil dari fitur produk. Sebagai contoh, Tabel 2.1 menunjukkan sebuah daftar fitur dari produk kamera (Felfernig dkk., 2009)

Sebuah *constraint satisfaction problem* (CSP) (Burke, 2012) dapat diformalisasikan sebagai sebuah *tuple* (*V*, *D*, *C*), dengan :

V = himpunan variabel
D = himpunan domain dari *V*
C = himpunan constraint yang menggambarkan kombinasi dari nilai-nilai variabel
$V = V_C \cup V_{PROD}$ dan $C = C_R \cup C_F \cup C_{PROD}$

Tabel 2.2 menunjukkan variabel-variabel dan *constraints* untuk *digital camera recommender* (Jannach, 2004)

Tabel 2.2 Variabel dan *constraints* untuk *digital camera* (Jannach, 2004)

V_C	{max-price(0..100), usage(digital, small-print, large-print), photography (sports, landscape, portrait, macro)}
V_{PROD}	{price(0..1000), mpix(3.0 .. 12.0), opt-zoom(4x .. 12x), lcd-size(2.5 .. 3.0), movies(yes, no), sound(yes, no), waterproof(yes, no)
C_F	{usage = large-print → mpix > 5.0} {usage : property dari pelanggan, dan mpix adalah property dari produk)
C_R	{usage = large-print → max-price > 200) (usage dan max-price adalah property dari pelanggan)

C_{PROD}	{(id=p1 ∧ price=148 ∧ mpix=8.0 ∧ opt-zoom=4x ∧ lcd-size=2.5 ∧ movies=no ∧ sound=no ∧ waterproof=no) ∨ {(id=p8 ∧ price=278 ∧ mpix=9.1 ∧ opt-zoom=10x ∧ lcd-size=3.0 ∧ movies=yes ∧ sound=yes ∧ waterproof=yes)}
REQ	{max-price = 300, usage = large-print, photography = sports}
RES	{max-price=300, usage=large-print, photography=sports, id=p8, price=278, mpix=9.1, opt-zoom=10x, lcd-size=3.0, movies=yes, sound=yes, waterproof=yes

V_C = *property* dari pelanggan, merupakan kebutuhan-kebutuhan pelanggan yang mungkin

V_{PROD} = *property* dari produk, merupakan fitur-fitur dari produk

C_R = *compatibility constraints*, merupakan nilai-nilai dari *property* pelanggan yang *compatible* dengan nilai-nilai *property* pelanggan yang telah didefiniskan. Misal jika dibutuhkan *large-print photoprint*, maka harga harus lebih dari 200

C_F = *Filter conditions*, merupakan hubungan antara *property* pelanggan dan *property* produk. Misal jika membutuhkan *large-size photoprints*, maka resolusi harus lebih dari 5 mpix.

C_{PROD} = *product constraints*, merupakan produk-produk yang tersedia.

RES = *recommendation*, merupakan produk yang direkomendasikan

Pengembangan CRS yang disajikan dalam buku ini, mengambil konsep *constraint satisfaction problem* (CSP) dalam *constraint-based recommender system*. Rekomendasi produk dipandang sebagai pemenuhan *constraint* yang terdapat pada *user profile model*. Rekomendasi dilakukan berdasarkan *rules* yang secara implisit terdapat dalam ontologi (sebagai basis pengetahuan). Dalam penelitian-penelitian terkait dengan CSP, sejauh ini belum ditemukan penelitian yang mengembangkan CRS dengan memadukan NBA dan NBP. Jannach (2004) telah mengembangkan CRS dengan memanfaatkan CSP, sehingga memungkinkan interaksi mengacu pada kebutuhan fungsional, dengan strategy NBA. Tidak ada interaksi yang mengakomodasi kemungkinan adanya *feedback* pengguna terhadap produk yang direkomendasikan. Ontologi dipandang sesuai sebagai bentuk dari basis pengetahuan, karena pemetaan antara kebutuhan fungsional dengan produk, serta *path* untuk pembangkitan pertanyaan dapat direpresentasikan melalui relasi-relasi semantik yang ada dalam ontologi. Dengan demikian,

dapat dibangun model pembangkit interaksi, yang meliputi pemberian pertanyaan, rekomendasi produk dan pemberian penjelasan, yang merepresentasikan perpaduan NBA dan NBP.

2.5 *Query Refinement* dalam CRS

Motivasi utama dari pengembangan CRS adalah adanya kemungkinan pelanggan (calon pembeli) masih belum dapat mengungkapkan kebutuhan awalnya di awal interaksi (Bridge dkk, 2006). Melalui mekanisme percakapan, CRS harus mempunyai kemampuan memandu pengguna dalam mengungkapkan kebutuhan. Menurut Shimazu, kebutuhan pengguna dapat *over-specified* maupun *under-specified*. Jika *overspecified*, maka tidak ada produk yang sesuai dengan kebutuhan, sementara itu jika *under-specified*, terlalu banyak produk yang sesuai dengan kebutuhan. Dalam keadaaan *over-specification*, CRS harus melakukan *relaxation* terhadap kebutuhan pengguna, sedangkan dalam keadaaan *under-specification*, CRS harus melakukan *query refinement*, untuk mengerucutkan (*refining*) kebutuhan pengguna.

Efisiensi CRS dapat dilihat dari kemampuan model interaksi dalam melakukan *query refinement*. Evaluasi terhadap mekanisme *query* (kebutuhan) *refinement* dilihat dari kemampuan model dalam mengurangi sisa jumlah *record* dari setiap interaksi (Hu dan Aufaure 2013; Mirzadeh dkk., 2005, Shimazu, 2002, McGinty, 2002). Gambar II.2 adalah contoh percakapan, sebagai mekanisme *query refinement* yang dilakukan oleh sistem *ExpertClerk* (Shimazu, 2002).

Mekanisme *query refinement* dapat dilakukan dengan cara memberikan pertanyaan-pertanyaan agar kebutuhan pengguna dapat lebih spesifik (NBA) atau dapat juga dengan memberikan contoh-contoh produk serta memberikan penjelasan tentang produk tersebut (NBP). Dalam banyak penelitian yang telah dikembangkan, penelitian bertujuan untuk menghasilkan interaksi yang efisien, dengan semakin sedikit *cycle* interaksi, atau dapat direpresentasikan dengan *trend* penurunan jumlah *record* dari setiap interaksi. Dalam buku ini, performa efisiensi dilihat dari *trend* penurunan jumlah sisa *record* per tahap interaksi. Namun karena dalam CRS yang dikembangkan memungkinkan pengguna untuk mengubah kebutuhan di tengah percakapan, jadi evaluasi dilakukan hanya pada saat strategi NBA dijalankan, sampai direkomendasikan produk.

Shopper : Please show me that blouse
ExpertClerk :
Ok, here it is. We have suit color that may also suit you. And this is a blouse of the same type but with a different design
Shopper : Well, the design is fine, but the neck looks very tight
SalesClerk :
How about this one, if you prefer a loose neckband? That one also has a loose neckband and interesting design
Shopper : I like this one. How much is it?
SalesClerk : It is $200
Shopper : Wow $200 is too expensive
.....

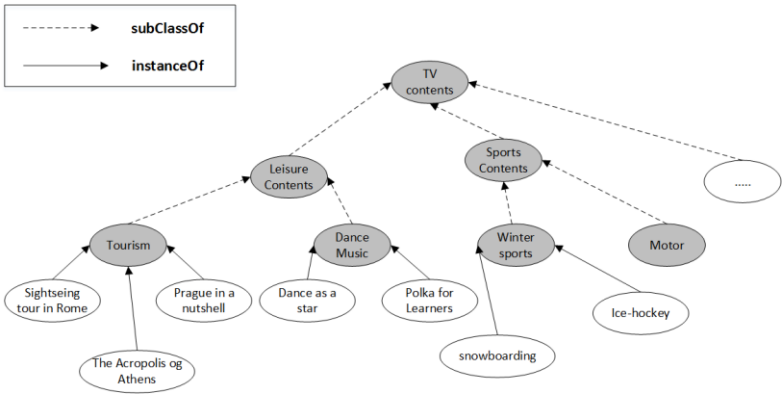
Gambar 2.2 Contoh percakapan untuk *query refinement* dalam *expertclerk* (Shimazu, 2002)

2.6 *Recommender System* Berbasis pada Ontologi

Untuk dapat menggali kebutuhan pengguna dari sisi kebutuhan fungsional, *knowledge base* yang digunakan berupa sebuah ontologi. Ontologi didefinisikan sebagai konseptualisasi dari domain pengetahuan. Elemen-elemen pengetahuan dinyatakan dalam *entity*, *attributes*, *relationships* dan *axiom* yang dapat dipahami oleh manusia maupun mesin (Guarino dan Giaretta, 1995). Kelebihan dari ontologi adalah kemampuannya memberikan gambaran konseptual dari domain pengetahuan. Berdasarkan definisi ini, hubungan antar entitas dinyatakan dalam sebuah relasi semantik. Relasi semantik mampu menghubungkan entitas-entitas dalam pengetahuan dengan lebih fleksibel, sehingga diharapkan dapat mendukung terbentuknya sebuah mekanisme interaksi yang lebih natural.

Dalam beberapa penelitian yang terkait dengan *recommender system* dengan menggunakan ontologi sebagai basis pengetahuan, mesin inferensi

digunakan untuk menyimpulkan fakta berdasarkan ontologi domain dan *rules* yang dibangun (Matobuwana dkk., 2009; Kato dkk., 2010; Chen dkk., 2012; Vesin dkk., 2012)..

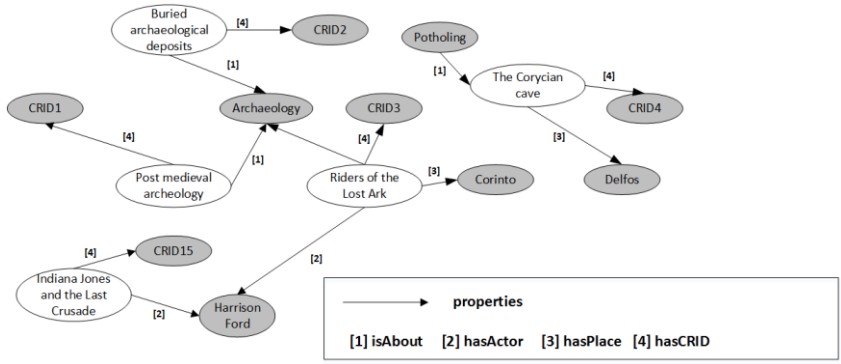


Gambar 2.3 Contoh potongan ontologi program TV dengan struktur multi hirarki (Blanco-Fernadez dkk., 2008a; 2008b)

Telah banyak penelitian yang menggunakan ontologi sebagai basis pengetahuan dalam *recommender system*. Penalaran untuk rekomendasi produk pun dilakukan dengan menelusuri relasi antar *entity* pada ontologi, yang sering disebut sebagai *semantic reasoning*. Ontologi domain merupakan pengetahuan dari sistem. *User profile model* dibangun dengan *domain ontology reuse*, yang memuat fakta-fakta terkait dengan *reasoning*. Struktur dari *domain ontology* dapat berupa sebuah hirarki saja, dalam arti hanya memuat hubungan *class-subclass* saja (Weng dan Chang, 2008; Yang, 2010; Moreno dkk., 2012), maupun berupa hirarki serta relasi antar hirarki (Lee dkk., 2009; Castillo dkk., 2008; Blanco-Fernadez dkk 2008a; 2008b). Struktur ontologi ini mengandung relasi semantik yang lebih kaya, untuk mendukung pendekatan *semantic reasoning* yang dikembangkan.

Dalam Blanco-Fernandez dkk. (2008), strategy berbasis *reasoning* ini dilakukan dengan mengacu pada *domain ontology* (basis pengetahuan) dan *user model* (fakta). Penelitian ini mengambil domain program acara televisi. Gambar II.3 menunjukkan contoh potongan *domain ontology* program TV dalam penelitian tersebut, dengan menerapkan struktur multi hirarki, sedangkan Gambar 2.4 menunjukkan contoh relasi (direpresentasikan

dengan *property*) antar *instance* dalam ontologi. Dalam penelitian ini, *user model* berisi informasi program – program TV yang disukai maupun tidak disukai oleh pengguna (dinyatakan dalam preferensi positif dan negatif), atribut-atribut utama, dan *genre* program TV. *Reasoning* yang dilakukan, terdiri dari dua tahap, yaitu tahap *filtering* dan tahap rekomendasi.



Gambar 2.4 Contoh hubungan antar *instance* (individu) dalam ontologi (Blanco-Fernadez dkk., 2008a; 2008b)

Tahap *filtering* : Berawal dari *class* program TV yang ada dalam *user model*, sistem menelusuri *properties* sampai pada *instances* baru dalam *domain ontology*. Kemudian sistem akan menentukan nilai relevansi dari *node* yang didapatkan. Jika nilai relevansi tidak lebih besar dari suatu nilai batas yang ditentukan, maka dilakukan inferensi lagi dengan cara menelusuri kembali *properties* dalam ontologi untuk mencari *node* (*instance*) baru, dan seterusnya. Dalam penelitian ini, terdapat 4 macam cara menentukan nilai relevansi, (i) berdasarkan panjang *path* dari properties, (ii) berdasarkan *degree of interest* (skala [-1,1], semakin mendekati -1 semakin tidak disukai, semakin mendekati 1, semakin disukai), (iii) hubungan hirarkis antar *node*, (iv) berdasarkan nilai *betweeness*.

Tahap rekomendasi : Untuk merekomendasikan program TV ke pengguna, diterapkan *spreading activation techniques* (SA). Teknik SA ini akan menentukan nilai relevansi lebih lanjut dari masing-masing *node* yang telah didapatkan pada tahap filtering, dengan mempertimbangkan hubungan hirarkis antar *node*.

Jika *user profile model* pada penelitian-penelitian di atas dibangun berdasarkan *rating* pengguna terhadap produk, Hu dan Aufaure (2006) mengembangkan sebuah struktur ontologi dalam sebuah CRS. *User profile* dibangun berdasarkan kebutuhan yang dinyatakan secara eksplisit dari pengguna.

Model dalam pembangunan CRS juga memanfaatkan ontologi domain sebagai basis pengetahuan. Selain itu, *user profile model* merupakan komponen penting dalam sistem untuk belajar tentang kebutuhan pengguna selama interaksi berjalan. *User profile model* ini memuat *log* kebutuhan pengguna selama sesi percakapan, yang berperan sebagai *constraint* (dalam konteks rekomendasi produk) atau fakta (dalam konteks *reasoning*). Berbeda dengan penelitian-penelitian sebelumnya, model yang dikembangkan dalam membangun CRS, memanfaatkan kebutuhan fungsional produk sebagai dasar untuk interaksi percakapan. Interaksi ini dihasilkan dengan memadukan strategi NBA and NBP, sehingga dapat berjalan layaknya percakapan antara pelanggan dengan *professional sales support*. *User profile model* dibangun berdasarkan interaksi ini.

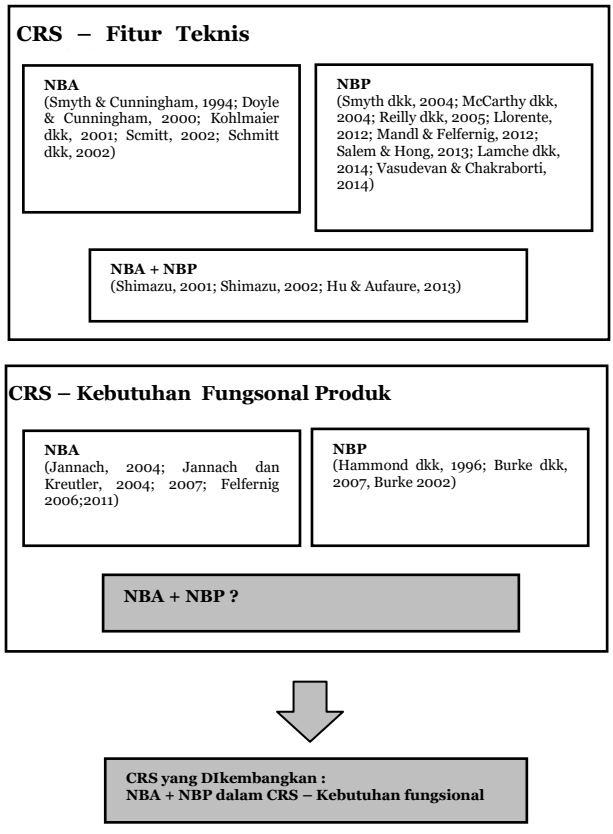
2.7 Resume

Recommender system telah banyak dikembangkan dalam banyak penelitian, untuk membantu pengguna mendapatkan *personalized item recommendation*. Teknik-teknik yang cukup terkenal dikembangkan adalah *collaborative* dan *content based filtering*. Namun kedua teknik ini membutuhkan data *history*, baik data pengguna ataupun produk, sehingga *cold start problem* masih menjadi kelemahan utama kedua teknik yang populer ini. *Knowledge-based recommender system* merupakan salah satu solusi untuk mengatasi kelemahan yang terkait dengan *cold start problem*. Dalam hal ini, sistem mendapatkan kebutuhan pengguna secara eksplisit dari pengguna (Felfernig dkk., 2015). Namun, dalam kenyataan, pengguna sering belum dapat mengungkapkan kebutuhannya secara pasti di awal interaksi, selain itu, pengguna juga mungkin belum puas dengan rekomendasi yang diberikan oleh sistem. Dengan demikian, sistem perlu mengadopsi mekanisme percakapan untuk menggali kebutuhan pengguna. Sistem ini disebut sebagai *conversational recommender system* (CRS).

Dalam CRS, terdapat dua macam strategi untuk membangkitkan interaksi pengguna-sistem, yaitu *navigation by asking* (NBA) dan *navigation by proposing* (NBP) (Bridge dkk., 2006). Dalam NBA, sistem secara iteratif

memberikan pertanyaan untuk menggali kebutuhan pengguna, dan interaksi diakhiri dengan sesi rekomendasi. Sementara itu, dalam NBP, sistem menggali kebutuhan pengguna melalui pemberian sampel produk. Berdasarkan *feedback* pengguna terhadap produk yang direkomendasikan, sistem memberikan sampel produk kembali yang mendekati dengan kebutuhan pengguna.

Kombinasi NBA dan NBP merupakan suatu solusi untuk menciptakan sebuah sistem yang dapat berperan layaknya seorang *professional sales support*. Dalam dunia nyata, seorang *professional sales support* harus mampu memberikan pertanyaan-pertanyaan, memberikan sampel produk kepada calon pembeli, beserta penjelasan mengapa produk-produk tertentu direkomendasikan. Dengan demikian, sistem harus mampu melakukan *switch* antara NBA dan NBP, karena pemberian pertanyaan maupun pemberian sampel produk dapat berlangsung secara berulang. Shimazu (2001; 2002) merupakan salah satu pelopor dalam penelitian pengembangan CRS yang mengkombinasikan NBA dan NBP. Hu dan Aufaure (2013) juga telah memperkenalkan sebuah *CRS* yang terdiri dari model pembangkit interaksi yang mampu mengakomodasi kebutuhan yang bersifat *multi attribute*, serta arsitektur ontologi dari *user profile model* yang merupakan hasil ekstraksi dari ontologi basis pengetahuan.



Gambar 2.5 Ruang lingkup dari CRS yang dikembangkan

Namun interaksi yang dihasilkan pada penelitian-penelitian di atas, berbasis pada fitur produk secara langsung. Untuk produk *hi-tech*, mempunyai banyak fitur dan multi fungsi seperti laptop, server, kamera, mobil, *smartphone*, tentu tidak semua pengguna familiar dengan fitur teknis produk. Cara yang lebih natural untuk menggali kebutuhan pengguna adalah dengan menanyakan peruntukan dari produk yang akan dicari (kebutuhan fungsional produk).

Untuk lebih jelas tentang ruang lingkup dari CRS yang dikembangkan, dapat dilihat pada Gambar 2.5. Ide mekanisme interaksi dalam CRS, terinspirasi oleh mekanisme interaksi yang dikembangkan Shimazu (2002) dan Hu dan Aufaure (2013). Namun mekanisme interaksi berdasarkan kebutuhan fungsional tentu lebih kompleks daripada fitur teknis produk gambar 2.5 menunjukkan pengelompokan metode-metode yang telah dikembangkan dalam mendukung pembangkitan interaksi pada CRS. Pendekatan *rule-based* sebenarnya merupakan pendekatan yang sesuai juga untuk membangkitkan interaksi berbasis pada kebutuhan fungsional. Namun pendekatan ini kurang fleksibel untuk pembangkitan interaksi pada saat pengguna memberikan *feedback* terhadap produk yang direkomendasikan (NBP). Selain itu, pendekatan *rule-based* akan sulit dimanfaatkan untuk membangun CRS yang nantinya dapat diaplikasikan untuk domain berbeda. Sebagai sebuah *knowledge-based recommender system*, CRS yang dibangun harus didukung oleh suatu pengetahuan yang mampu memetakan kebutuhan fungsional - fitur teknis produk – produk, dan juga mampu membangkitkan pertanyaan berdasarkan kebutuhan fungsional. Representasi pengetahuan menggunakan ontologi dipilih untuk menangani hal ini. Perpaduan ontologi dengan *rules set* menciptakan sebuah pendekatan *rule based* yang mampu membangkitkan interaksi berdasarkan kebutuhan fungsional. Namun cara ini akan menjadi kurang fleksibel jika suatu saat akan membangun CRS pada domain berbeda.

Struktur hirarki dalam memanfaatkan struktur ontologi multi hirarki, tanpa melibatkan *rules set* (Blanco Fernandez 20081; 2008b; Hu dan Aufaure, 2013). Struktur ontologi ini mengandung relasi semantik yang lebih kaya, untuk mendukung pendekatan *semantic reasoning* dalam rangka pengembangan model pembangkit interaksi. Sistem memanfaatkan relasi-relasi semantik dalam ontologi untuk membangkitkan interaksi.

Ontologi merupakan suatu relasi pengetahuan yang mampu membuat pemetaan kebutuhan fungsional – produk, serta memungkinkan untuk penyusunan suatu konsep secara hirarkis. Struktur hirarkis ini berguna untuk merepresentasikan konsep-konsep kebutuhan fungsional, yang nantinya terkait dengan pembangkitan pertanyaan. Selain itu, dengan struktur ini, akan lebih memudahkan dalam pembangunan sebuah *generic ontologi structure*, sehingga mudah jika akan diimplementasikan untuk domain yang berbeda (yang mempunyai karakteristik yang sama). Dalam pembangunan CRS ini, karakteristik domain yang dapat diterapkan adalah produk yang bersifat multifungsi dan mempunyai banyak fitur.

3

FORMALISASI ONTOLOGI

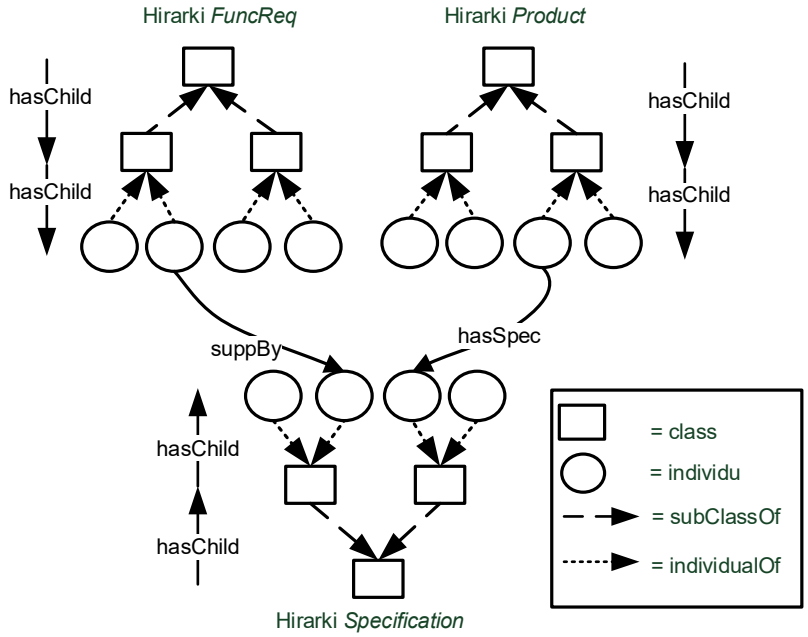
3.1 Pendahuluan

Pembahasan mengenai ontologi meliputi pembahasan struktur ontologi secara umum, definisi beberapa relasi, fungsi-fungsi dasar dalam ontologi, aksioma-aksioma yang berlaku pada ontologi, dan definisi formal dari ontologi. Ontologi dibangun dengan mengikuti standard *ontology web language* (OWL). Dalam bab ini akan dibahas unsur-unsur penting dalam formalisasi ontologi, sehingga dapat merupakan pedoman dalam pengembangan ontologi pada domain yang lain.

3.2 Struktur Ontologi

Buku ini menyajikan suatu struktur ontologi yang mendukung mekanisme interaksi yang mampu menggali kebutuhan pengguna dari sisi kebutuhan fungsional, dan model pembangkitan interaksi dalam CRS yang meliputi interaksi tanya jawab, proses rekomendasi beserta fasilitas penjelasan. Penelitian ini mengadopsi struktur multi hirarki (Blanco-Fernandez dkk., 2008a; 2008b) untuk memetakan kebutuhan fungsional dengan produk, dengan mengikuti standar ontology web language (OWL).

Struktur ontologi terdiri dari tiga *classes* utama, *FuncReq*, *Products* dan *Specification*. Semua *classes* tersebut mempunyai *subclasses* dan individu yang merepresentasikan sebuah hirarki. *Class FuncReq* merepresentasikan kebutuhan fungsional dari pengguna, *class Product* merepresentasikan produk yang ada dalam domain, dan *class Specification* merepresentasikan spesifikasi yang memetakan antara kebutuhan fungsional dan produk. Masing-masing hirarki dari *class* ini mempunyai struktur *rooted tree*, dengan internal *nodes* merupakan *class*, sedangkan daun merupakan individu. *Class* merupakan deskripsi dari konsep dalam suatu domain, sedangkan *individu* merupakan realisasi dari konsep. Untuk selanjutnya, didefinisikan himpunan *node* sebagai gabungan dari himpunan *class* dan himpunan individu. Struktur baku ontologi, diilustrasikan pada Gambar 3.1.



Gambar 3.1 Struktur ontologi

Dalam ontologi terdapat dua macam relasi, relasi antar hirarki dan relasi dalam hirarki. Himpunan relasi antar ontologi adalah relasi yang menghubungkan 2 individu antar hirarki, yang terdiri atas *suppBy*, dan *hasSpec*. Himpunan relasi dalam hirarki adalah relasi yang menghubungkan dua *node* dalam masing-masing hirarki, yang terdiri dari relasi *individualOf*, *subClassOf* dan *hasChild*.

Struktur ontologi ini bersifat *generic*, dan dapat diterapkan dalam berbagai domain, khususnya domain produk yang bersifat multi fungsi dan mempunyai banyak fitur (*notebook*, PC, *server*, kamera, mobil, *smartphone*, dsb). Dalam buku ini, dibuat contoh model ontologi dalam domain *smartphone*.

Ontologi dibangun dengan melibatkan *domain expert*, khususnya dalam bidang *smartphone*. Terdapat tiga macam pengetahuan yang diperlukan dari *expert* : 1) Macam-macam kebutuhan fungsional dalam domain *smartphone*, 2) Tingkatan kualitas dari masing-masing spesifikasi, 3) pemetaan antara kebutuhan fungsional dan spesifikasi.

3.2.1 Relasi

Nodes dalam ontologi dihubungkan dengan relasi biner, yang merupakan relasi dari 2 argumen. Himpunan relasi dalam ontologi dinotasikan dengan *Rel*, sedangkan himpunan *node* dinotasikan dengan $\mathcal{N}$. Sebuah relasi $r \in Rel$ yang menghubungkan dua *node* x dan y didefinisikan sebagai berikut :

$$r(x, y) = \{\langle x, y\rangle | x, y \in \mathcal{N}\} \qquad (3.1)$$

Himpunan relasi antar *node* dalam suatu hirarki dinyatakan dengan $\mathcal{R}_H$. Himpunan relasi ini terdiri dari relasi *subClassOf*, *individualOf* dan *hasChild*. Definisi dari ketiga relasi ini disajikan dalam Definisi 3.1, 3.2 dan 3.3.

Definisi 3.1 (Relasi *subClassOf*). Relasi *subClassOf* menyatakan bahwa suatu *class* tertentu merupakan *subclass* dari *class* yang lain. Misal C adalah himpunan *classes*, sedemikian hingga untuk $x, y \in C$, relasi ***x is subclass* of *y*** didefinsikan sebagai berikut:

$$subClassOf\left(x, y \middle| \{\langle x, y\rangle \in C, x \text{ adalah } descendant \text{ dari } y\}\right) \quad (3.2)$$

Definisi 3.2 (Relasi *individualOf*). Sebuah *class* adalah deskripsi dari konsep dalam domain, sedangkan individu adalah sebuah realisasi dari konsep. Relasi *individualOf* menyatakan bahwa sebuah *node* tertentu adalah individu dari sebuah *class*. Misal $\mathcal{I}$ menyatakan himpunan individu dan C

adalah himpunan *class*. Dengan demikian untuk $x \in I$, $y \in C$, relasi ***x is individual of y*** didefinisikan sebagai berikut :

$$individualOf(x, y) = \{\langle x, y\rangle | x \in \mathcal{I}, y \in C, x \text{ adalah } descendant \text{ dari } y\} \quad (3.3)$$

Definisi 3.3 (Relasi *hasChild*). Relasi *hasChild* menghubungkan sebuah *node* dengan *direct descendant*-nya. Misal C dan $\mathcal{I}$ adalah himpunan *class* dan himpunan individu, dan diketahui $\mathcal{N} = C \cup \mathcal{I}$, maka relasi ***x has child y*** didefinisikan sebagai berikut :

$$hasChild(x, y) = \{\langle x, y\rangle | x \in C, y \in \mathcal{N}, y \text{ adalah } direct\ descendant \text{ dari } x\} \quad (3.4)$$

Himpunan relasi antar *node* dari hirarki-hirarki yang berbeda, yang diyatakan dengan $\mathcal{R}_O$, terdiri dari relasi *suppBy* dan *hasSpec*. Keduanya diperlukan untuk memetakan kebutuhan fungsional dengan produk. Definisi-definisi dari kedua relasi ini disajikan dalam Definisi 3.4 dan 3.5.

Definisi 3.4 (Relasi *suppBy*). Misal $\mathcal{I}_{spec}$ dan $\mathcal{I}_{func}$ adalah himpunan individu dalam hirarki *Specification* dan *FuncReq*. Relasi *suppBy* menghubungkan sebuah elemen dari $\mathcal{I}_{func}$ dengan sebuah elemen dari $\mathcal{I}_{spec}$. Relasi ini menyatakan bahwa sebuah kebutuhan fungsional didukung oleh sebuah spesifikasi tertentu. Misal $x \in \mathcal{I}_{func}$, $y \in \mathcal{I}_{spec}$, relasi ***x is supported by y*** didefinsikan sebagai berikut:

$$suppBy(x, y) = \{\langle x, y\rangle | x \in \mathcal{I}_{func}, y \in \mathcal{I}_{spec}\} \quad (3.5)$$

Definisi 3.5 (Relasi *hasSpec*). Relasi *hasSpec* menghubungkan sebuah elemen dari himpunan individu dalam hirarki *Product* ($\mathcal{I}_{prod}$) dengan sebuah elemen dari $\mathcal{I}_{spec}$. Relasi ini menyatakan bahwa sebuah produk mempunyai sebuah spesifikasi tertentu. Misal $x \in \mathcal{I}_{prod}$, $y \in \mathcal{I}_{spec}$, relasi ***x has specification y*** didefiniskan sebagai berikut:

$$hasSpec(x, y) = \{\langle x, y\rangle | x \in \mathcal{I}_{prod}, y \in \mathcal{I}_{spec}\} \quad (3.6)$$

Dari definisi relasi *hasChild* dan *individualOf*, dapat diturunkan fungsi *children*, *parents* dan *allInd*, yang disajikan dalam Definisi 3.6, 3.7 dan 3.8.

Definisi 3.6 (Fungsi *children*). Fungsi *children* merupakan fungsi yang menghasilkan himpunan semua *anak* dari himpunan *class* tertentu dalam sebuah hirarki. Misal $\mathcal{N}$ adalah himpunan *node* dan C adalah himpunan *class*, dan dikeyahui $\mathcal{N} = C \cup I$. Dengan demikian fungsi ini memetakan C ke $\mathcal{N}$ sebagai berikut:

$$children : C \rightarrow \mathcal{N}$$

Misal diketahui himpunan *node* $A=\{a_i\}$, $i > 0$. Dengan demikian *children* dari himpunan *node* A didapatkan dengan,

$$children(A) = \bigcup_{a_i \in A} \left\{ x \in \mathcal{N} \middle| hasChild(a_i, x) \right\} \qquad (3.7)$$

Definisi 3.7 (**Fungsi *parents***) Fungsi *parents* merupakan sebuah fungsi untuk mendapatkan himpunan *parents* dari sebuah himpunan *node* (*invers* dari fungsi *children*). Misal $\mathcal{N}$ menyatakan himpunan *node*, dan $\mathcal{N} = C \cup I$, dengan demikian fungsi *parents* memetakan himpunan $\mathcal{N}$ ke himpunan C sebagai berikut:

$$parents : \mathcal{N} \rightarrow C$$

Misal diketahui himpunan *node* $A=\{a_i\}$, $i >0$, maka *parents* dari himpunan *node* A,

$$parents(A) = \bigcup_{a_i \in A} \left\{ x \in \mathcal{N} \middle| hasChild^{-1}(a_i, x) \right\} \qquad (3.8)$$

Definisi 3.8 (Fungsi *allInd*). Fungsi *allInd* digunakan untuk mendapatkan himpunan individu (I) dari suatu himpunan *class* (C). Fungsi ini memetakan himpunan $\mathcal{N}$ ke himpunan I sebagai berikut,

$$allInd : C \rightarrow I$$

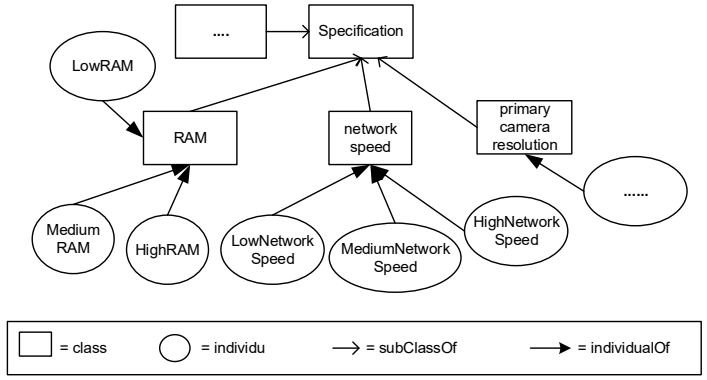
Sebagai contoh, diketahui himpunan *node* $A=\{a_i\}$, $i> 0$, individu-individu dari himpunan A didapatkan melalui:

$$allInd(A) = \bigcup_{a_i \in A} \left\{ x \in \mathcal{N} \middle| individualOf(x, a_i) \right\} \qquad (3.9)$$

3.2.2 Hirarki *Class Specification*

Hirarki *class Specification* merepresentasikan hirarki dari spesifikasi produk dalam domain. Hirarki ini digunakan untuk memetakan kebutuhan fungsional dengan produk yang sesuai. Aturan penyusunan *internal nodes* dan daun dalam hirarki *Specification* disajikan dalam Aksioma 3.1.

Aksioma 3.1. Hirarki *Specification* mempunyai tiga level, dengan level 0 merupakan *class Specification* sebagai sebuah *class* akar, *nodes* pada level 1 berupa *class-class* yang mendefinisikan jenis spesifikasi, sementara itu level 2 (daun) merupakan individu yang mendefinisikan tingkatan kualitas dari jenis spesifikasi secara kualitatif.



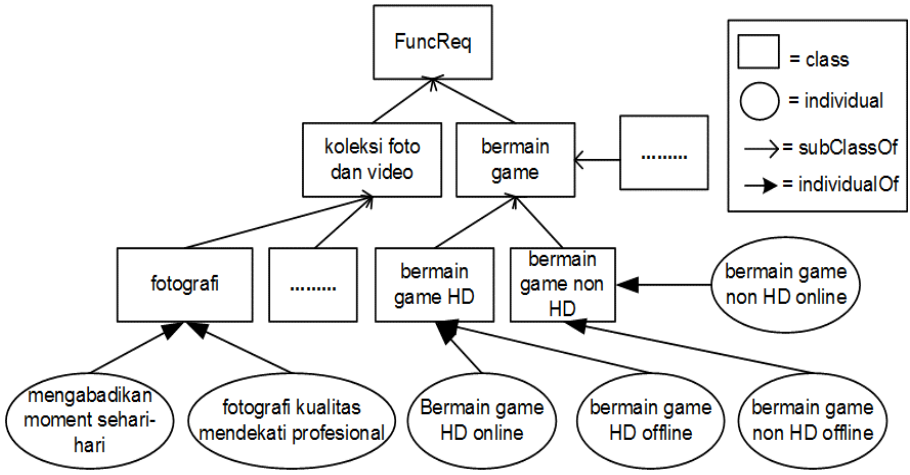
Gambar 3.2 Contoh hirarki *Specification* dalam domain *smartphone*.

Dalam diskusi selanjutnya, himpunan *class* dalam hirarki *Specification* dinyatakan dengan C_{spec}, sementara itu himpunan individu dalam hirarki *Specification* dinyatakan dengan I_{spec}. Individu dalam hirarki *Specification* menyatakan tingkatan kualitas dari jenis spesifikasi (misal Low RAM, Medium RAM, High RAM), karena pemetaan nilai kuantitatif dari jenis spesifikasi dan kebutuhan fungsional yang didukung, sulit untuk dilakukan normalisasi. Sebagai contoh, tingkat kualitas dari RAM dengan kapasitas 1 GB untuk perangkat iOS tentu berbeda dengan RAM dengan kapasitas yang sama untuk perangkat Android. Contoh struktur hirarki *class specification* pada domain *smartphone* ditunjukkan pada Gambar 3.2.

3.2.3 Hirarki *FuncReq*

Hirarki *class FuncReq* merupakan sebuah hirarki *class* yang merepresentasikan kebutuhan fungsional dalam domain tertentu. Melalui hirarki *FuncReq*, pertanyaan dan penjelasan dibangkitkan. Kedalaman maksimum dari hirarki *class FuncReq* tergantung pada hasil akuisisi pengetahuan dalam suatu domain. Contoh struktur hirarki *FuncReq* untuk domain *smartphone* disajikan pada Gambar 3.3. Bagaimana menyusun hirarki *FuncReq* diatur dalam Aksioma 3.2 dan 3.3.

Aksioma 3.2. *Internal nodes* atau *class* dalam hirarki *FuncReq* merupakan kebutuhan fungsional yang masih dapat dipecah lagi menjadi beberapa kebutuhan fungsional yang lebih spesifik, dan daun atau individu merupakan kebutuhan fungsional yang paling spesifik, yang tidak dapat dipecah lagi menjadi beberapa kebutuhan fungsional yang lebih spesifik. Susunan seperti ini bermanfaat dalam mekanisme *query refinement*, suatu mekanisme untuk lebih men-spesifik-kan kebutuhan pada saat percakapan.



Gambar 3.3 Contoh hirarki *FuncReq* dalam domain *smartphone*.

Definisi 3.9 (Fungsi *specF*). Fungsi *specF* menghasilkan himpunan spesifikasi yang mendukung sebuah kebutuhan fungsional. Fungsi ini memetakan himpunan individu dalam hirarki hierarchy *FuncReq* ke himpunan hirarki *Specification* sebagai berikut,

$$specF : \mathcal{I}_{func} \rightarrow \mathcal{I}_{spec}$$

Misal untuk sebuah kebutuhan fungsional $f \in \mathcal{I}_{func}$, himpunan spesifikasi yang mendukung f didapatkan dengan perumusan sebagai berikut,

$$specF(f) = \{s \in \mathcal{I}_{spec} | suppBy(f, s)\} \qquad (3.10)$$

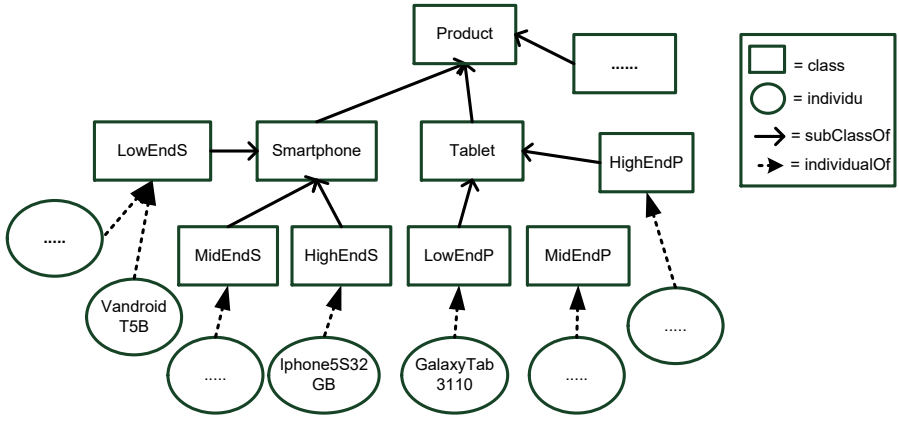
Aksioma 3.3. Misal $\mathcal{I}_{func}$ adalah himpunan individu hirarki *FuncReq*, dan $\mathcal{I}_{spec}$ adalah himpunan individu hirarki *Specification*. Untuk dua individu *FuncReq* yang berbeda, pasti terdapat minimal satu individu *Specification* pendukung yang berbeda. Secara formal dapat dituliskan,

Misal $f_1, f_2 \in \mathrm{I}_{func}$, sedemikain hingga,

$$f_1 \neq f_2 \quad jhj \quad \exists a_1 \in specF(f_1), a_2 \in specF(f_2)[a_1 \neq a_2] \qquad (3.11)$$

3.2.4 Hirarki *Class Product*

Hirarki dari *class Product* merepresentasikan hirarki dari jenis-jenis produk dalam suatu domain. Penyusunan hirarki jenis produk bisa berbeda, tergantung domain. Aturan dalam penyusunan *internal nodes* dan daun dalam hirarki *product*, disajikan dalam Aksioma 3.4. Sementara itu Definisi 3.10 mendefinisikan fungsi untuk mendapatkan spesifikasi-spesifikasi yang dipunyai oleh sebuah unit produk.



Gambar 3.4 Contoh struktur hirarki *Product*

Aksioma 3.4. *Internal nodes* dalam hirarki *Product* adalah *class* yang mendefinisikan jenis produk, yang diklasifikasikan dari jenis produk yang

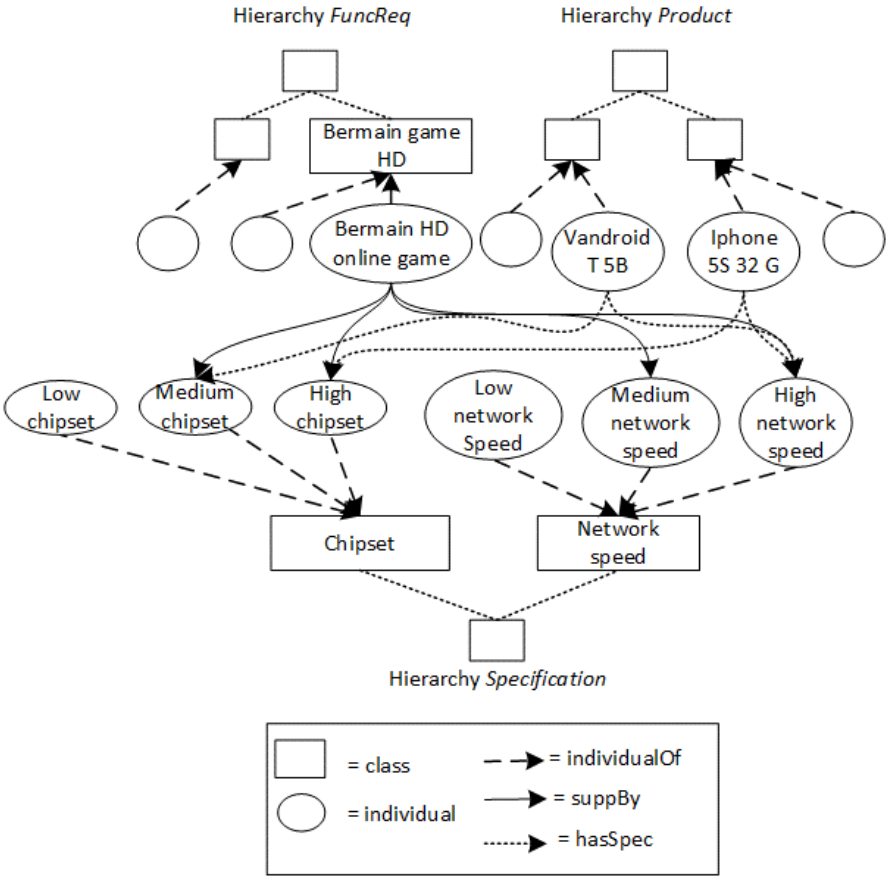
paling umum sampai yang paling spesifik. Sementara itu daun adalah individu yang merepresentasikan unit produk. Dalam diskusi selanjutnya, himpunan *class* dalam hirarki *Product* dinyatakan dengan C_{prod}, sementara himpunan individu dinyatakan dengan $\mathcal{I}_{prod}$.

Definisi 3.10 (Fungsi *specP*). Fungsi *SpecP* merupakan fungsi untuk mendapatkan himpunan spesifikasi yang dimiliki oleh sebuah produk *p*. Fungsi ini memetakan himpunan individu pada hirarki *Product*($\mathcal{I}_{prod}$) ke himpunan individu hirarki *Specification*($\mathcal{I}_{spec}$), sebagai berikut,

$$specP : \mathcal{I}_{prod} \rightarrow \mathcal{I}_{spec}$$

Untuk sebuah produk $p \in \mathcal{I}_{prod}$, himpunan spesifikasi yang dimiliki oleh *p* didapatkan dengan cara sebagai berikut,

$$specP(p) = \left\{ s \in \mathcal{I}_{spec} \middle| hasSpec(p, s), p \in \mathcal{I}_{prod} \right\} \qquad (3.12)$$



Gambar 3.5 Contoh potongan ontologi

Setiap individu hirarki *Product* (unit produk) mempunyai *property*: *idProduct*, *brand*, *max-harga* dan *detailSpec*. Sebagai contoh, sebuah individu product mempunyai *idProduct*: **Iphone4 8GB**, *Brand*: **Apple**, *Price*: **Rp 3.500.000**, dan *DetailSpec*: **RAM 512**, **Processor A4 1Ghz**, **Secondary Camera Resolution 5 Mpix**, dsb. *detailSpec* berbeda dengan individu dari hirarki *Specification*. Individu pada hirarki *Specification* merepresentasikan gradasi dari tingkatan kualitas spesifikasi secara ordinal. Contoh struktur hirarki *Product* dalam domain *smartphone* disajikan dalam Gambar 3.4. Sementara itu gambar 3.5 menunjukkan contoh potongan ontologi. Dalam gambar ini, dapat dilihat bahwa kebutuhan fungsional **Bermain HD Online Game** didukung oleh kualitas dari 2 jenis spesifikasi: **chipset** dan **network speed**. Tingkatan kualitas dari chipset

yang mendukung kebutuhan fungsional ini adalah medium chipset atau high chipset, sedangkan tingkatan kualitas dari network speed yang mendukung kebutuhan adalah medium **network speed** atau **high network speed**. Sementara itu, produk **IPhone 5s 32GB** mempunyai spesifikasi : **medium chipset, high network speed**.

3.3 Definisi Formal dari Ontologi

Dalam hal ini, definisi ontologi menggunakan pendekatan definisi bahasa formal (Ibanez dkk., 2009; Nguyen, 2008; Aubrech dkk., 2005). Berdasarkan pembahasan tentang struktur ontologi yang telah dijelaskan sebelumnya, definisi dari model ontologi yang diajukan, disajikan pada Definisi 3.11. Definisi formal ini sekaligus mengemas semua unsur-unsur dalam ontologi, sehingga dapat dijadikan pedoman untuk penerapan ontologi pada domain lain, khususnya untuk domain produk yang bersifat multi fungsi dan punya banyak fitur.

Definisi 3.11 (Definisi Formal Ontologi). Ontologi untuk *Conversational Recommender System* berbasis kebutuhan fungsional produk didefinisikan secara formal dalam sebuah 5-*tuple O*,

$$O = \langle \mathcal{N}, \mathcal{N}_D, \mathcal{R}_O, \mathcal{R}_H, OP \rangle$$

dengan $\mathcal{N}$ adalah himpunan *node*, terdiri dari *class* dan individu pada masing-masing hirarki, $\mathcal{N}_D$ adalah himpunan batasan untuk $\mathcal{N}$. $\mathcal{R}_O$ adalah himpunan relasi antar hirarki, $\mathcal{R}_H$ adalah himpunan relasi dalam hirarki dan OP adalah himpunan fungsi yang berlaku dalam ontologi.

$$\mathcal{N} = \{\mathcal{N}_{func}, \mathcal{N}_{spec}, \mathcal{N}_{prod}, C_{func}, C_{spec}, C_{prod}, \mathcal{I}_{func}, \mathcal{I}_{spec}, \mathcal{I}_{prod}\}$$

$$\mathcal{N}_{\mathcal{D}} = \{\mathcal{N}_{func} = C_{func} \cup \mathcal{I}_{func}, \mathcal{N}_{spec} = C_{spec} \cup \mathcal{I}_{spec}, \mathcal{N}_{prod} = C_{prod} \cup \mathcal{I}_{prod}\}$$

$$\mathrm{R}_O = \{suppBy, hasSpec\}$$

$$\mathrm{R}_H = \{individualOf, subClassOf, hasChild\}$$

$$\mathrm{OP} = \{specF, specP, children, parents, allInd\}$$

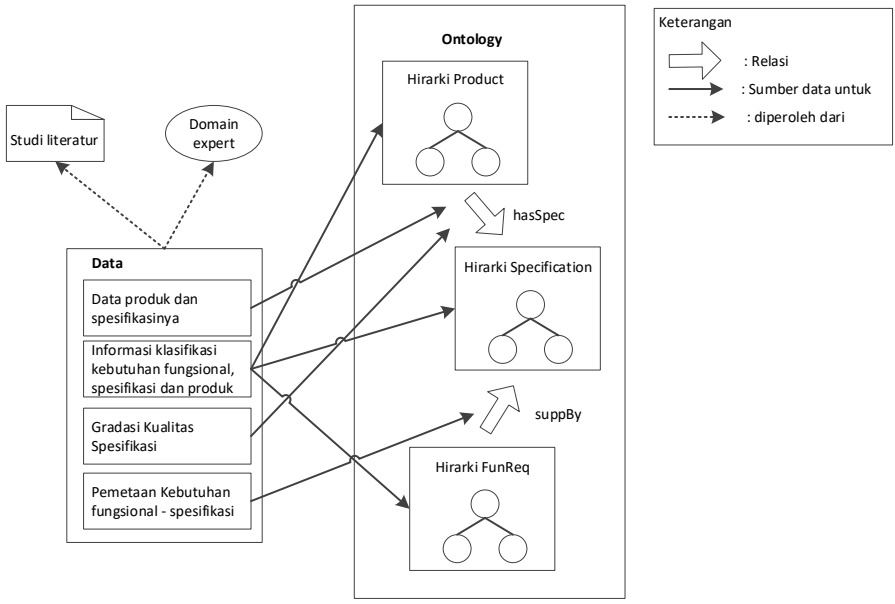
Dengan $\mathcal{N}_{func}$ adalah himpunan *node* dalam hirarki *FuncReq*, $\mathcal{N}_{spec}$ adalah himpunan *node* dalam hirarki *Specification*, $\mathcal{N}_{prod}$ adalah himpunan *node* dalam hirarki *Product*, C_{func} adalah himpunan *class* dalam hirarki *FuncReq*,

C_{spec} adalah himpunan *class* dalam hirarki *Specification*, C_{prod} adalah himpunan *class* dalam hirarki *Product*, I_{func} adalah himpunan individu dalam hirarki *FuncReq*, I_{spec} adalah himpunan individu dalam hirarki *Specification*dan I_{prod} adalah himpunan *class* dalam hirarki *Product*.

3.4 Pembangunan *Instance* dari Ontologi

Berdasarkan perumusan mekanisme interaksi yang telah dihasilkan, langkah selanjutnya adalah membangun *instance* dari ontologi, sebagai basis pengetahuan. Karena CRS yang dibangun merupakan salah satu bentuk dari *knowledge-based recommender system*, maka basis pengetahuan ini memegang peranan dalam membangkitkan interaksi. Dalam pengembangan CRS kali ini, dipilih smartphone sebagai domain dari basis pengetahuan, karena produk ini merupakan salah satu produk yang punya banyak kebutuhan fungsional, dan mempunyai banyak fitur teknis, dan telah menjadi kebutuhan utama pada masyarakat urban.

Untuk membangun *ontology instance* dalam domain *smartphone*, perlu dilakukan akuisisi pengetahuan tentang berbagai hal yang terkait dengan produk *smartphone*, meliputi : 1) Data produk dan spesifikasinya, 2) penentuan gradasi kualitas untuk masing-masing jenis spesifikasi, 3) kebutuhan-kebutuhan fungsional yang ada dalam domain *smartphone*, dan 4) pemetaan antara kebutuhan fungsional dengan jenis spesifikasi. Akuisisi pengetahuan terkait dengan domain produk didapatkan melalui studi literatur dari media *online* maupun *offline*, serta wawancara/konsultasi dengan *domain expert*.



Gambar 3.6. Sumber data untuk pembangunan ontologi

Data yang terkait dengan kebutuhan fungsional, dilakukan pengamatan pada forum diskusi pada www.kaskus.co.id, review produk smartphone dari www.amazon.com, serta www.gsmarena.com. Data produk dari media lokal diperlukan untuk mengetahui produk-produk yang dipasarkan saat ini di Indonesia serta perkembangan harga. Fitur teknis dari produk diperoleh dari www.gsmarena.com. Situs ini merupakan situs yang sering menjadi referensi untuk mengetahui spesifikasi produk-produk *smartphone*. Penentuan kualitas untuk beberapa spesifikasi khusus, mengacu pada *benchmark* yang dilakukan dalam situs www.gsmarena.com (*benchmark* untuk ketahanan baterai) dan www.primatelabs.com (*benchmark* untuk kinerja processor). Proses penentuan gradasi kualitas untuk kedua spesifikasi ini diilustrasikan pada Lampiran A.

Wawancara dengan *domain expert* perlu dilakukan untuk verifikasi tentang data kebutuhan fungsional yang telah dikumpulkan dari pengamatan di berbagai media. Selain itu, data tentang penentuan gradasi kualitas untuk masing-masing jenis spesifikasi dan pemetaan antara kebutuhan fungsional dengan jenis spesifikasi, murni berasal dari *domain expert*.

Tabel 3.1 Contoh penentuan gradasi kualitas untuk masing-masing jenis spesifikasi

Tingkatan Spesifikasi (individu dari hirarki *Specification*)	FiturTeknis
High RAM	**Android :** RAM > 2 GB RAM
	Windows Phone, Blackberry&Blackberry 10 : RAM = 1 GB
	iPhone : RAM = 1 GB
Medium RAM	**Android :** 1GB ≤ RAM ≤ 2 GB RAM
	iPhone, Windows Phone, Blackberry & Blackberry 10 : RAM = 512
Low RAM	**Android :** RAM≤ 512 MB
High Internal Memory	>= 32 GB
Medium High Internal Memory	16 GB ≤ Internal Memory < 32 GB
Medium Low Internal Memory	8 ≤ Internal Memory < 16 GB
Low Internal Memory	4GB <= Internal memory < 8GB
…….	…….

Gambar 3.6 menunjukkan beberapa data, dalam kaitannya dengan pembangunan ontologi. Data klasifikasi kebutuhan fungsional, spesifikasi dan produk merupakan sumber data dalam menyusun masing-masing hirarki *FuncReq*, *Specification* dan *Product*. Gradasi kualitas dan informasi spesifikasi produk menjadi sumber data untuk membentuk relasi *hasSpec*, sebuah relasi yang menghubungkan individu pada hirarki *Produk* dengan individu hirarki *Specification*.

Contoh dari penentuan gradasi kualitas dapat ditunjukkan pada Tabel 3.2. Tingkatan spesifikasi nantinya merupakan individu dari hirarki *Specification*. Sementara itu Tabel 3.3 menunjukkan contoh pemetaan antara kebutuhan fungsional dengan spesifikasi. Kedua tabel ini hanya menunjukkan sebagian hasil akuisisi pengetahuan, sedangkan hasil selengkapnya dapat dilihat pada Lampiran A. Pemetaan antara kebutuhan fungsional dan spesifikasi merupakan dasar untuk membuat relasi *suppBy*,

sebuah relasi yang menghubungkan individu-individu hirarki *FuncReq* dan individu-individu hirarki *Specification* (Gambar 3.2)

Tabel 3.2 Contoh pemetaan antara kebutuhan fungsional dan spesifikasi

Individu Kebutuhan Fungsional	Tingkatan dari Spesifikasi Pendukung
Standard Offline Gaming	CHIPSET >= Low CHIPSET
	RAM >= Medium Low RAM
	Screen Technology >= Medium
	Internal Memory >= Medium Low Internal Memory
	Battery Endurance >= Medium Battery
	Resolution >= Low Resolution
	OS = Android, Windows Phone, iOS, BB 10
	Screen Size >= Low Screen Size
Standard Online Gaming	CHIPSET >= Low CHIPSET
	RAM >= Medium Low RAM
	Internal Memory >= Medium Low Internal Memory
	Battery Endurance >= Medium Battery
	Screen Size >= Low Screen Size
	Network Speed >= Medium Network Speed
	OS : Android, Windows Phone, iOS
	Screen Technology >= Medium
HD Offline Gaming	CHIPSET >= Medium CHIPSET
	Internal Memory = High Internal Memory
	Battery Endurance = High Battery
	Resolution = Medium
	OS = Android, Windows Phone, iOS
.....	.....

Dalam Tabel 3.2, tingkatan spesifikasi nantinya merupakan individu dari hirarki *Specification*. Ontologi dibangun dengan menggunakan PROTEGE 4.0, yang merupakan alat bantu yang secara umum digunakan para peneliti, dan sudah mendukung OWL (*ontology web language*).

3.5 *Resume*

Bab ini membahas mengenai ontologi dalam pembangunan CRS berbasis kebutuhan fungsional produk. Ontologi dibangun secara *generic*, sehingga

dapat menjadi pedoman dalam pengembangan ontologi dalam domain lain, khususnya produk yang bersifat multi fungsi dan punya banyak fitur. Selain itu, berdasarkan unsur-unsur dalam ontologi ini, dapat dikembangkan model pembangkitan interaksi pada CRS berbasis pada kebutuhan fungsional. Model untuk pembangkitan interaksi berdasarkan ontologi yang telah dibahas, akan dibahas pada Bab 5.

4

PEMODELAN KEBUTUHAN PENGGUNA

4.1 Pendahuluan

Salah satu *value* dari sebuah *conversational recommender system* adalah kemampuannya dalam melakukan interaksi atau dialog dengan pengguna secara eksplisit, dalam pencarian produk. Sistem mempunyai kemampuan belajar tentang kebutuhan pengguna berdasarkan interaksi yang berlangsung. Oleh karena itu, sistem memodelkan kebutuhan pengguna dari *feedback* pengguna yang didapatkan selama interaksi, dalam *user profile model*. Berdasarkan reasoning yang melibatkan *user profile model* dan ontologi, dihasilkan sebuah *personalized interaction* antara pengguna dan sistem. Dalam bab ini juga dibahas mengenai perumusan secara umum dari mekanisme interaksi dalam CRS berbasis kebutuhan fungsional produk. Mekanisme interaksi ini merupakan gambaran dari *personalized interaction*, yang akan menjadi salah satu kunci dalam CRS yang dikembangkan.

4.2 *User Profile Model*

Sistem menggali kebutuhan pengguna secara eksplisit melalui pemberian pertanyaan, rekomendasi produk dan penjelasan mengapa suatu produk direkomendasikan. Sistem belajar tentang kebutuhan pengguna berdasarkan feedback pengguna dari setiap tahapan interaksi. Dengan demikian, sistem menyimpan *log* kebutuhan pengguna dalam *user profile model*. Dengan memanfaatkan *user profile model* (sebagai fakta) dan ontologi (sebagai pengetahuan), sistem mampu menghasilkan *personalized dialogue* untuk pengguna.

Sebuah pertanyaan terdiri dari beberapa opsi kebutuhan fungsional, jenis produk dan properti produk. Pada setiap interaksi, sistem akan memilih beberapa *node* dalam ontologi sebagai pertanyaan, dan berdasarkan pada jawaban pengguna, sistem akan menyimpan *nodes* tersebut dalam *user profile model* sebagai kebutuhan pengguna. Kebutuhan pengguna didefinisikan sebagai sebuah 5-*tuple* R= <*Fm*, *Fo*, *Fx*, *Ptype*, *Pproperty*>, dengan *Fm* himpunan *mandatory functional requirements*, *Fo* adalah himpunan *optional functional requirements*, *Fx* adalah himpunan *not required functional requirements*, *PType* adalah himpunan jenis produk yang diinginkan dan *Pproperty* adalah himpunan *property*i produk yang dipilih. *Property*i produk adalah fitur-fitur produk yang banyak diketahui oleh pengguna, seperti *brand*, sistem operasi dan max-harga (dalam hal ini adalah *maksimum budget*).

Kebutuhan pengguna R berubah-ubah secara dinamis sepanjang interaksi berjalan, dan semuanya dikelola dalam *user profile model*. *User profile model* merepresentasikan *log* dari kebutuhan pengguna, sehingga proses *updating* dilakukan secara *incremental* selama interaksi. Dengan demikian, bersama dengan ontologi, *user profile model* memegang peranan penting untuk membangun interaksi serta merekomendasikan produk. *User profile model* merupakan subset dari hirarki *FuncReq* pada ontologi, dan mewarisi struktur yang ada dalam hirarki. Setiap *node* disimpan dalam *user profile model* berdasarkan level yang bersesuaian dalam hirarki *FuncReq*, seperti dapat dilihat dalam Gambar 4.1.

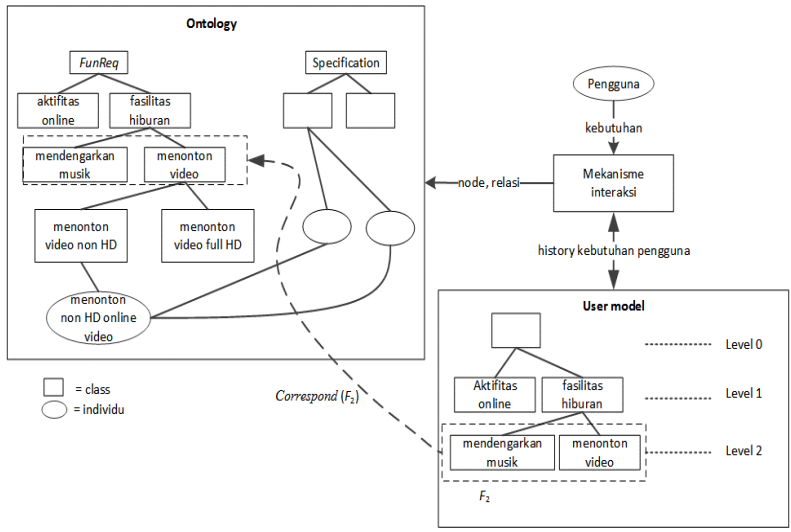
Dalam pembangkitan interaksi, sistem mengeksplorasi *nodes* dalam ontologi berdasarkan pada kondisi *user profile model* saat ini. Dengan demikian, pemetaan antara *user profile model* dan ontologi perlu dilakukan. Pemetaan

(korespondensi) ini dilakukan oleh function *correspond* yang dinyatakan dalam Definisi 4.1.

Definisi 4.1 (Fungsi *correspond*). Fungsi ini menghasilkan himpunan *node* pada hirarki *FuncReq* ($\mathcal{N}_{func}$) yang berkorespondensi dengan kebutuhan fungsional dalam *user profile model* pada level tertentu. Misal F_i adalah himpunan kebutuhan fungsional dalam *user profile model* pada level *i*. Fungsi ini dapat diformulasikan sebagai berikut,

$$correspond(F_i) = \bigcup_{f \in F_i} f' \in \mathrm{N}_{\mathrm{func}}, \textit{ that corresponds to } f \tag{4.1}$$

Gambar 4.1 memberikan gambaran tentang fungsi ini. Fungsi ini menghasilkan $\mathcal{N}_{func}$ yang berkorespondensi dengan himpunan *node* pada level 2 pada *user profile model*.



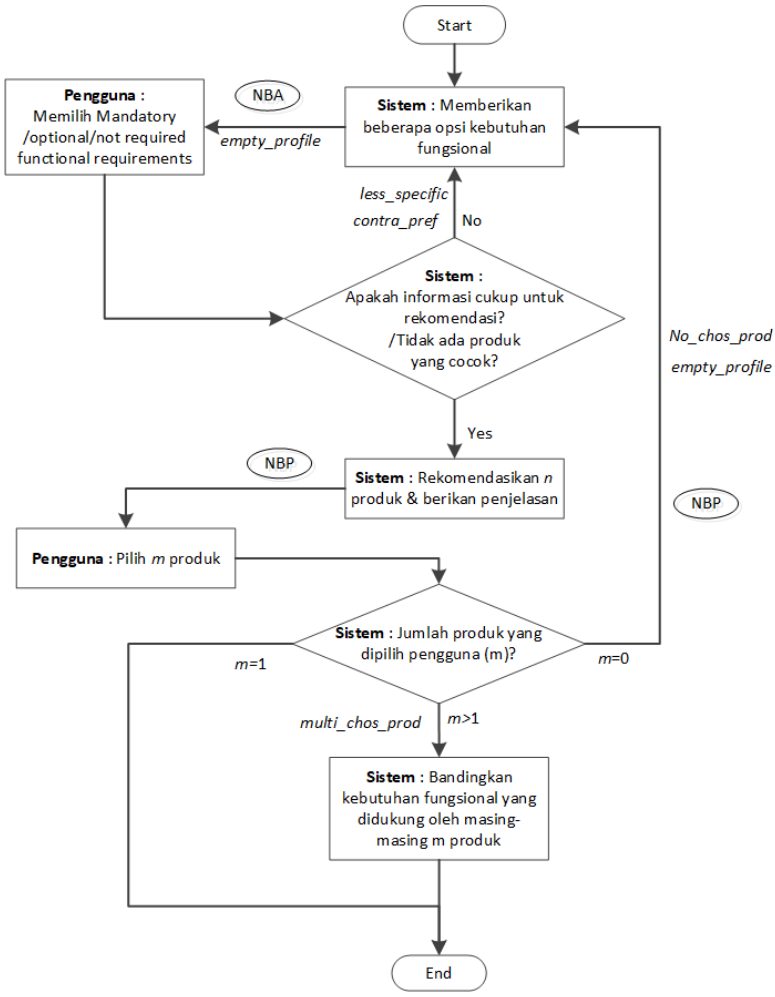
Gambar 4.1 *User profile model* dan fungsi *correspond*

4.3 Mekanisme Interaksi

Mekanisme interaksi meliputi dialog tanya jawab, pemberian sampel produk dan penjelasan mengapa sebuah produk direkomendasikan. Mekanisme interaksi ini memadukan strategy *navigation by asking* (NBA) dan

navigation by proposing (NBP). Dengan demikian, kebutuhan pengguna dapat digali, sehingga sistem dapat melakukan pemetaan antara kebutuhan fungsional dengan produk yang ada. Pengembangan mekanisme interaksi yang ada saat ini, lebih fokus menggali kebutuhan dari fitur produk secara langsung (Shimazu, 2001; Shimazu, 2002; Hu dan Aufaure, 2013). Sementara itu A*dvisor Suite* (Jannach, 2004; Felfernig dkk., 2006; Jannach dan Kreutler, 2004;2007), merupakan suatu CRS yang mempunyai suatu mekanisme interaksi untuk kebutuhan fungsional, namun sistem ini tidak ada menangani kritik pengguna untuk produk yang direkomendasikan (strategi NBA saja), sehingga kurang mewakili suatu interaksi antara calon pembeli dengan *professional sales support*.

Mekanisme interaksi melakukan penggalian kebutuhan berangkat dari kebutuhan fungsional dari level paling umum ke yang lebih spesifik (mekanisme *query refinement*). Dibandingkan penggalian kebutuhan yang langsung mengacu pada fitur produk maupun *Advisor Suite*, mekanisme interaksi yang akan dikembangkan lebih kompleks, namun lebih natural layaknya percakapan antara calon pembeli dengan *professional sales support*.



Gambar 4.2 Skema interaksi antara pengguna dan sistem

Gambar 4.2 merangkum proses interaksi antar pengguna dengan sistem. Pada awal interaksi, sistem memberikan beberapa pilihan kebutuhan fungsional yang diinginkan, kemudian pengguna memilih satu atau beberapa pilihan tersebut. Kemudian pengguna dapat memilih beberapa kebutuhan fungsional tersebut sebagai bagian dari himpunan *Fm*, *Fo* atau *Fx*. Jika kebutuhan pengguna dipandang cukup untuk rekomendasi produk, maka sistem akan merekomendasikan beberapa produk serta memberi penjelasan mengapa masing-masing produk direkomendasikan. Jika

informasi dipandang belum cukup, maka sistem akan memberikan pertanyaan pada tingkatan kebutuhan fungsional yang lebih spesifik.

Pada tahap selanjutnya, pengguna diberikan kesempatan untuk memilih satu atau lebih produk yang direkomendasikan, namun ada pilihan untuk menolak semua produk yang direkomendasikan. Jika memilih salah satu produk, maka dianggap pengguna telah menetapkan pilihan dan interaksi selesai. Namun jika tidak ada produk yang dipilih, dianggap pengguna belum puas dengan rekomendasi yang diberikan, sehingga sistem akan memberikan alternatif pertanyaan lain. Jika pengguna memilih lebih dari satu produk, mengindikasikan bahwa pengguna masih ragu memutuskan antara beberapa produk yang dipilih tersebut. Dengan demikian sistem akan membandingkan beberapa produk tersebut, terkait dengan kebutuhan fungsional yang didukung oleh masing-masing produk, sehingga memudahkan pengguna untuk mengambil keputusan.

4.4 Perumusan Kasus-kasus *Feedback* (Kasus Interaksi) Pengguna

Pada subbab sebelumnya dibahas tentang skema alur interaksi antara pengguna dan sistem. Berdasarkan skema tersebut, dapat dirumuskan beberapa kasus *feedback* pengguna, untuk menghasilkan suatu p*ersonalized interaction* (interaksi personal). *Feedback* dapat berupa jawaban atas pertanyaan yang diajukan sistem, atau bisa juga respon pengguna terhadap produk-produk yang direkomendasikan. Personalisasi interaksi ini dapat dibangun berkat peran dari *user profile model*. Sistem membangkitkan interaksi personal berdasarkan 5 kasus *feedback* pengguna, yaitu :

1. User profile model dalam keadaaan kosong (*empty_profile*)
   User profile model dalam keadaan kosong dapat terjadi pada awal interaksi, maupun di tengah interaksi. Kasus ini terjadi di tengah interaksi, jika pengguna ingin mengubah preferensi kebutuhan dari awal, setelah berinteraksi dengan sistem. Jika kasus ini terjadi, maka sistem akan menanyakan kebutuhan fungsional yang masih bersifat umum. Kebutuhan yang bersifat umum ini terdapat pada level awal dari hirarki *FuncReq* dalam ontologi. Jawaban dari pengguna akan ditambahkan pada *user profile model.*
2. Pengguna memilih lebih dari satu produk yang direkomendasikan (*multi_chos_prod*)

Setelah sistem memberikan beberapa sampel produk untuk direkomendasikan, terdapat kemungkinan bahwa pengguna ragu untuk memilih antara beberapa produk yang direkomendasikan. Pengguna diperbolehkan memilih sejumlah *k* produk, dan sistem akan memberikan panduan bagi pengguna untuk mengambil keputusan dengan cara menampilkan perbandingan kebutuhan fungsional pengguna yang didukung oleh masing-masing produk, tetapi tidak didukung oleh produk yang lain. Sistem akan melakukan penelusuran pada *user profile model* untuk mendapatkan kebutuhan fungsional pengguna. Kemudian dengan memanfaatkan fungsi *correspond*, sistem dapat menelusuri ontologi untuk mendapatkan *nodes* kebutuhan fungsional yang didukung oleh produk yang terkait.

3. Pengguna tidak memilih satupun produk yang direkomendasikan (*no_chos_prod*)
   Adakalanya, dari sekian produk yang direkomendasikan sistem, tidak ada satupun yang disukai oleh pengguna. Oleh karena itu, sistem akan mengajukan pertanyaan kebutuhan fungsional yang potensial untuk disukai oleh pengguna. Sistem akan menelusuri *unexplored functional requirements* dalam ontologi, berdasarkan informasi kebutuhan pengguna dalam *user profile model* saat ini.
4. Informasi tentang kebutuhan pengguna belum cukup untuk rekomendasi produk (*less_specific*)
   Keadaan ini terjadi saat kebutuhan pengguna dalam *user profile model* masih terlalu umum, dan menghasilkan terlalu banyak produk yang direkomendasikan. Dalam kasus ini, sistem akan memberikan pertanyaan kebutuhan fungsional yang lebih spesifik, dengan menelusuri *nodes* pada hirarki *FuncReq* yang lebih bawah.
5. Tidak ada produk yang sesuai dengan kebutuhan pengguna (*contra_pref*)
   Hal ini terjadi jika terdapat kontradiksi antara kebutuhan-kebutuhan pengguna dalam *user profile model*, sehingga sistem akan menandai kebutuhan-kebutuhan yang saling kontradiktif tersebut. Kemudian sistem akan memberikan pertanyaan-pertanyaan kembali, dan memberikan pengguna kesempatan untuk merevisi kebutuhan-kebutuhan yang kontradiktif tadi.

Masing-masing kasus interaksi ini juga dapat digambarkan dalam mekanisme interaksi, seperti ditunjukkan pada Gambar 4.2.

4.5 *Resume*

Bab ini membahas tentang bagaimana sistem belajar tentang kebutuhan pengguna, untuk dapat menciptakan interaksi personal. Sistem dapat belajar, jika sistem mampu memodelkan kebutuhan pengguna, berdasarkan feedback pengguna, melalui sebuah *temporary knowledge*. *Temporary knowledge* ini yang disebut sebagai *user profile model*. Bab ini membahas mengenai *user profile model*, serta mekanisme interaksi secara umum. Selain itu, dibahas juga perumusan kasus-kasus *feedback* pengguna yang mungkin terjadi saat interaksi berlangsung, beserta strategi penanganan masing-masing kasus secara umum. Strategi penanganan masing-masing kasus *feedback* pengguna ini akan dibahas pada Bab 4, yang direalisasikan dalam bentuk model komputasional pembangkitan pertanyaan.

5

MODEL KOMPUTASIONAL UNTUK REKOMENDASI PRODUK

5.1 Pendahuluan

Tugas utama dari sebuah *recommender system* adalah merekomendasikan produk, yang sesuai dengan kebutuhan pengguna. Sebagai sebuah *knowledge-based recommender system*, kebutuhan pengguna ini dinyatakan secara eksplisit oleh pengguna. Selama interaksi berjalan, kebutuhan pengguna (berupa kebutuhan fungsional) ini disimpan dalam *user profile model*. Dalam bab ini akan dibahas mengenai model komputasional dalam rekomendasi produk, yang disajikan dalam bentuk definisi fungsi – fungsi dasar, aksioma dan algoritma.

5.2 Fungsi-fungsi Dasar dan Algoritma Rekomendasi Produk

Sistem melakukan pencarian produk yang memenuhi *Fm*, *Fo*, *Ptype*, *Pprop* dalam *user profile model*. *Fm* dan *Fo* merepresentasikan kebutuhan fungsional yang diinginkan pengguna, yang didapatkan selama interaksi. Permasalahan utama dalam rekomendasi produk adalah menentukan apakah suatu individu produk memenuhi suatu individu kebutuhan fungsional tertentu, sebagai *constraint* yang harus dipenuhi. Masalah ini terkait dengan jenis spesifikasi yang mendukung kebutuhan fungsional tersebut maupun jenis spesifikasi yang dimiliki oleh produk. Seperti telah dijelaskan pada Subbab 3.2.2, individu dari hirarki *Specification* adalah tingkatan kualitas dari spesifikasi (misal low RAM, middle RAM, high RAM, dsb). *Parent* dari individu tersebut disebut sebagai jenis spesifikasi (misal RAM, *processor*, *primary camera resolution*, dsbdan sebagainya). Beberapa definisi yang terkait dengan rekomendasi produk disajikan dalam Definisi 5.1 sampai Definisi 5.4.

Definisi 5.1 (Fungsi *isSatIndF*). Fungsi *isSatIndF* digunakan untuk memeriksa apakah suatu produk memenuhi sebuah individu dari kebutuhan fungsional. Fungsi ini memetakan himpunan individu dalam hirarki *Produk* dan himpunan individu dalam hirarki *FuncReq* ke *Boolean* sebagai berikut,

$$isSatIndF : \mathcal{I}_{prod} \times \mathcal{I}_{func} \rightarrow Boolean$$

Dengan $\mathcal{I}_{prod}$ adalah himpunan individu dalam hirarki *Product* dan $\mathcal{I}_{func}$ adalah himpunan individu dalam hirarki *FuncReq*. Jika sebuah produk memenuhi sebuah individu kebutuhan fungsional, maka *isSatIndF* menghasilkan nilai *true*, dan *false* sebaliknya. Product $p \in \mathrm{I}_{prod}$ dikatakan memenuhi sebuah individu kebutuhan fungsional $f \in \mathrm{I}_{func}$, jika dan hanya jika *p* mempunyai semua jenis spesifikasi yang mendukung *f*. Seperti telah dijelaskan sebelumnya, jenis spesifikasi didapatkan dengan mengakses *parent* dari individual dari *Specification* (level 1 dari hirarki *Specification*). Karena $specF(f)$ menyatakan himpunan individu spesifikasi (I_{spec}) yang mendukung kebutuhan fungsional *f*, maka himpunan jenis spesifikasi yang mendukung *f* didapatkan melalui $parents(specF(f))$. Dengan cara yang sama, $parents(specF(f) \cap specP(p))$ menghasilkan himpunan jenis spesifikasi dari: spesifikasi yang dipunyai produk *p* yang juga sekaligus mendukung

kebutuhan fungsional *f*. Dengan demikian, fungsi *isSatIndF* didefinisikan sebagai berikut,

$$isSatIndF(p,f)=\begin{cases} true, & parents(specF(f))= \\ & parents(specF(f)\cap specP(p)) \\ false, & otherwise \end{cases} \quad (5.1)$$

Dalam diskusi selanjutnya, istilah "produk" akan selalu mengacu pada individu pada hirarki *Product*. Contoh V.1 dapat memperjelas definisi fungsi ini.

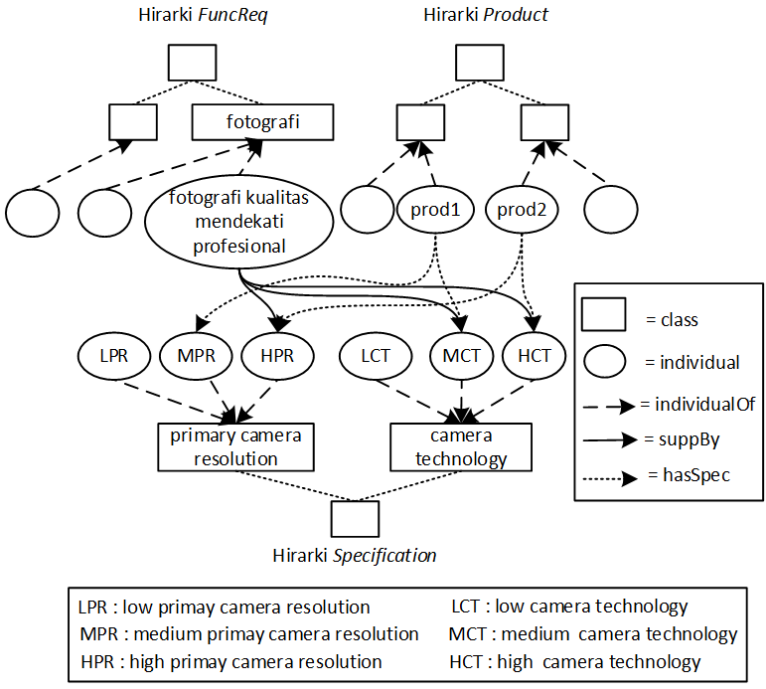
Contoh 5.1. Gambar 5.1 menunjukkan contoh hubungan dari beberapa individu dalam ontologi. Sebagai contoh, ingin diuji apakah *prod1*, *prod2* memenuhi kebutuhan fungsional *f* = *fotografi kualitas mendekati profesional*. Dalam Gambar 5.1, parent dari individu-individu LPR, MPR dan HPR adalah *primary camera resolution* (jenis spesifikasi). Sementara itu, parent dari individu-individu LCT, MCT dan HCT adalah *camera technology*. Dengan demikian didapatkan,

1. **Spesifikasi-spesifikasi yang mendukung kebutuhan fungsional *f***

   SpecF(*f*) = {HPR, MCT, HCT}.

2. **Jenis – jenis spesifikasi yang mendukung kebutuhan fungsional *f***

   parents(*specF*(*f*)) = {*primary camera resolution*, *camera technology*}



Gambar 5.1 Contoh potongan ontologi untuk kasus rekomendasi

3. **Tingkatan-tingkatan kualitas spesifikasi yang dimiliki oleh *prod1* dan *prod2***

   $specP(prod1) = \{MPR, MCT\}$, $specP(prod2) = \{HPR, HCT\}$

4. **Tingkatan-tingkatan kualitas spesifikasi yang mendukung *f*, dan dipunyai oleh masing-masing produk**

   $specF(f) \cap specP(prod1) = \{MCT\}$
   $specF(f) \cap specP(prod2) = \{HPR, HCT\}$

5. **Jenis-jenis spesifikasi yang mendukung f dan dimiliki oleh masing-masing produk**

   $parents(specF(f) \cap specP(prod1)) = \{camera\ technology\}$
   $parents(specF(f) \cap specP(prod2)) = \{primary\ camera\ resolution, camera\ technology\}$

Dengan mengacu pada definisi dari fungsi *isSatIndF* dan Persamaan 5.1, didapatkan,

$$parents(specF(f)) \neq parents(specF(f) \cap specP(prod1))$$
$$parents(specF(f)) = parents(specF(f) \cap specP(prod2))$$

Dengan demikian dapat disimpulkan bahwa *prod2* memenuhi *f*, sementara itu *prod1* tidak memenuhi *f*.

Pada suatu tahapan interaksi tertentu, himpunan kebutuhan fungsional yang dipilih pengguna, dapat berupa himpunan *class* dalam hirarki *FuncReq* (C_{func}) atau himpunan individu dalam hirarki *FuncReq* (I_{func}). Jika suatu kebutuhan fungsional berupa I_{func}, produk *p* yang direkomendasikan dapat didapatkan melalui fungsi *isSatIndF* secara langsung. Tetapi jika kebutuhan fungsional berupa C_{func}, maka perlu diuji apakah suatu produk *p* memenuhi suatu C_{func}. Perumusan untuk hal ini disajikan dalam Definisi 5.2.

Definisi 5.2 (Fungsi *isSatClassF*). Fungsi *isSatClassF* digunakan untuk memeriksa apakah sebuah produk memenuhi sebuah *class* kebutuhan fungsional tertentu. Fungsi ini memetakan himpunan individu pada hirarki *Product* (I_{prod}) dan himpunan *class* pada hirarki *FuncReq* (C_{func}) ke Boolean sebagai berikut,

$$isSatClassF : I_{prod} \times C_{func} \rightarrow Boolean$$

Jika sebuah produk memenuhi sebuah *class* kebutuhan fungsional, maka *isSatClassF* memberikan nilai **true**, dan **false** untuk sebaliknya. Untuk sebuah kebutuhan fungsional $f \in C_{func}$ dan sebuah produk $p \in I_{prod}$, *p* dikatakan memenuhi *f* jika dan hanya jika *p* memenuhi paling sedikit satu individu dari *f*. Dengan demikian fungsi *isSatClassF* dapat didefinisikan sebagai berikut,

$$isSatClassF(p, f) = \bigvee_{f_i \in allInd(f)} isSatIndF(p, f_i),\ i > 0, \quad (5.2)$$

Dari definisi *isSatIndF* dan *isSatClassF,* dapat diturunkan fungsi *satProd* yang disajikan dalam Definisi 5.3.

Definisi 5.3 (Fungsi *satProd*). Fungsi *satProd* menghasilkan himpunan produk yang memenuhi suatu himpunan kebutuhan fungsional (*F*). Fungsi ini memetakan himpunan *node* dalam hirarki *FuncReq* ($\mathcal{N}_{func}$) dan himpunan individu dalam hirarki *Product*, ke himpunan individu dalam hirarki *Product* sebagai berikut,

$$satProd : \mathcal{N}_{func} \times \mathcal{I}_{prod} \rightarrow \mathcal{I}_{prod}$$

Misal *P* adalah himpunan produk, dengan $P \subseteq \mathrm{I}_{prod}$, dan $F = \{f_i | f_i \in \mathcal{I}_{func}\} \cup \{f_c | f_c \in C_{func}\}$ adalah himpunan kebutuhan fungsional. Dengan demikian, produk-produk dalam *P* yang memenuhi *F* didapatkan dengan mengkombinasikan produk-produk yang memenuhi f_i dan f_c sebagai berikut, $satProd(P,F) = \{p \in P | satF(p,F)\}$ (5.3)

dengan,

$$satF(p,F) = \bigwedge_{f_i, f_c \in F} \begin{cases} isSatIndF(p, f_i), & f_i \in \mathcal{I}_{func} \\ isSatClassF(p, f_c) & f_c \in C_{func} \end{cases} \quad (5.4)$$

Aksioma 5.1 berikut menjelaskan hubungan antara struktur hirarki *FuncReq* dan Definisi 5.2.

Aksioma 5.1. Semakin dalam level *node* dalam hirarki *FuncReq*, merepresentasikan kebutuhan fungsional yang lebih spesifik. Misal *f* adalah sebuah *class* dalam hirarki *FuncReq*. Himpunan produk yang memenuhi *f* adalah sama dengan himpunan produk yang memenuhi paling sedikit satu anak dari *f*,

$$\{p \in I_{prod} | isSatClassF(p,f)\} = \left\{ p \in I_{prod} \middle| \bigvee_{g_i \in children(f)} isSatClassF(p, g_i) \right\}, \text{ for } i > 0 \quad (5.5)$$

Pada saat sistem merekomendasikan produk, sistem hanya menampilkan *k* produk dengan nilai *utility* tertinggi $(k \geq 1)$. Nilai *utility* dari masing-masing produk didapatkan dengan melalui fungsi *utility*, yang didefinisikan dalam Definisi 5.4.

recommend(*userModel*) → himpunan produk yang direkomendasikan
//Himpunan produk yang direkomendasikan = $\{p_1, p_2, ..., p_n\}$, $p_i \in \mathcal{I}_{prod}$. $1 < i < n$.

input : *userModel*, user profile model.
1. Dapatkan *Fm*, *Fo*, *Ptype*, *Pproperty* dari *userModel*
2. **Initialize** *Prod*← semua individu dalam hirarki *Product*
3. **if** *Pproperty*≠ϕ **then**
4. *Pprop* ← himpunan produk yang memenuhi *Pproperty*
5. **if** *Ptype* ≠ ϕ **then**
6. *Prod* ← *Pprop* ∩ ***allInd***(*Ptype)*
7. else
8. *Prod* ← *Pprop*
9. else
10. **if** *Ptype* ≠ ϕ **then**
11. *Prod* ← ***allInd***(*Ptype*)
12. **end if**
13. **if** *Fm* ≠ϕ **then**
14. *recProd* ←***satProd***(*Prod*, *Fm*) //*recProd* adalah himpunan produk yang memenuhi *Fm*
15. **else**
16. *recProd*←*Prod*
17. **end if**
18. **if** *Fo*≠ϕ, **then**
19. **for each** $p \in recProd$ **do**
20. Tentukan *utility*(*p*) menggunakan persamaaan (5.5)
21. **end for**
22. **end if**
23. Urutkan *recProd* berdasarkan nilai *utility*(*p*)
24. **return** *recProd*

Gambar 5.2 Algoritma untuk merekomendasikan produ

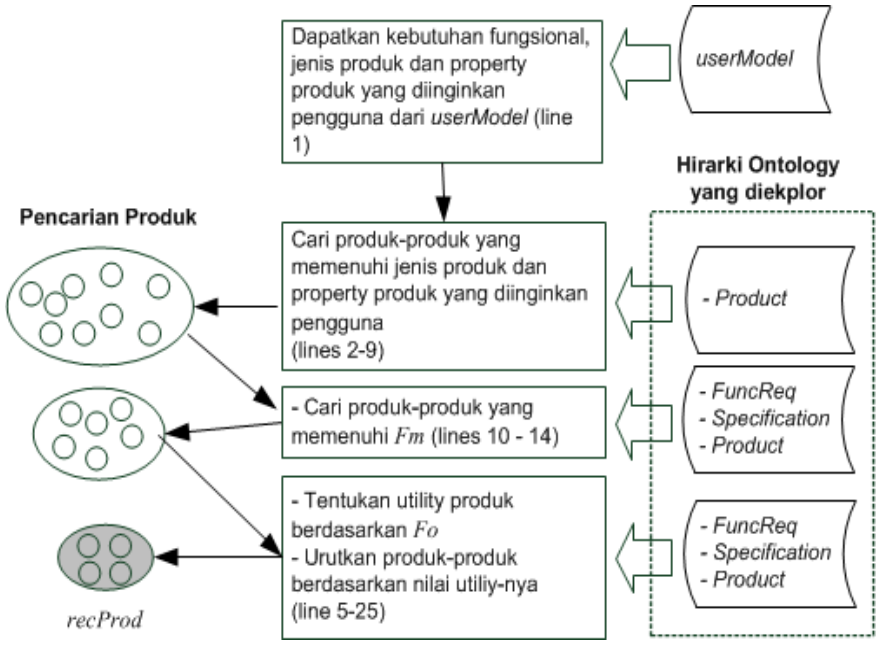
Definisi 5.4 (Fungsi *utility*). Fungsi *utility* adalah sebuah fungsi yang menghasilkan jumlah *optional functional requirement* yang dipenuhi oleh sebuah produk. Nilai *utility* diperlukan untuk meranking produk dalam sebuah proses rekomendasi. Fungsi ini memetakan himpunan individu dalam hirarki *Product* ke himpunan integer (*INT*).

$$utility : I_{prod} \rightarrow INT$$

Untuk sebuah produk $p \in I_{prod}$ dan himpunan *optional functional requirementFo*, utility dari produk *p* didefinisikan sebagai berikut,

$$utility(p) = \left| \{ fo \in allInd(Fo) | isSatIndF(p, fo) \} \right| \qquad (5.6)$$

Produk-produk yang direkomendasikan, diperoleh dengan algoritma *recommend* yang ditunjukkan pada Gambar 5.2. Secara garis besar, langkah-langkah dalam algoritma *recommend* diilustrasikan pada Gambar 5.3. Dalam Gambar 5.3 ditunjukkan bahwa, *mandatory functional requirements* (*Fm*), property produk (*Pproperty*) dan jenis produk (*Ptype*) merupakan constraint yang harus dipenuhi dalam merekomendasikan produk. Pencarian produk berdasarkan *Pproperty* dan *Ptype* melibatkan penelusuran relasi dalam hirarki *class Product*. Sedangkan pencarian produk yang memenuhi *Fm* melibatkan penelusuran relasi-relasi dan *nodes* dalam ketiga hirarki *class* ; *FuncReq*, *Specification*, dan *Product* (lihat contoh 5.2).



Gambar 5.1 Ilustrasi langkah-langkah pada algoritma *recommend*

Optional functional requirements (*Fo*) berperan dalam menentukan nilai *utility* setiap produk yang direkomendasikan. Penentuan nilai *utility* ini melibatkan fungsi *isSatIndF* (lihat contoh 5.1), sehingga inferensi melibatkan penelusuran *nodes* dan relasi-relasi pada ketiga hirarki *class*. Nilai *utility* ini merepresentasikan kelebihan dari setiap produk yang direkomendasikan, dengan melihat banyaknya kebutuhan fungsional

spesifik yang dipenuhi oleh masing-masing produk. Penjelasan detil tentang algoritma ***recommend***, disajikan melalui contoh 5.2.

Contoh 5.2. Pertama-tama, system mengambil nilai dari semua mandatory *functional requirements (Fm) and optional functional requirements (Fo) yang disimpan dalam user profile model (baris 1). Misal Fm : {Fotografi kualitas mendekati profesional (individu), Bermain game HD (class)}, Fo : {Mendengarkan musik}. Ptype: {smartphone}, PProperty : {rentang-harga: 2000.000 - 4.500.000 ; brand: samsung, lenovo; sistem operasi: {Android, Windows phone}.*

(i) Inisialisasi himpunan produk *Prod*
Inisialisasi himpunan produk (*Prod*), yaitu semua individu dalam hirarki *Product* (baris 2 dari algoritma)

(ii) Filtering produk berdasarkan *Pproperty* dan *Ptype*
Misal *Pproperty* dan *Ptype* $\neq \phi$. Dengan demikian, himpunan produk yang memenuhi *Pproperty* (*Pprop*), adalah semua produk yang memenuhi *Pproperty* (baris 4 dari algoritma), misal *Prop* = {*Samsung1, Samsung 2, Samsung 3, Lenovo1, Lenovo2*}. Selanjutnya, dicari himpunan produk yang memenuhi memenuhi *Pproperty* dan *Ptype* dengan *statement* pada baris 6 dari algoritma.

$$Prod = Pprop \cap \textbf{\textit{allInd}}(Ptype)$$

allInd(*Ptype*) adalah semua individu dari himpunan *Ptype*, sesuai dengan Persamaan III.10. Sementara itu, *Ptype* merupakan *class* dalam hirarki *Product*.

$$allInd(Ptype) = \bigcup_{a_i \in Ptype} \{x \in \mathcal{N} \mid individualOf(x, a_i)\}$$

Misal allInd(Ptype) = {Samsung1, samsung3, Lenovo1, Lenovo5, Lenovo6}, dengan demikian

Prod = {Samsung1, Samsung2, Samsung3, Lenovo1, Lenovo2} ∩
{Samsung1, samsung3, Lenovo1, Lenovo5, Lenovo6}
= {Samsung1, samsung3, Lenovo1}

(iii) Filtering produk perdasarkan *Fm*
Tahap selanjutnya adalah melakukan filtering, untuk mencari produk-produk dalam himpunan *Prod*, yang memenuhi *Fm* (baris 13-16 dari

algoritma). Dalam tahap ini, sistem menggunakan fungsi *satProd*(*Prod, Fm*), mengacu pada Persamaan 5.3.

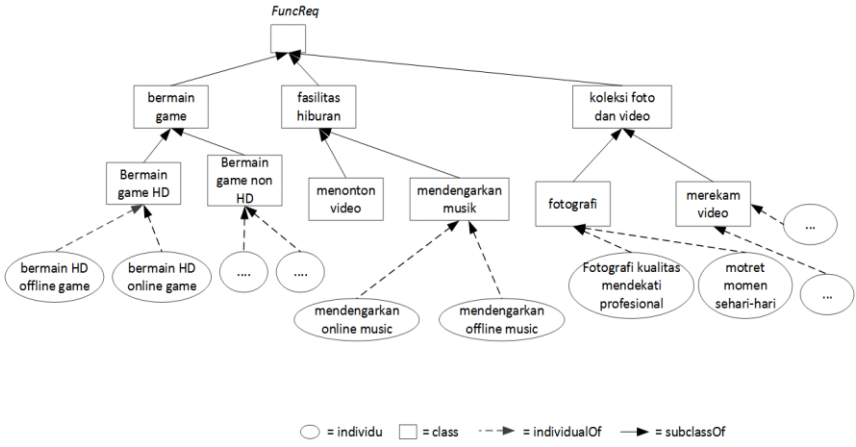
$$satProd(Prod, Fm) = \{p \in Prod | satF(p, Fm)\}, \text{ dengan}$$

$$satF(p, Fm) = \bigwedge_{fm_i, fm_C \in Fm} \begin{cases} isSatIndF(p, fm_i), & fm_i \in \mathcal{I}_{func} \\ isSatClassF(p, fm_c) & fm_c \in C_{func} \end{cases},$$

Jadi, untuk setiap $p \in Prod$, diperiksa apakah memenuhi *Fm* dengan persamaan *satF*(*p*, *Fm*). Dari contoh kasus ini diketahui, bahwa kebutuhan Fotografi kualitas mendekati profesional adalah individu, sehingga termasuk fm_i, dan Bermain game HD adalah *class*, sehingga termasuk fm_c. Sebagai contoh, misal untuk produk Samsung1,

satF(samsung1,Fm) = isSatIndF(Samsung1, Fotografi kualitas mendekati profesional) ∧ isSatClassF(Samsung1, Bermain game HD)

Untuk fungsi *isSatClass*, dapat dilihat pada Persamaan 5.2.



Gambar 5.2. Contoh potongan ontologi untuk hirarki *FuncReq*

Gambar 5.4 menunjukkan contoh potongan ontologi. Dari potongan ontologi ini, dapat dilihat bahwa allInd(bermain game HD) = {*bermain HD online game, bermain HD offline game*), sehingga,

$$isSatClassF(Samsung1, Bermain\ game\ HD)$$
$$= \bigvee_{f_i \in allInd(Bermain\ game\ HD)} isSatIndF(Samsung1, f_i)$$
$$= isSatIndF(Samsung1,\ bermain\ HD\ online\ game) \vee isSatIndF(Samsung1,\ bermain\ HD\ offline\ game)$$

Dengan demikian,
satF(samsung1, Fm) =
isSatIndF(Samsung1, fotografi kualitas mendekati profesional)
$\wedge$ *(isSatIndF(Samsung1, bermain HD online game)*
$\vee$ *isSatIndF(Samsung1, bermain HD offline game))*

Pemeriksaan nilai *isSatIndF* dilakukan dengan penelusuran relasi pada ontologi, seperti dapat dilihat pada contoh 5.1.
Misal,
satF(samsung1, *Fm*) = TRUE, *satF*(*samsung3*, *Fm*) = FALSE, dan *satF*(*Lenovo1*, *Fm*) = TRUE, maka produk yang direkomendasikan,
recProd = {*Samsung1, Lenovo1*}

(iv) Menghitung nilai *utility* untuk setiap produk yang direkomendasikan

Nilai *utility* dari masing-masing produk yang direkomendasikan (*recProd*) ditentukan dengan langkah-langkah pada baris 18 sampai 22. Produk-produk yang direkomendasikan adalah *n* produk dengan nilai *utility* tertinggi.

Dengan demikian, untuk setiap $p \in recProd$, ditentukan nilai *utility*(*p*). Sebagai contoh, untuk produk Samsung1,

$$utility(Samsung1) = \left| \{ fo \in allInd(\{mendengarkan\ musik\}) | isSatIndF(Samsung1, fo) \} \right|$$

Dari penelusuran relasi ontologi yang ditunjukkan pada Gambar V.4, diketahui,
allInd ({mendengarkan Musik}) = *{mendengarkan offline music, mendengarkan online music}*

sehingga, dilakukan pemeriksaan untuk setiap *fo* ∈ { *mendengarkan offline music*, *mendengarkan online music*}, apakah memang didukung oleh Samsung1.

Misal, isSatInd(Samsung1, mendengarkan offline music) = TRUE,
isSatInd(Samsung1, mendengarkan online music) = TRUE,
maka,

utility(Samsung1) = 2.

Cara yang sama dilakukan juga untuk produk Lenovo1. Proses mendapatkan nilai *isSatInd* dilakukan dengan penelusuran pada ontologi seperti pada contoh 5.2.

5.3 Kajian Kompeksitas dari Model Rekomendasi

Dari semua model pembangkitan interaksi, operasi yang dominan (membutuhkan *effort* terbesar) untuk komputasi adalah algoritma rekomendasi (algoritma *recommend*), khususnya pencarian produk yang memenuhi *Fm*, karena melibatkan banyak operasi pada *nodes* ontologi secara langsung. Sementara dalam pembangkitan pertanyaan, operasi lebih banyak pada nodes di *user profile model*, serta *nodes* ontologi yang berkorespondensi dengan node pada *user profile model* tersebut. Penentuan himpunan produk yang memenuhi himpunan kebutuhan fungsional *Fm*, mengacu pada *satProd* pada Persamaan 5.3,

$$satProd(P, Fm) = \{p \in P | satF(p, Fm)\}$$

dengan,

$$satF(p, Fm) = \bigwedge_{fm_i, fm_c \in Fm} \begin{cases} isSatIndF(p, fm_i), & fm_i \in I_{func} \\ isSatClassF(p, fm_c) & fm_c \in C_{func} \end{cases}$$

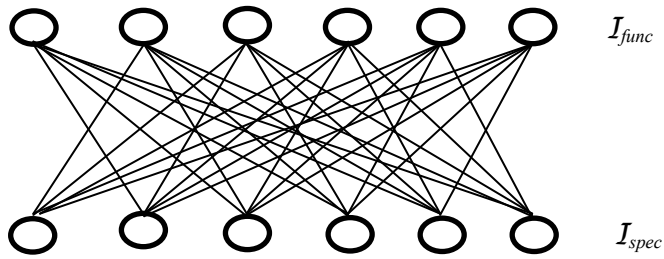
mengacu pada Persamaan 5.4.

Karena *isSatClassF*(*p*,*fm*$_c$) sebenarnya adalah *disjunction* dari *isSatIndF* (mengacu pada Persamaan 5.2),

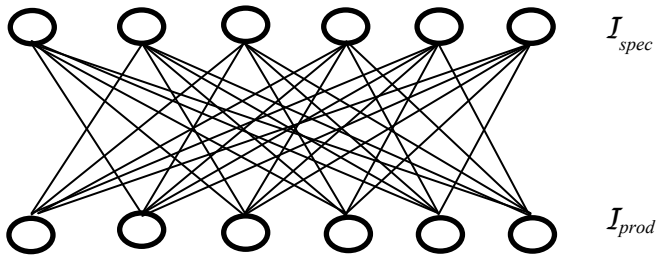
$$isSatClassF(p, fm) = \bigvee_{f_k \in allInd(fm)} isSatIndF(p, fm_k), \quad i > 0$$

Dengan demikian, pertama-tama, kompleksitas dapat ditinjau dari operasi *isSatIndF*(*p*,*fm*) untuk suatu $fm \in Fm$ dan $p \in I_{\text{prod}}$

Sebagai *worst case*, misal graf yang terbentuk antara I_{func} dan I_{spec}, serta I_{spec} dan I_{prod}, masing-masing menyerupai sebuah graf *bipartite*, seperti ditunjukkan pada Gambar 5.5 dan 5.6.



Gambar 5.5. Graf *bipartite* antara I_{func} dan I_{spec}



Gambar 5.6. Graf *bipartite* antara I_{spec} dan I_{prod}

Dari formulasi fungsi *isSatIndF*(*p*,*fm*),

$$isSatIndF(p, fm) = \begin{cases} true, & parents(specF(fm)) = \\ & parents(specF(fm) \cap specP(p)) \\ false, & otherwise \end{cases}$$

Untuk suatu *fm*, *specF*(*fm*) menghasilkan sebuah himpunan I_{spec} dengan kardinalitas |*Ispec*|. Sementara itu, *specP*(*p*) menghasilkan himpunan

dengan kardinalitas $|\mathcal{I}_{spec}|$. Operasi yang dominan dalam *isSatIndF* ini adalah operasi irisan dari *specF*(*fm*) dan *specP*(*p*), dengan banyak operasi pencocokan adalah $|\mathcal{I}_{spec}|^2$. Dengan demikian, kompleksitas untuk operasi irisan ini adalah sebagai berikut,

$$T(|\mathcal{I}_{spec}|) = |\mathcal{I}_{spec}|^2 \qquad (5.7)$$

Karena jumlah node *parents*(*specF*(*fm*)) maupun *parents*(*specF*(*fm*) ∩ *specP*(*p*)) jauh lebih kecil daripada $|\mathcal{I}_{spec}|$, maka hal ini dapat diabaikan.

Mengacu pada Gambar 5.5, untuk sebuah kebutuhan fungsional *fm*, sistem melakukan pemeriksaan terhadap semua individu produk. Dengan mengacu pada Persamaan 5.6, banyaknya operasi pencocokan dalam proses ini,

$$T(|\mathcal{I}_{spec}|, |\mathcal{I}_{prod}|) = |\mathcal{I}_{spec}|^2 \, |\mathcal{I}_{prod}| \qquad (5.8)$$

Pada algoritma *recommend*, sistem mencari produk yang memenuhi semua $fm \in Fm$ memenuhi. Untuk *worstcase*, seperti yang ditunjukkan pada gambar 5.5 dan 5.6, sistem akan melakukan operasi pada Persamaan 5.7 untuk semua $allInd(fm) \in Fm$, sehingga, kompleksitas waktu untuk operasi ini dapat didefinisikan,

$$T(|\mathcal{I}_{spec}|, |\mathcal{I}_{prod}|, |allInd(fm)|) = |\mathcal{I}_{spec}|^2 \, |\mathcal{I}_{prod}| \, |allInd(fm)| \qquad (5.9)$$

Dalam dunia nyata, variabel yang lebih cepat bertumbuh adalah $|\mathcal{I}_{prod}|$, seiring dengan banyaknya produk yang diproduksi, walopun tidak bersifat kumulatif, karena tentu produk lama juga akan hilang dari pasar. Sementara itu, *nodes* pada hirarki *FuncReq*, *Specification*, tidak mengalami banyak perubahan. Dengan demikian, variabel yang paling berpengaruh dalam skalabilitas sistem adalah $|\mathcal{I}_{prod}|$.

Jika dipandang kompleksitas waktu sebagai fungsi dari $|\mathcal{I}_{prod}|$, dengan,

$$c = |\mathcal{I}_{spec}|^2 \, |allInd(fm)|$$

maka kompleksitas waktu dapat diformulasikan dengan,

$$T(|\mathcal{I}_{prod}|) = c \, |\mathcal{I}_{prod}| \qquad (5.10)$$

Sehingga, kompleksitas waktu,

$$T(|\mathcal{I}_{prod}|) \in O(|\mathcal{I}_{prod}|) \qquad (5.11)$$

Kompleksitas waktu *asymptotic*k pada persamaan 5.10 menunjukkan bahwa kompleksitas waktu algoritma masih dalam kelas linier, sehingga masih *acceptable* dari sisi waktu. Dalam ontologi, $|\mathcal{N}_{func}| = 51$, $|\mathcal{I}_{func}| = 31$, $|\mathcal{I}_{spec}| = 60$, serta dan $|\mathcal{I}_{prod}| = 288$.

5.4 *Resume*

Bab ini membahas mengenai model komputasional untuk rekomendasi produk dalam CRS. Model komputasional meliputi definisi fungsi matematis, algoritma, aksioma yang berlaku pada model rekomendasi ini. Model komputasional untuk rekomendasi ini memanfaatkan eksplorasi relasi semantik dalam ontologi. Untuk bab selanjutnya, akan dibahas mengenai model komputasional untuk pembangkitan pertanyaan serta penjelasan mengapa suatu produk direkomendasikan. Model pembangkitan pertanyaan dan penjelasan ini memanfaatkan fungsi-fungsi dan aksioma yang telah didefinisikan dalam bab ini.

6

MODEL KOMPUTASIONAL UNTUK PEMBANGKITAN PERTANYAAN DAN PENJELASAN

6.1 Pendahuluan

Untuk mengetahui kebutuhan pengguna, sistem memberikan beberapa pertanyaan secara eksplisit kepada pengguna. Pada bab ini akan dibahas model dalam membangkitkan pertanyaan dan penjelasan, yang disajikan dalam bentuk definisi fungsi-fungsi, algoritma beserta aksioma. Sebelum melangkah ke pembahasan model yang lebih detil, perlu dilakukan *review* ke belakang tentang mekanisme interaksi dalam CRS. Kebutuhan pengguna yang didapatkan selama interaksi disimpan dalam *user profile model*. *User profile model* ini selalu di-*update*mutakhirkan sepanjang interaksi (*node* bisa bertambah maupun berkurang). Sebuah pertanyaan berupa sekumpulan opsi, dan pengguna dapat memilih satu atau lebih opsi tersebut.

Pada saat awal interaksi, pertanyaan terdiri dari kebutuhan-kebutuhan fungsional, jenis-jenis produk (*PType*) atau properti-properti produk (*Pproperty*). Untuk setiap kebutuhan-kebutuhan fungsional yang dipilih, pengguna dapat memutuskan apakah kebutuhan-kebutuhan tersebut merupakan himpunan *mandatory functional requirement* (*Fm*) atau himpunan *optional functional requirement* (*Fo*). Pada interaksi berikutnya, pertanyaan berupa beberapa opsi kebutuhan fungsional saja.

Goal dari model pembangkit pertanyaan dalam CRS adalah mendapatkan himpunan kebutuhan yang sebenarnya dari pengguna, melalui pertanyaan-pertanyaan yang diajukan. Model komputasional ini dapat dipandang sebagai algoritma *local search*. Setiap tahap interaksi dalam percakapan, dipandang sebagai tahap pencarian solusi. Pada setiap tahap, pencarian solusi dilakukan secara lokal, dengan kriteria yang akan dibahas lebih detil dalam bab ini (tergantung pada *feedback* pengguna). Dari *nodes* yang didapatkan tersebut (selanjutnya disebut sebagai *candidate nodes*, sebagai kebutuhan yang potensial disukai oleh pengguna), kemudian dipilih beberapa *nodes* secara random, untuk ditanyakan. Himpunan *nodes* yang ditanyakan pada suatu tahap interaksi dianggap merupakan solusi (walaupun mungkin belum solusi yang seharusnya).

Solusi yang dihasilkan oleh model pembangkit interaksi ini, bersifat *complete*. Solusi *complete* bisa didapatkan, karena pengambilan random dari *candidate nodes* dilakukan tanpa pengembalian (*without replacement*), dan terdapat suatu mekanisme *backtracking* untuk mencari nodes yang potensial untuk disukai pengguna dan belum ditanyakan (selanjutnya disebut sebagai *unexplored candidate nodes*). Solusi dari *goal* didapatkan jika pengguna memilih satu atau lebih produk yang direkomendasikan.

Strategi pembangkitan pertanyaan tergantung pada *feedback* dari pengguna. *Feedback* dari pengguna berupa jawaban pertanyaan maupun pemilihan produk dari daftar produk yang direkomendasikan. Dalam Bab 5 telah dibahas tentang 5 kasus *feedback* dari pengguna, maupun strategi penanganannya secara umum. Dalam bab ini akan dibahas secara lebih detil strategi pembangkitan pertanyaan berdasarkan 5 macam kasus *feedback* tersebut. Selain itu, bab ini juga membahas tentang model untuk pembangkitan fasilitas penjelasan, mengapa suatu produk direkomendasikan.

6.2 Model untuk Pembangkitan Pertanyaan

Permasalahan utama dalam pembangkitan pertanyaan adalah menentukan beberapa *node* dalam hirarki *FuncReq* ($\mathcal{N}_{func}$) yang potensial untuk ditanyakan (untuk selanjutnya, disebut sebagai *candidat nodes*). Aturan dasar untuk menentukan *candidat nodes* disajikan dalam Aksioma 6.1. *Candidate nodes* ini didapatkan berdasarkan *reasoning* yang melibatkan *user profile model* dan ontologi. Sementara itu, Definisi VI.1 mendefinisikan fungsi *selectQ* untuk memilih beberapa *node* dari *candidate nodes* sebagai pertanyaan.

Aksioma 6.1. Jika A adalah himpunan *mandatory functional requirements* (Fm) atau optional functional requirements (Fo), maka children(A) adalah himpunan candidate nodes.

Berdasarkan Aksioma 6.1, pencarian *candidat nodes* melibatkan kebutuhan pengguna dalam *user profile model*, tanpa membedakan *Fm* dan *Fo*. Hal ini dikarenakan, sistem berusaha memandu pengguna memberikan pertanyaan (dalam proses *query refinement*), dengan pertanyaan yang lebih spesifik dari pertanyaan sebelumnya, baik itu *Fm* maupun *Fo*. Sistem juga membuka peluang bagi pengguna untuk mengubah kebutuhan yang mungkin tadinya *Fo*, dapat menjadi *Fm*, atau sebaliknya (walaupun ini berarti terdapat inkonsistensi). Dengan demikian, kebutuhan pengguna akan selalu dinyatakan sebagai *F*, dengan $F=Fm \cup Fo$. *Fm* dan *Fo* hanya dibedakan pada saat rekomendasi produk. *Fm* berpengaruh untuk menentukan produk-produk yang direkomendasikan, sedangkan *Fo* untuk menentukan nilai *utility* dari tiap produk.

Berdasarkan hasil konsultasi dengan praktisi di bidang *sales*, seorang *sales* tidak menanyakan banyak pertanyaan pada satu tahap interaksi, sehingga tidak terlalu membebani pelanggan. Oleh karena itu, dalam proses pembangkitan pertanyaan pada model yang diajukan, pada setiap interaksi, sistem tidak menanyakan semua *candidat nodes*, tetapi diambil beberapa dari *candidate nodes* untuk ditanyakan, dengan fungsi *selectQ,* seperti didefinisikan pada Definisi 6.1.

Definisi 6.1 (Fungsi *selectQ*). Fungsi ini memilih sejumlah *node B* dari *candidate nodes* A sebagai pertanyaan. Fungsi ini memetakan himpunan integer (*INT*) dan himpunan *node* dalam hirarki *FuncReq*, ke himpunan *node* dalam in hirarki *FuncReq*.

$$selectQ: INT \times \mathcal{N}_{func} \rightarrow \mathcal{N}_{func}$$

Misal *max_questions* adalah integer yang menyatakan jumlah maksimal *node* yang ditanyakan. Dengan demikian, himpunan *node* yang akan ditanyakan diperoleh melalui,

$$selectQ(\max\_questions, A) = \left\{b \in B \middle| B \subseteq A, |B| = \min\left\{\max\_question, |A|\right\}\right\}$$

Dengan *A* adalah himpunan *candidate nodes*.
Pendekatan paling sederhana, sebagaimana diimplementasikan dalam sistem ini, memilih secara random *nodes* dari himpunan *candidate nodes*, sebagai pertanyaan.

Sistem mempersiapkan pertanyaan berdasarkan 5 macam *feedback* pengguna, 1) *empty user profile*, 2) terdapat lebih dari satu produk yang dipilih pengguna, 3) tidak ada produk yang dipilih oleh pengguna, 4) definisi kebutuhan belum cukup untuk membangkitkan rekomendasi, 5) tidak ada produk yang sesuai dengan *user profile model*. Model pembangkitan pertanyaan dikembangkan berdasarkan pengamatan tentang bagaimana cara seorang *domain expert* memandu seorang peminat produk, melalui percakapan yang dilakukan.

6.2.1 Kasus empty_profile – User Profile Model Masih Kosong

Kasus ini dapat terjadi di awal maupun tengah interaksi. Kasus *empty_profile* terjadi di tengah interaksi, dikarenakan sistem membaca bahwa pengguna ingin mengubah kebutuhan dari awal, di tengah interaksi. Dengan demikian sistem perlu melakukan reset terhadap *user profile model* (lihat Subbab 6.2.3 tentang kasus *no_chos_prod*). Gambar 6.1 menunjukkan langkah dalam algoritma untuk membangkitkan pertanyaan untuk kasus *empty_profile*.

***generateGeneralQ*(α)** → himpunan *node* untuk ditanyakan (pertanyaan) //Membangkitkan pertanyaan yang paling umum ketika *user profile model* masih kosong

input : α adalah maksimum jumlah *node* yang ditanyakan
// $N_{func(1)}$ *is* N_{func} *level* 1 dalam hirarki *FuncReq*
1. **if** initial_interaction **then**
2. **return**(s***elect*****Q**(α,$N_{func(1)}$)) $\cup$*PType*$\cup$*Pproperty*
3. **else**
4. return (***select*****Q**(α,$N_{func(1)}$))
5. **end if**

Gambar 6.1 Algoritma untuk membangkitkan pertanyaan saat *user profile model* kosong

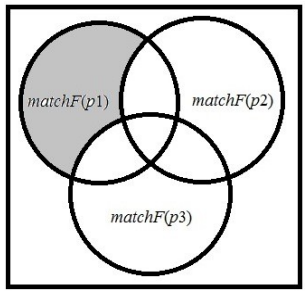
Jika kasus ini terjadi di awal interaksi, sistem memberikan pertanyaan berupa kebutuhan-kebutuhan fungsional, jenis-jenis produk (*PType*) atau properti-properti produk (*Pproperty*). Kebutuhan fungsional yang ditanyakan adalah kebutuhan fungsional yang masih bersifat umum, yaitu *node* level 1 pada hirarki *FuncReq* (Aksioma V.1), seperti ditunjukkan pada baris 2 dari algoritma. Jika kasus ini terjadi di tengah interaksi, sistem hanya menyakan kebutuhan fungsional yang bersifat umum (baris 4).

6.2.2 Kasus *multi_chos_prod* – Terdapat Lebih dari Satu Produk yang Dipilih Pengguna

Ketika pengguna memilih dua atau lebih produk, hal ini menandakan bahwa pengguna ragu-ragu antara k produk tersebut ($k > 1$). Dalam kasus ini, sistem membangkitkan k *distinctive functional requirements* dari masing-masing produk sebagai pertanyaan, untuk membantu pengguna memilih produk yang diinginkan. *Distinctive functional requirements* dari sebuah produk adalah kebutuhan fungsional dipenuhi oleh produk tersebut, tetapi tidak dipenuhi oleh produk lain.

Tujuan dari menampilkan *distinctive functional requirements* adalah untuk membantu pengguna mengambil keputusan dengan memperlihatkan perbedaan kebutuhan fungsional produk yang dipenuhi masing-masing produk. Misal $P = \{p_1, p_2, p_3\}$ adalah himpunan produk yang dipilih oleh pengguna. Sementara itu, $DF(p_1)$ adalah himpunan *distinctive functional requirements* dari produk p_1. Dalam gambar 6.2, $DF(p_1)$ diilustrasikan sebagai bagian yang diarsir. Misal $match(p_i)$ adalah himpunan individu dari kebutuhan fungsional yang dipenuhi oleh sebuah produk p_i, dengan demikian $DF(p_i)$ dapat diformulasikan sebagai berikut,

$$DF(p_1) = matchF(p_1) - (matchF(p_2) \cup matchF(p_3)) \quad (6.2)$$



Gambar 6.2 Ilustrasi untuk *distinctive functional requirements* dari product p_1

Misal $P = \{p_i, p_2,..., p_k\}$ adalah himpunan produk yang dipilih oleh pengguna. Dengan demikian, untuk setiap $p \in P$, perumusan secara umum untuk $DF(p)$ adalah sebagai berikut,

$$DF(p) = matchF(p) - \left(\bigcap_{p' \in (P - \{p\})}^{|P-\{p\}|} matchF(p') \right) \quad (6.3)$$

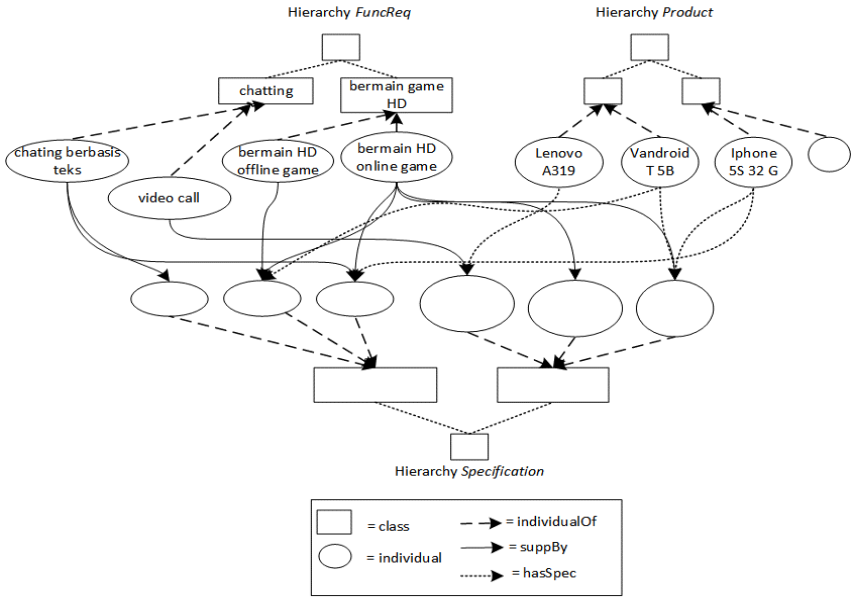
dengan,

$$match(p) = \{ f \in \mathrm{I}_{func} \mid isSatIndF(p, f) \} \quad (6.4)$$

generateDistinctiveQ(P) → himpunan *node distinctive functional requirements* untuk masing-masing produk yang dipilih pengguna //Membangkitkan *distinctive functional requirements* untuk masing-masing *k* produk yang dipilih pengguna.
input : *P* adalah himpunan produk-produk yang dipilih pengguna 1. **for each** $p \in P$ **do** 2. Tentukan *matchF*(*p*) menggunakan Persamaan 6.4 3. **end for** 4. $Q \leftarrow \{\}$ 5. **for each** $p \in P$ **do** 6. Tentukan *DF*(*p*), menggunakan Persamaan 6.3 7. **end for** 8. **for each** $p \in P$ **do** 9. $Q \leftarrow Q \cup DF(p)$ 10. **end for** 11. **return** *Q*

Gambar 6.3 Algoritma untuk membangkitkan saat pengguna memilih lebih dari satu produk

Gambar 6.3 menunjukkan algoritma ***generateDistinctive*** untuk menyajikan *distinctive functional requirements* dari *k* produk yang dipilih pengguna. Pertama-tama, sistem menentukan kebutuhan fungsional yang dipenuhi oleh masing-masing produk. Sebagai ilustrasi, contoh ontologi dapat dilihat pada Gambar 6.4. Misal pengguna memilih 3 produk dari sekian produk yang direkomendasikan, iphone 5S 32G, Lenovo A319 dan Vandroid T5B. Untuk menentukan *matchF*(*p*) untuk masing-masing produk $p \in P$ (baris 1 dan 2 algoritma), pemeriksaan *isSatIndF*(*p*,*f*) dilakukan untuk setiap individu kebutuhan fungsional, $f \in \mathcal{I}_{func}$. Sebagai ilustrasi sederhana, misal terdapat 4 individu kebutuhan fungsional, *chatting berbasis teks*, *video call*, *bermain HD online game*, *bermain HD offline game*. Pemeriksaan nilai *isSatIndF*(*p*,*f*) dilakukan dengan cara yang sama dengan contoh 5.1, untuk mendapatkan *matchF*(*p*) masing-masing produk. Seperti halnya pada contoh 5.1, pemeriksaan ini melibatkan relasi *individualOf*, *suppBy* dan *hasSpec* pada ontologi, seperti ditunjukkan pada Gambar 6.4. Misal didapatkan *matchF*(iphone 5S 32G) = {*chatting berbasis teks*, *video call*, *bermain HD offline game*}, *matchF*(Lenovo A319) = {*chatting berbasis teks*} dan *matchF* (Vandroid T5B) = {*chatting berbasis teks*, *bermain HD online game*}.



Gambar 6.4 Contoh potongan ontologi untuk kasus *multi_chos_prod*

Pada baris 5-6 algoritma, ditentukan himpunan *DF*(*p*) untuk masing-masing produk dengan Persamaan 6.3.

DF(iphone 5S 32G) = *matchF*(iphone 5S 32G) – (*matchF*(Lenovo A319) ∪ *matchF*(Vandroid T5B)) = {*video call*, *bermain HD offline game*}

DF(Lenovo A319) = matchF(Lenovo A319) – (matchF(iphone 5S 32G) ∪ matchF(Vandroid T5B)) = { }
DF(Vandroid T5B)) = matchF(Vandroid T5B) – (matchF(Lenovo A319) ∪ matchF(iphone 5S 32G)) = {bermain HD online game}

Dari sini, pengguna dapat melihat, bahwa Lenovo A319 *inferior* daripada dua produk yang lain. Selain itu, pengguna dapat mempertimbangkan keunggulan dari masing-masing 2 produk yang lain.

Dari contoh di atas, dapat dilihat bahwa *DF*(Lenovo A319) adalah himpunan kosong. Terdapat dua macam kasus, saat himpunan *distinctive functional requirements* dari sebuah produk adalah himpunan kosong:

1. Jika sebuah produk *inferior* dari produk lain, maka himpunan *distinctive functional requirements* dari produk tersebut adalah kosong. Secara formal dapat dirumuskan sebagai berikut,
   jika $matchF(p_i, F) \subset matchF(p_j, F)$ maka $DF(p_i, F) = \varphi,\ p_i \neq p_j$
   Notasi "$\subset$" menyatakan *proper subset*
2. Jika individu dari kebutuhan fungsional yang dipenuhi oleh dua produk adalah sama, maka himpunan *distinctive functional requirements* dari masing-masing produk adalah kosong.
   Jika $match(p_i, F) = match(p_j, F)$
   maka $DF(p_i, F) = DF(p_j, F) = \varphi, \quad \text{for } p_i \neq p_j$

Ketika *distinctive functional requirements* dari semua produk merupakan himpunan kosong, sistem akan menunjukkan pembandingan tingkatan kualitas spesifikasi dari masing-masing produk sebagai alternatif bantuan bagi pengguna.

6.2.3 Kasus *less_specific* – Kebutuhan Pengguna Masih Belum Cukup untuk Rekomendasi Produk

Keadaan ini terjadi saat kebutuhan pengguna dalam *user profile model* masih terlalu umum, dan menghasilkan terlalu banyak produk yang direkomendasikan. Dengan demikian sistem akan memberikan pertanyaan berupa kebutuhan fungsional yang lebih spesifik (proses *query refinement*). Berdasarkan Aksioma 6.1 dan 6.1, untuk mendapatkan kebutuhan fungsional yang lebih spesifik, sistem akan mengambil anak-anak dari *candidat nodes* saat ini. Algoritma dalam membangkitkan pertanyaan, selengkapnya dilakukan oleh algoritma *generateSpecificQ*, seperti ditunjukkan pada Gambar 6.5.

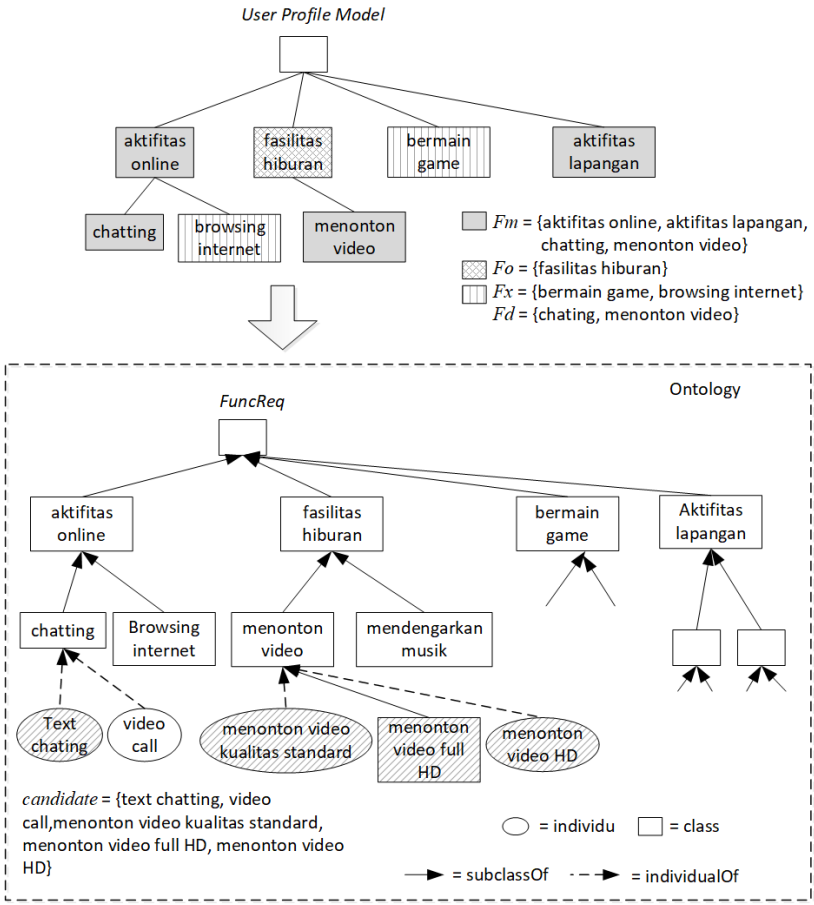
***generateSpecificQ*(*userModel*, α)**→ himpunan *node* yang ditanyakan (pertanyaan) //Membangkitkan pertanyaan yang lebih spesifik daripada pertanyaan saat ini
input : *userModel* adalah *user profile model* saat ini, α adalah maksimum jumlah pertanyaan 1. $d \leftarrow$ level terdalam dari *userModel* 2. // Kebutuhan fungsional yang diingginkan pengguna saat ini adalah $Fm \cup Fo$ pada level terdalam dalam *userModel* $F_d \leftarrow [Fm_d \cup Fo_d]$ 3. Tentukan himpunan *candidate nodes* yang lebih spesifik daripada F_d, menggunakan Persamaan 6.5 4. **if** *candidate* $\neq \phi$ **then** 5. **return** *selectQ*(α,*candidate*) 6. **else** 7. // jika semua kebutuhan fungsional pengguna adalah *node* daun **return *generateUnExploredQ*** (*userModel*) 8. **end if**

Gambar 6.5 Algoritma untuk membangkitkan pertanyaan saat *user profile model* belum cukup untuk rekomendasi

Untuk membangkitkan pertanyaan, sistem membutuhkan informasi tentang kebutuhan fungsional yang baru saja dipilih oleh pengguna (F_d) yang didapatkan dari $\left[Fm_d \cup Fo_d\right]$, notasi d menyatakan level paling dalam dari

user profile model. *Node Fx* tidak diperhitungkan dalam proses *query refinement* ini. Sebagai contoh kasus, dapat dilihat dalam Gambar 6.6. Misal F_d = {*Chatting*, *Surfing internet*, *Watching video*}.

Tugas dari sistem adalah mencari *nodes* dalam hirarki *FuncReq* ($\mathcal{N}_{func}$) sebagai himpunan *node* yang akan ditanyakan, sedemikian hingga jumlah produk yang memenuhi kebutuhan pengguna berdasarkan pertanyaan ini, lebih sedikit daripada sebelumnya.



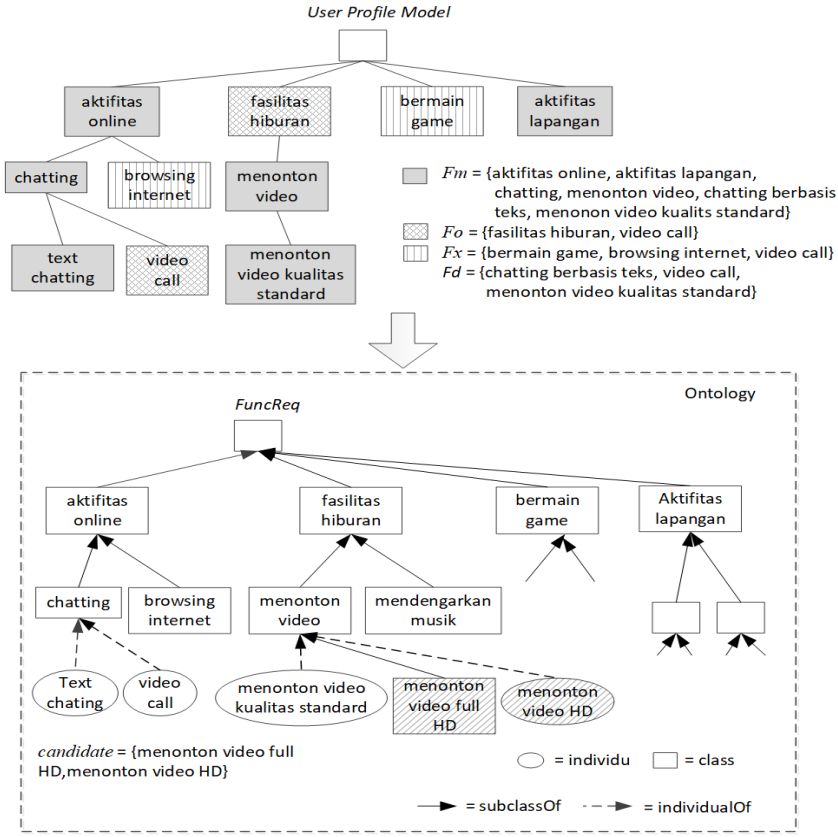
Gambar 6.6 Contoh kasus *less_specific* dengan *candidate* ≠ ϕ

Pertama-tama, sistem menentukan *candidate nodes* yang lebih spesifik daripada F_d (dinotasikan sebagai *candidate*). Berdasarkan Aksioma 5.1 dan 6.1, *candidate* merupakan anak-anak dari $\mathcal{N}_{func}$ yang berkorespondensi dengan F_d sebagai berikut (baris 2 dalam algoritma),

$$candidate = children(correspond(F_d)) \qquad (6.5)$$

Dalam kasus yang disajikan pada Gambar 6.6 , fungsi *correspond*(*Fd*) memetakan F_d dalam *user profile model* dengan *node* kebutuhan fungsional dalam ontologi yang bersesuaian. Dengan demikian didapatkan *candidate* = {*text chatting*, *menonton video kualitas standard*, *menonton video full HD*, *menonton video HD*}.

Dalam baris 5 algoritma, *nodes* yang ditanyakan dipilih secara random dari *candidate*, sebanyak maksimal *α node*. Pendekatan yang lebih *advanced* dari cara random adalah menggunakan mekanisme *learning* berdasarkan pengalaman dari interaksi - interaksi pengguna sebelumnya.



Gambar 6.7 Contoh kasus *less_specific* dengan *candidate* = ϕ

Terdapat keadaan jika kondisi *candidate* = ϕ, dalam arti semua elemen dalam *Fd* adalah individu (baris 6, 7 dari algoritma). Gambar 6.7 mengilustrasikan kasus ini. Misal F_d = {*text chatting, video call, menonton video kualitas standard* }. Berdasarkan pemetaan yang dilakukan dalam fungsi *correspond*(F_d), diketahui *nodes* ini merupakan individu dalam ontologi. Dengan demikian, mereka tidak mempunyai anak. Untuk kasus ini, sistem akan mencari *candidat nodes* alternatif yang belum ditanyakan (dinyatakan sebagai *unexplored candidat nodes*), dengan memanggil algoritma ***generateUnexploredQ***(*userModel*) (lihat Subbab 6.2.3). Dalam kasus yang disajikan pada Gambar 6.7, sistem mendapatkan *unexplored candidat nodes* dengan mencari *children* dari [$Fm \cup Fo$] yang berada dalam level di atas F_d. Dalam kasus ini, *unexplored candidat nodes* = {*menonton video full HD*, *Menonton video HD*}.

Dalam baris 5 algoritma, sistem memilih paling banyak α *node* dari *candidate nodes*, dan menghasilkan *subset* dari *candidate*. Dengan demikian, jumlah produk yang memenuhi subset ini akan lebih sedikit daripada jumlah produk yang memenuhi F_d. Kondisi ini terpenuhi dengan asumsi bahwa untuk setiap *mandatory functional requirement* Fm_d, pengguna selalu memilih minimal satu dari *children*(Fm_d) sebagai *mandatory functional requirement*. Proposisi 6.3 di bawah, dapat memperjelas kondisi ini.

Proposisi 6.3. Misal *P* adalah himpunan produk dan $A = \{a_1, a_2, ..., a_n\}$ adalah himpunan *mandatory functional requirements* dalam hirarki *FuncReq*. Jika *B* adalah subset dari *children*(*A*), maka jumlah produk yang memenuhi *B* akan lebih sedikit daripada produk yang memenuhi *A*. Secara formal, dapat dinyatakan sebagai berikut,

Jika $B \subseteq children(A)$

maka

$$|satProd(P, B)| \subseteq |satProd(P, A)| \qquad (6.6)$$

Bukti: Misal $B \subseteq children(A)$,

dengan $B = B_1 \cup B_2 \cup ... \cup B_m$ dan $B_1 \subseteq children(a_1)$, $B_2 \subseteq children(a_2)$,..., $B_m \subseteq children(a_m)$

Untuk setiap $a_i \in A$, produk-produk yang memenuhi semua $b_{ij} \in B_i$ (i, j>0) merupakan subset dari produk-produk yang memenuhi paling sedikit salah satu dari *children*(a_i),

$$\left\{ p \in P \middle|_{\bigwedge_{b_{ij} \in B_i}} isSatClassF(p, b_{ij}) \right\} \subseteq \left\{ p \in P \middle|_{\bigvee_{a_{ij} \in children(a_i)}} isSatClassF(p, a_{ij}) \right\},$$

untuk setiap $a_i \in A$ (6.7)

Berdasarkan Aksioma V.1, untuk masing-masing $a_i \in A$, produk-produk yang memenuhi a_i adalah sama dengan produk-produk yang memenuhi paling sedikit salah satu dari *children*(*A*),

$$\{p \in P | isSatClassF(p, a_i)\} = \left\{ p \in P \middle| \bigvee_{a_{ij} \in children(a_i)} isSatClassF(p, a_{ij}) \right\},$$

untuk setiap $a_i \in A$, $i > 0$ (6.8)

Dari Persamaan 6.7 dan 6.8, dapat disimpulkan bahwa produk-produk yang memenuhi semua of $b_{ij} \in B_i$ ($i, j > 0$) merupakan *subset* dari produk-produk yang memenuhi a_i,

$$\left\{ p \in P \middle| \bigwedge_{b_{ij} \in B_i} isSatClassF(p, b_{ij}) \right\} \subseteq \{p \in P | isSatClassF(p, a_i)\},$$

Untuk setiap $a_i \in A$ (6.9)

Berdasarkan Definisi 5.3, untuk himpunan *mandatory functional requirements A* dan himpunan produk *P*, produk-produk yang memenuhi *A* adalah produk-produk yang memenuhi semua elemen dari *A*, sehingga dari Persamaan VI.9, didapatkan formulasi sebagai berikut,

$$\left\{ p \in P \middle| \bigwedge_{B_i \subseteq children(A)} \left(\bigwedge_{b_{ij} \in B_i} isSatClassF(p, b_{ij}) \right) \right\} \subseteq \left\{ p \in P \middle| \bigwedge_{a_i \in A} (isSatClassF(p, a_i)) \right\} \quad (6.10)$$

Dengan demikian, dapat disimpulkan bahwa,

$$satProd(P, B) \subseteq satProd(P, A)$$

Dapat juga dikatakan bahwa jumlah produk yang memenuhi *B* adalah kurang dari jumlah produk yang memenuhi *A*,

$$|satProd(P, B)| \leq |satProd(P, A)| \quad (6.11)$$

Jika semua kebutuhan fungsional dari pengguna berupa individu, eksplorasi lebih lanjut terhadap *user profile model* harus dilakukan untuk mendapatkan *unexplored candidate nodes*, seperti ditunjukkan pada baris 6 dan 7 dari algoritma.

6.2.4 Kasus *no_chos_prod* – Tidak Ada Produk yang Dipilih Pengguna

Selama proses rekomendasi produk, sistem akan menyajikan *n* produk yang sesuai dengan kebutuhan pengguna, diurutkan berdasarkan nilai *utility*. Dengan demikian, jika seorang pengguna tidak memilih salah satu produk-pun, maka sistem akan merekomendasikan *n* produk lain dengan nilai *utility* tertinggi. Jika tidak ada produk yang dapat direkomendasikan lagi, sistem akan memanfaatkan beberapa *unexplored candidate nodes* sebagai pertanyaan berikutnya.

***generateUnExploredQ*(*userModel*, α)**→ himpunan *node* yang ditanyakan (pertanyaan) //Membangkitkan beberapa *unexplored candidate nodes* sebagai pertanyaan
input : *userMode l*adalah *user profile model*, α adalah jumlah maksimum *node* yang ditanyakan 1. *j*← level *node* dalam *userModel* yang ditanyakan saat ini 2. Tentukan himpunan *unexplored candidate nodes* (*unExCandidate*) *unExCandidate* = ***searchUnExCandidate*(*j*)** 3. **if** *unExCandidate* ≠ ϕ **then** 4. **return *selectQ***(α, *unExCandidate*) 6. **else** 7. Kembalikan *userModel* menjadi Kosong 8. **return *generateGeneralQ*(α)** 9. **end if**

Gambar 6.8 Algoritma untuk membangkitkan pertanyaan saat pengguna tidak memilih satupun produk yang direkomendasikan

Gambar 6.8 menunjukkan langkah-langkah dalam algoritma *generateUnExploredQ* untuk mengatasi keadaan, saat tidak ada produk yang dapat diajukan lagi kepada pengguna. Pertanyaan dibangkitkan berdasarkan level dari *node* pada *user profile model* yang ditanyakan saat ini (baris 1). Kemudian, *unexplored candidate node*s didapatkan melalui algoritma *searchUnExCandidate*, seperti ditunjukkan dalam baris 2 dari algoritma. Kemudian sejumlah maksimal α *node* akan dipilih dari *unexplored candidate nodes* sebagai pertanyaan (baris 4). Jika *unexplored candidate nodes* sudah tidak tersedia lagi, hal ini menandakan bahwa pengguna ingin mengubah kebutuhannya. Dengan demikian, *user profile*

model akan di-*reset* dan kemudian sistem akan membangkitkan kebutuhan fungsional yang paling umum sebagai pertanyaan (baris 6 - 9).

Gambar 6.9 menunjukkan langkah - langkah dalam algoritma *searchUnExCandidate* yang digunakan untuk mencari *unexplored candidate nodes*. Ini merupakan proses rekursif yang melibatkan *candidate nodes* dan *explored candidate nodes* dalam berbagai level. Proses ini dimulai dari level *node* yang baru saja ditanyakan dan berhenti ketika proses mencapai *node* level 1 pada *user profile model*. *Candidate nodes* dalam level tertentu merupakan *nodes* yang merepresentasikan *Fm* atau *Fo* pada level tersebut. Proses ini melibatkan eksplorasi dan pemetaan antara *nodes* dalam *user profile model* dan hirarki *FuncReq*.

***searchUnExCandidate*(*i*)**→ himpunan *unexplored candidate nodes* //Melakukan *Backtracking* untuk mendapatkan *unexplored candidate nodes*
input *: i,* (*i*>*0*) adalah integer yang menyatakan level dari *node* dalam *userModel*. 1. **if** $i = 1$ (pada level1) **then** 2. Dapatkan *unexplored candidate nodes* pada level 1 ($unExCandidate_1$) menggunakan Persamaan VI.16 3. **return** $unExCandidate_1$ 4. **else** (tidak pada level1) 5. Dapatkan *unexplored candidat nodes* pada level *i* ($unExCandidate_i$) menggunakan Persamaan VI.15 6. **if** $unExCandidate_i \neq \phi$ **then** 7. **return** $unExCandidate_i$ 8. **else** 9. //Backtrack kelevel di atas level saat ini, secara rekursif **return** *searchUnExCandidate*(*i*-1) 10. **end if** 11. **end if**

Gambar 6.9 Algoritma untuk mencari *unexplored candidate nodes*

Misal *i* adalah level dari *node* dalam *user profile model* yang diobservasi. Jika $i > 1$ maka sistem mencari *unexplored candidate nodes* pada level *i* (baris 5). *Unexplored candidate nodes* didapatkan melalui langkah-langkah sebagai berikut. Misal $candidate_i$ adalah himpunan *candidate nodes* pada level *i*. Berdasarkan Aksioma 6.1, himpunan $candidate_i$ diperoleh dengan mencari anak-anak dari $\mathcal{N}_{func}$ level *i*-1 yang berkorespondensi dengan *mandatory* atau *optional functional requirements* $(Fm \cup Fo)$ pada level di atasnya, seperti ditunjukkan pada Persamaan 6.12. Demi konsistensi, akan

selalu didefinisikan $F = Fm \cup Fo$, dengan F adalah kebutuhan fungsional yang diinginkan oleh pengguna.

$$candidat_i = children(correspond(F_{i-1})) \qquad (6.12)$$

Sementara itu, himpunan *node*s pada level *i* yang sudah ditanyakan, didefinisikan sebagai *explored candidat nodes* pada level *i* (dinotasikan dengan $exCandidate_i$). Dengan demikian, $exCandidate_i$ merupakan $\mathcal{N}_{func}$ yang berkorespondensi dengan himpunan kebutuhan fungsional yang diinginkan pengguna (*F*), atau kebutuhan yang tidak diperlukan pengguna (*Fx*) dalam *user profile model* pada level *i*, yang dapat diformulasikan sebagai berikut,

$$exCandidate_i = correspond(F_i \cup Fx_i) \quad (6.13)$$

Tugas sistem adalah mencari *candidat nodes* pada level *i* yang belum ditanyakan, (dinotasikan dengan $unExCandidat_i$), yang dapat dinyatakan dalam persamaan sebagai berikut,

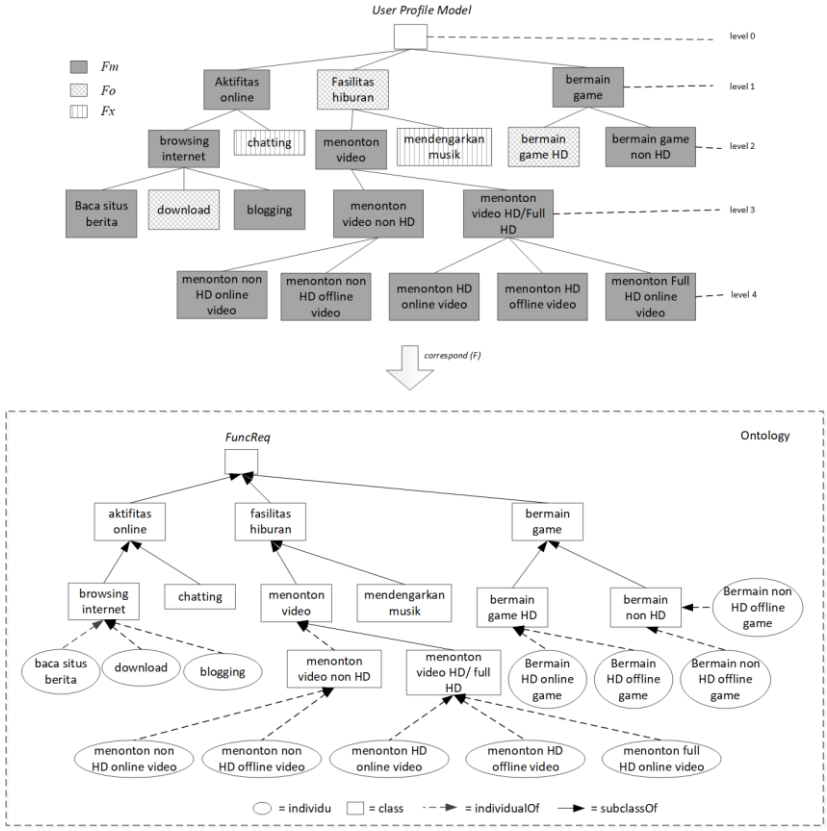
$$unExCandidate_i = candidate_i - exCandidate_i \quad (6.14)$$

Berdasarkan Persamaan 6.12 dan 6.13, persamaan 6.14 dapat diformulasikan kembali sebagai berikut,

$$unExCandidat_i = children(correspond(F_{i-1})) - correspond(F_i \cup Fx_i) \quad (6.15)$$

Jika tidak ditemukan *unexplored candidate nodes* pada suatu level, maka dilakukan *backtracking* menuju $\mathcal{N}_{func}$ pada level di atasnya, untuk mencari *unexplored candidate nodes* (baris 9). Proses *backtracking* akan berhenti ketika sudah mencapai level 1. Pada level ini, $candidate_1$ adalah $\mathcal{N}_{func}$ pada level 1($\mathcal{N}_{func(1)}$). Berdasarkan Persamaan 6.13 dan 6.14, *unexplored candidate nodes* dapat diperoleh dengan perumusan sebagai berikut,

$$unExCandidate_1 = \mathcal{N}_{func(1)} - correspond(F_1 \cup Fx_1) \quad (6.16)$$



Gambar 6.10 Contoh untuk kasus *no_chos_prod*

Contoh 6.1. Misal Keadaan *user profile model* ditunjukkan pada Gambar 6.10. Berdasarkan informasi kebutuhan yang ada dalam *user profile model*, sistem merekomendasikan produk-produk, dan pengguna tidak memilih satupun produk yang direkomendasikan. Dengan demikian, sistem harus memberikan pertanyaan kembali dengan mencari unexplored candidat *node*s. Mengacu pada algoritma *generateUnExploredQ*, pertama-tama, sistem akan membaca level *node* yang ditanyakan saat ini. Mengacu contoh kasus pada Gambar 6.10, misal sistem baru saja menanyakan pertanyaan *Q* = {*menonton non HD online video*, *menonton HD online video*}, yang berada dalam level 4. Selain itu, misal semua *node* lain pada level 4 sudah ditanyakan. Dengan demikian, sistem akan mencari *unexplored candidat nodes* pada level 3, untuk mendapatkan F_3. Dalam hal ini,

F_3 = {baca situs berita, download, blogging, menonton video non HD, menonton video HD/Full HD}

F_4 = {menonton non HD online video, menonton non HD offline video, menonton non HD online video, menonton HD online video, menonton Full HD online video}
$Fx_4 = \Phi$

dengan menggunakan Persamaan 6.15 (algoritma *searchUnExCandidate* baris 5),
$unExCandidate_4$ =
children(*correspond*(*baca situs berita, download, blogging, menonton video non HD, menonton video HD/Full HD*) – *correspond*(*menonton non HD online video, menonton non HD offline video, menonton non HD online video, menonton HD online video, menonton Full HD online video*)
= Φ

Karena hasilnya *Φ*, maka sistem akan mencari kembali *unexplored candidate node*s pada level lebih atas (level 3), dengan mencari *node Fm* dan *Fo* pada level 2, dan menerapkan Persamaan 6.15 kembali.

F_2 = {browsing internet, menonton video, bermain game HD, bermain game non HD}
F_3 = {baca situs berita, download, blogging, menonton video non HD, menonton video HD/Full HD}
$Fx_3 = \Phi$
$unExCandidate_4$ =
children(*correspond*(*browsing internet, menonton video, bermain game HD, bermain game non HD*) – *correspond*(*baca situs berita, download, blogging, menonton video non HD, menonton video HD/Full HD*)
= {Bermain HD online game, Bermain HD offline game, Bermain non HD offline game}

6.2.5 Kasus *contra_pref*– Tidak Ada Produk yang Sesuai dengan *User Profile Model*

Dalam kasus ini, tidak ada produk yang direkomendasikan berdasarkan kebutuhan pengguna yang disimpan dalam *user profile model*. Kondisi ini

muncul ketika paling sedikit satu *subset* dari $[Fm \cup PType \cup Pproperty]$ tidak menghasilkan produk yang sesuai (disebut sebagai *contradictory subset*). Mengacu pada algoritma ***recommend*** pada Gambar 5.2, ketiga himpunan ini merupakan *constraint* yang harus dipenuhi dalam merekomandasikan produk. Dalam keadaan ini, sistem akan meminta pengguna untuk melonggarkan *contradictory subset* ini.

generateContradictoryQ(userModel) → himpunan *node* yang ditanyakan (pertanyaan) //Membangkitkan pertanyaan yang terdiri dari himpunan bagian kebutuhan pengguna yang kontradiktif, yang menyebabkan tidak ada produk yang direkomendasikan
input : *userModel*, *user profile model* saat ini 1. *PR*← [*Fm* ∪ *Ptype* ∪ *Pproperty*] dari *userModel*. 2. Pangkas *PR* secara random satu per satu elemen, sampai ***recommend***(*PR*) ≠ ϕ 3. **return** *PR* terakhir yang dipangkas

Gambar 6.11 Algoritma untuk membangkitkan pertanyaan saat tidak ada produk yang sesuai *user profile model*

Gambar 6.11 menunjukkan langkah-langkah dalam algoritma *GenerateContradictoryQ* untuk menangani kasus ini. Tujuan dari algoritma ini adalah mencari *contradictory subset* secara heuristik. *Contradictory subset* ini diperoleh melalui pemangkasan (*pruning*) satu elemen dari $[Fm \cup PType \cup Pproperty]$ secara iteratif, sampai didapatkan himpunan produk yang sesuai. Pengambilan elemen yang dipangkas, dilakukan secara random tanpa pengembalian, dan *contradictory subset* yang didapatkan adalah $[Fm \cup PType \cup Pproperty]$ terakhir yang dipangkas. *Contradictory subset* ini akan disampaikan sebagai pertanyaan, dan pengguna dapat melonggarkan masing-masing elemen *Fm* dengan cara menghapus atau mengubahnya menjadi *optional functional requirement*, Sementara itu, *Ptype* dan *Pproperty* dapat dilonggarkan dengan cara memodifikasi nilainya.

Contoh 6.2. Misal kebutuhan dari pengguna, Misal *Fm* : {*mengedit dokumen, mendengarkan musik*}, *Fo* : {*bermain game HD*}. *Ptype*: {smartphone}, *Pproperty*: {rentang-harga: 2000.000 - 4.500.000 ; brand: apple; sistem operasi: {*Android, windows phone*}.

$PR = [Fm \cup PType \cup Pproperty] = \{$*mengedit dokumen, mendengarkan musik, smartphone, 2000.000 - 4.500.000, apple, Android, windows phone*$\}$.

Langkah pertama, pangkas *PR* dengan mengambil satu (tanpa pengembalian) elemen dari *PR*, dan lakukan rekomendasi produk berdasarkan PR yang baru. Misal,
PR' = {*mengedit dokumen, smartphone, 2000.000 - 4.500.000, apple, Android, windows phone*}

Dengan melakukan inferensi pada ontologi (lihat contoh V.2), misal masih tidak ditemukan produk, maka dilakukan pemangkasan kembali, dan menghasilkan himpunan *constraint* PR" baru yang sudah terpangkas satu elemen, misal :
PR" = {mengedit dokumen, smartphone, 2000.000 - 4.500.000, Android, windows phone}

Rekomendasi berdasarkan himpunan constraint *PR"* ini ternyata menghasilkan produk, sehingga sistem mengajukan *PR'* (himpunan *constraint* terakhir sebelum pemangkasan yang terakhir) sebagai pertanyaan ke pengguna, untuk direvisi atau dilonggarkan.

6.3 Model untuk Pembangkitan Penjelasan

Setelah proses rekomendasi, sistem akan memberikan penjelasan mengapa produk tersebut direkomendasikan. Penjelasan ini disajikan dalam bahasa natural dengan memanfaatkan *template-based explanation* (Bilsus dan Pazzani, 1999). Penjelasan ini menyatakan kebutuhan fungsional yang dipenuhi oleh produk, termasuk juga kebutuhan-kebutuhan fungsional spesifik yang terkait.

<*p*> "**sesuai dengan kebutuhan anda, karena produk ini merupakan** "<*type*(*p*)>" **dengan**" <*prop*(*p*)>. **Produk ini juga memenuhi kebutuhan anda untuk**: < f_i > "**khususnya untuk"** <f_ind_{ij}> untuk $i, j > 0$

Gambar 6.12 *Template* dari fasilitas penjelasan

Misal *recProd* adalah himpunan produk yang direkomendasikan dan $F = Fm \cup Fo$ adalah himpunan kebutuhan fungsional yang diinginkan pengguna, maka untuk setiap $p \in recProd$, *template* dari fasilitas penjelasan mengapa sebuah produk *p* direkomendasikan, disajikan oleh Gambar 6.12.

Dengan *prop*(*p*) adalah properti produk yang dimiliki oleh *p*, *type*(*p*) adalah jenis produk yang dimiliki oleh *p*. Unsur dalam penjelasan selanjutnya adalah komponen kebutuhan fungsional pengguna yang dipenuhi oleh produk (f_i) beserta kebutuhan fungsional spesifiknya (f_ind_{ij}).

Misal untuk suatu produk $p \in recProd$, dan $F = Fm \cup Fo$, kebutuhan fungsional pengguna dan dipenuhi oleh produk *p* didapatkan dengan,

$$req(p,F) = \{f \in F \mid satF(p,F)\} \qquad (6.17)$$

Fungsi *satReq*(*p*,*F*) disajikan dalam Persamaan 6.4.

Dengan demikian, *req*(*p*,*F*) menghasilkan sebuah himpunan {f_1, f_2, f_3 ..}. Untuk suatu $f_i \in \{f_1, f_2, f_3 ..\}$, kebutuhan spesifik dari f_i yang dipenuhi oleh produk, didapatkan melalui Persamaan 6.18 sebagai berikut,

$$reqInd(p,f_i) = \{f\_ind \in allInd(f_i) \mid satF(p,F), f_I \in F\} \qquad (6.18)$$

Dengan demikian, untuk suatu $f_i \in \{f_1, f_2, f_3 ..\}$, *reqInd*(*p*,$f_i$) menghasilkan sebuah himpunan individu kebutuhan fungsional {$f\_ind_{i1}, f\_ind_{i2}, f\_ind_{i3}$ ..}, Komponen-komponen kebutuhan fungsional dalam fasilitas penjelasan ini didapatkan pada saat melakukan proses rekomendasi (lihat Contoh V.2). Pada saat proses rekomendasi ini, sistem dapat menyimpan {f_1, f_2, f_3 ..} maupun {$f\_ind_{i1}, f\_ind_{i2}, f\_ind_{i3}$ ..} ini.

Sebagaimana telah dibahas sebelumnya, $F = Fm \cup Fo$ (dalam *user profile model*)

Nilai *prop*(*p*) adalah properti dari produk *p*, sedangkan *type*(*p*) diperoleh melalui irisan antara *PType* dengan leluhur dari *p*. Secara formal, *type*(*p*) dapat dirumuskan sebagai berikut ,

$$type(p) = PType \cap \{y \in \mathcal{N}_{func} \mid individualOf(p,y)\} \qquad (6.19)$$

Galaxy Core Duos sesuai dengan kebutuhan anda, karena produk ini merupakan handphone, merk Samsung, harga 2.500.000. Produk ini juga memenuhi kebutuhan anda untuk:

- Aktifitas Online, khususnya untuk Video Chatting, Text Chatting, Browsing
- Fotografi, khususnya untuk memotret moment sehari-hari, memotret untuk upload di media sosial
- Bermain game, khususnya untuk Bermain game non HD Secara Online, Bermain game HD Secara Online
- Aktifitas Online, khususnya untuk Video call, Chatting berbasis text, Membuka situs-situs di internet

Gambar 6.13 Contoh penjelasan

Sebagai contoh, *Fm*: **Aktifitas online**, **Fotografi**, *Fo*: **Bermain game**, *Ptype*: **Smartphone**, *Pproperty1*(*Max-harga*): **Rp 3.000.000**, *Pproperty2*(*Brand*): **Samsung**>. Misal **Samsung Galaxy Core Duos** adalah salah satu produk yang direkomendasikan. Contoh penjelasan yang disampaikan ke pengguna ditunujukkan oleh Gambar 6.9.

6.4 *Resume*

Bab ini membahas mengenai model komputasional untuk pembangkitan pertanyaan dalam CRS. Model pembangkitan pertanyaan didasarkan pada eksplorasi relasi semantik pada ontologi. Dengan demikian, *CRS* ini tidak memerlukan mesin inferensi yang terdiri dari sekumpulan *rules*. *Rules* sudah ada dalam struktur ontologi tersebut secara implisit. Dengan model ini, CRS dapat membangkitkan pertanyaan-pertanyaan dan penjelasan yang mengacu pada kebutuhan fungsional, dan dapat memberikan panduan layaknya dalam percakapan antara seorang pelanggan dengan *professional sales support*.

7

EVALUASI

7.1 Pendahuluan

Evaluasi terhadap kinerja CRS, ditinjau dari efisiensi dan efektifitas:

1. Efisiensi
   Efisiensi dari model interaksi, diukur dari kemampuan sistem dalam melakukan *query refinement*, yaitu kemampuan model interaksi dalam mengurangi sisa jumlah *record* dari setiap interaksi.
2. Efektifitas, dengan cara melihat kemampuan model interaksi dalam hal:
   a. Memandu pengguna untuk mengungkapkan kebutuhan
   b. Dilakukan *user study*, dengan menganalisa proporsi pengguna yang memilih *proposed interaction model*, dibandingkan *baseline model*
   c. Meningkatkan persepsi positif pengguna, dibandingkan *baseline model*
   d. Mempengaruhi pengguna untuk mengadopsi model interaksi (sistem)
   e. Dalam evaluasi ini, dianalisa beberapa faktor yang mempengaruhi pengguna untuk mengadopsi model interaksi, menggunakan *Technology Acceptance Model* (TAM).

Evaluasi tentang efisiensi dilihat dari kemampuan sistem dalam melakukan *query refinement*. Cara kerja *query refinement* dalam sistem ini terkait

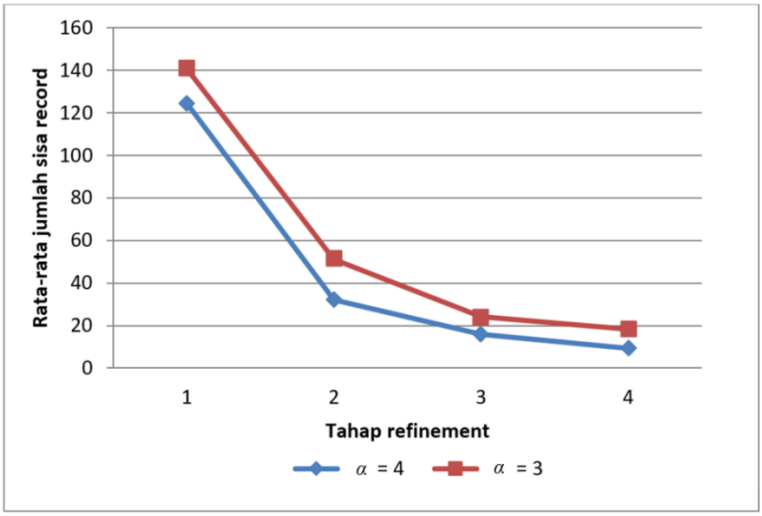
dengan struktur ontologi, algoritma pencarian produk dan model pembangkitan pertanyaan. Evaluasi dari sisi efektifitas di atas, memerlukan sebuah *user study*, dengan melibatkan pengguna sistem sebagai responden. Interaksi berbasis pada kebutuhan fungsional diharapkan mampu memandu pengguna dalam mengungkapkan kebutuhan dan mengambil keputusan, dibandingkan dengan interaksi berbasis pada fitur teknis yang lazim ada dalam situs-situs *e-commerce*. Dengan evaluasi kedua ini, dapat dilihat apakah tujuan utama dari interaksi dalam CRS ini dapat tercapai.

7.2 Efisiensi dari Model Interaksi

Performansi dari sisi efisensi dilihat dari kemampuan model interaksi dalam melakukan *query refinement*. Evaluasi terhadap mekanisme *query* (kebutuhan) *refinement* dilihat dari kemampuan model dalam mengurangi sisa jumlah *record* dari setiap interaksi (Hu dan Aufaure 2013; Mirzadeh dkk., 2005, Shimazu, 2002, McGinty, 2002). Efisiensi tidak dilihat dari jumlah interaksi keseluruhan, karena model interaksi dalam CRS, memungkinkan pengguna untuk memberikan *feedback* terhadap produk yang direkomendasikan, dan memungkinkan pengguna untuk mengubah preferensi di tengah interaksi. Dengan demikian, efisiensi dianalisa dari mekanisme *query refinement*, dilihat dari sisi strategi *navigation by asking* (NBA).

Mekanisme *query refinement* dilakukan oleh algoritma *generateSpecificQ* (Gambar 6.4) untuk penanganan kasus *less_specific* (Subbab 5.2.4.). Algoritma ini akan membangkitkan pertanyaan yang lebih spesifik dan berpotensi untuk disukai pengguna, jika informasi yang diberikan pengguna masih belum cukup rekomendasi produk. Selain algoritma *generateSpecificQ* sendiri, kinerja mekanisme *query refinement* ini sangat terkait dengan desain struktur ontologi, akuisisi pengetahuan dan desain model dari pencarian *recommended products* yang telah dikembangkan. Evaluasi dilakukan berdasarkan simulasi interaksi. Dengan simulasi, dapat deilihat kinerja sistem tanpa terganggu dengan adanya *inconsistency* pengguna dalam pemilihan kebutuhan fungsional. Sebagai contoh *inconsistency*, yaitu jika kebutuhan fungsional *f* yang sebelumnya dipilih sebagai *mandatory requirement*, namun dalam tahap interaksi selanjutnya, semua kebutuhan yang lebih spesifik dari *f* dipilih sebagai *optional requirement*.

Dalam CRS yang dibangun, pada setiap interaksi, sistem mengajukan maksimal sebanyak α pertanyaan. Berdasarkan pada *survey* yang pernah dilakukan terhadap 31 responden, mayoritas responden menyukai α = 4 pertanyaan (41.94%), diikuti 3 pertanyaan (25.81%), 2 pertanyaan (12.90%), 5 pertanyaan (9.68%), 6 pertanyaan (3.23%) dan lebih dari 6 pertanyaan (6.45%). Umur responden yang terlibat dalam *survey*, berkisar antara 18-48 tahun, meliputi responden yang familiar dengan fitur teknis produk dan tidak familiar dengan fitur teknis produk. Skenario dari *survey* yang dilakukan adalah, responden diminta menggunakan aplikasi CRS. Jumlah pertanyaan yang diajukan sistem pada setiap interaksi dalam CRS ini adalah 4 pertanyaan. Pada akhir interaksi, pengguna diberi pertanyaan tentang jumlah opsi pertanyaan yang menurut mereka membantu untuk mengungkapkan kebutuhan. Berdasarkan pada hasil dari survey ini, maka diambil nilai α = 3 dan 4 untuk pengujian tentang mekanisme *query refinement* ini.



Gambar 7.1 Gambar rata-rata jumlah sisa *record* dari masing-masing tahap *refinement*

Untuk evaluasi mekanisme *query refinement* ini, simulasi dilakukan 40 kali, baik dengan α = 3 dan α =4 untuk setiap tahapan interaksi. Pada setiap tahap interaksi, dihitung rata-rata sisa jumlah *record* dari algoritma pencarian (rekomendasi) produk. *Trend* dari penurunan rata-rata jumlah *record* akan dilihat untuk maksimum 4 tahap interaksi. Pada setiap tahap interaksi, user memilih sebanyak α *mandatory requirement*. Hasil dari

percobaan dapat dilihat pada gambar 7.1. Dari hasil ini, baik dengan α = 3 maupun 4, mekanisme *query refinement* cukup efisien, dilihat dari trend pengurangan sisa jumlah *record* yang signifikan, dalam 4 tahap interaksi (jumlah produk = 288).

Dari hasil pengujian, jika dilihat untuk α = 4, pada interaksi ke 4, jumlah sisa *record* sebesar rata-rata 15 produk dari 288 produk, yang berarti dalam 4 interaksi, mampu mengurangi sebesar 5,2% dari seluruh *record* produk. Sementara itu, Mirzadeh dkk. (2005), mampu mengurangi 7,3% dari seluruh *record* produk. Selain itu, dalam *recommender system* yang dikembangkan oleh Shimazu (2002), dalam 4 interaksi mampu mengurangi 16,7%, sedangkan pada Hu dan Aufaure (2013) mampu mengurangi 25% dari total *record* produk pada interaksi ke 4.

7.3 Efektifitas Model Interaksi dalam Memandu Pengguna

Evaluasi dari efektifitas interaksi, didekati dengan menganalisa proporsi pengguna yang memilih *func-based model*, dibandingkan *tech-based model*, ketika mereka mencari produk. Hasil evaluasi akan dianalisis berdasarkan dua kategori pengguna: 1) familiar dengan fitur teknis (*expert user*), 2) tidak familiar dengan fitur teknis (*novice user*). Kuesioner untuk penentuan kategori ini terdapat pada lampiran B. Dalam kuesioner tersebut, disajikan 3 definisi terkait familiaritas pengguna terhadap fitur teknis produk, yaitu:

1. *Familiar*: Familiar dengan sebagian besar spesifikasi teknis *smartphone*
2. *Agak Familiar*: Familiar dengan sebagian kecil spesifikasi teknis *smartphone*
3. *Tidak Familiar*: Tidak familiar sama sekali dengan spesifikasi teknis *smartphone*

   Pembagian berdasarkan 3 definisi pada saat diajukan ke penguna, bertujuan untuk memudahkan pengguna untuk memilih kategori sesuai dengan keadaaan pengguna saat itu. Untuk keperluan analisa, kategori *agak familiar* dan *tidak familiar* digabung sebagai kategori *novice user*, sedangkan kategori *familiar* sebagai *expert user*. Total responden berjumlah 60 orang, dengan 30 orang untuk masing-masing kategori (*expert* dan *novice user*). Umur pengguna berkisar antara 18-55 tahun, serta mereka merupakan pengguna *smartphone* dan familiar dengan komputer.

Dalam pengujian ini, CRS berbasis kebutuhan fungsional yang dikembangkan (selanjutnya disebut sebagai ***func-based model***) akan dibandingkan dengan baseline, yaitu sebuah CRS berdasarkan fitur teknis produk (selanjutnya disebut sebagai ***tech-based model***). Dipilih *tech-based model* ini sebagai *baseline* karena mewakili sebuah CRS berbasis fitur teknis produk yang paling sederhana, tetapi menangani kasus-kasus interaksi yang ada dalam *func-based model*.

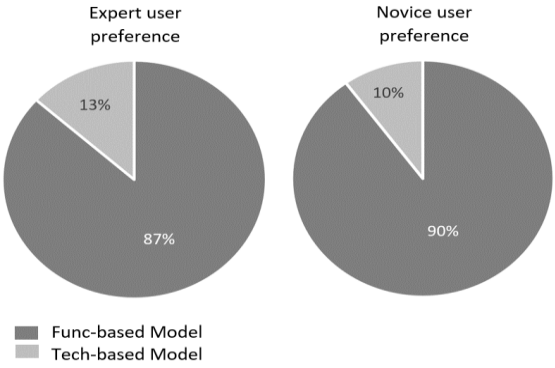
Tabel 7.1 Perbedaan strategi dari kedua model interaksi

Kasus: *empty_profile*	*Func-based model*: sistem memberikan pertanyaan-pertanyaan yang terkait dengan kebutuhan fungisional dari *smartphone*, tetapi masih bersifat umum
	Tech-based model: sistem memberikan pertanyaan-pertanyaan tentang fitur teknis *smartphone*
Kasus: *multi_chos_prod*	*Func-based model*: sistem menunjukkan pembandingan antara beberapa produk yang dipilih oleh pengguna, untuk membantu pengguna membuat keputusan
	Tech-based model: sistem memunculkan kembali produk-produk yang pengguna pilih sebelumnya, beserta fitur teknisnya
Kasus: *no_chos_prod*	*Func-based model*: sistem menanyakan kebutuhan fungsional yang potensial diinginkan oleh pengguna
	Tech-based model: sistem menanyakan kembali kebutuhan-kebutuhan (fitur teknis) seperti sebelumnya. Kebutuhan yang sebelumnya diinginkan pengguna, ditandai.
Kasus: *less_specific*	*Func-based model*: sistem memberikan pertanyaan kebutuhan fungsional yang lebih spesifik dari pada pertanyaan sebelumnya, dan potensial untuk disukai pengguna
	Tech-based model: sistem menampilkan semua produk yang memenuhi kebutuhan pengguna. Pengguna dapat memberikan kebutuhan yang lebih spesifik berdasarkan fitur teknis produk
Kasus: *contra_pref*	*Func-based model*: sistem memberikan pertanyaan yang berisi *contradictory subset* dari kebutuhan pengguna, yang menyebabkan tidak ada produk yang direkomendasikan

	Tech-based model: sistem memberikan pertanyaan-pertanyaan kembali seputar fitur teknis produk. Kebutuhan sebelumnya yang telah pengguna pilih, ditandai.
Fasilitas Penjelasan	*Func-based model*: sistem menyajikan penjelasan tentang mengapa produk direkomendasikan berdasarkan kebutuhan fungsional
	Tech-based model: sistem memberikan penjelasan berdasarkan fitur teknis dari masing-masing produk yang direkomendasikan

Demi *fairness*, kedua sistem (*func-based* dan *tech-based model*) mempunyai antar muka yang identik, data produk yang sama, dan hanya berbeda dalam strategi interaksi untuk mendapatkan kebutuhan pengguna. Tabel 7.1 menunjukkan perbedaan strategi interaksi dari kedua sistem ini. Penjelasan lengkap tentang perbedaan strategi antara kedua sistem (model interaksi) beserta antar muka-nya disajikan dalam Lampiran D.

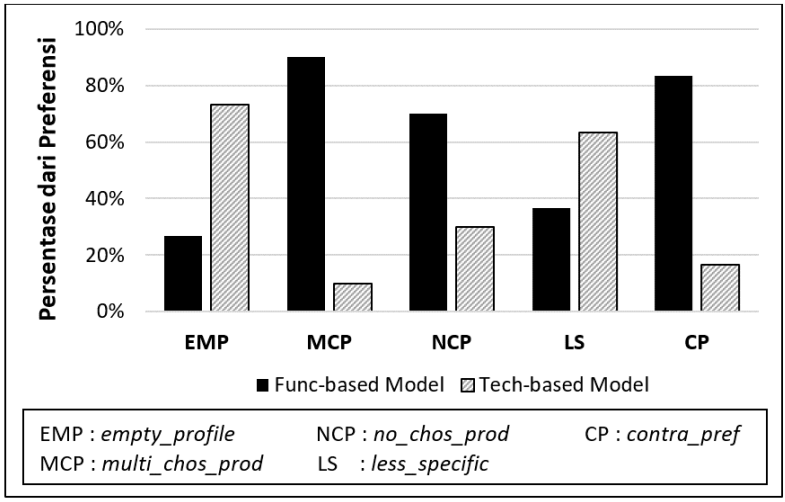
Dalam *user study*, pengguna diminta berinteraksi dengan kedua sistem, untuk mengungkapkan kebutuhannya secara bebas, sehingga pengguna dapat melakukan eksplorasi interaksi sistem secara keseluruhan. Setelah itu, pengguna diminta mengisi kuesioner, apakah dia menyukai *func-based model* atau model *tech-based model*. Sebagai tambahan evaluasi, pengguna diminta untuk mencoba per-kasus interaksi (dalam Tabel 7.1). Untuk evaluasi ini, beberapa *task* yang harus dilakukan pengguna dirancang secara khusus, sehingga pengguna merasakan pengalaman pada setiap kasus interaksi, untuk kedua sistem (model interaksi). Survey ini tidak mempertimbangkan emosi pengguna, khususnya setelah menggunakan sistem ini berkali-kali. Hal yang dipertimbangkan dalam melakukan *user study* ini adalah kemudahan bagi pengguna, dan tidak memberikan beban berlebih bagi pengguna sebagai responden, dalam memberikan pendapatnya tentang sistem yang diuji coba.



Gambar 7.2 Preferensi pengguna terhadap *func-based model* vs *tech-based model* untuk keseluruhan interaksi

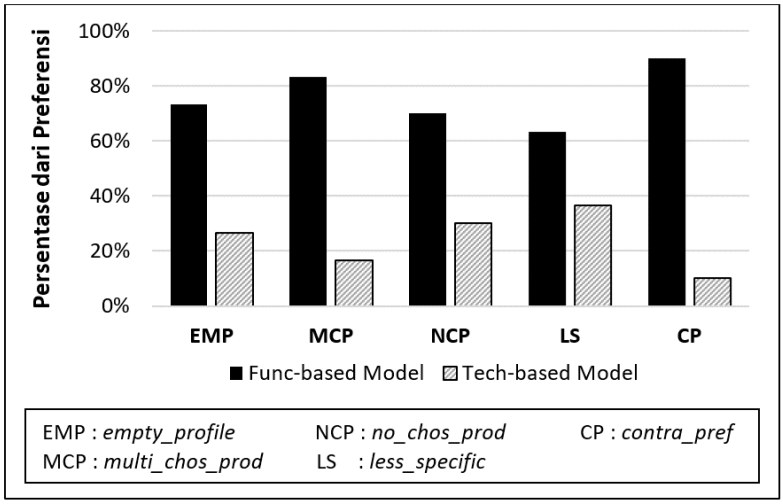
Preferensi pengguna untuk keseluruhan interaksi ditunjukkan pada Gambar 7.4. Dari hasil ini dapat dilihat bahwa semua pengguna menyukai interaksi pada *func-based model*. Hal yang menarik terjadi pada *expert users*. Hal ini menunjukkan, keseluruhan interaksi dalam *func-based model* sangat membantu pengguna dalam membeli sebuah produk.

Gambar 7.2 dan 7.3 menunjukkan preferensi pengguna terhadap *func-based model* vs *tech-based model* untuk setap kasus interaksi. Dari hasil ini, dapat dilihat bahwa 3 dari 5 kasus ; kasus *multi_chos_prod* (pengguna memilih lebih dari satu produk), *no_chos_prod* (pengguna tidak memilih satu produk pun), dan *contra_pref* (tidak ada produk yang sesuai dengan kebutuhan pengguna), baik *novice user* maupun *expert user* lebih menyukai interaksi pada *func-based model*.



Gambar 7.3 Preferensi *expert user* terhadap *func-based model* vs *tech-based model*

Dalam ketiga kasus tersebut, sistem memberikan panduan berupa pembandingan terkait kebutuhan fungsional khas dari masing-masing produk (kasus *multi_chos_prod*), sistem memberikan pertanyaan kembali tentang kebutuhan fungsional yang potensial disukai pengguna (kasus *no_chos_prod*), dan sistem memunculkan kebutuhan-kebutuhan yang membuat tidak ditemukannya produk yang sesuai (kasus *contra_pref*). Panduan untuk ketiga jenis kasus ini ternyata lebih disukai *expert user*, daripada *tech-based model*. Sementara itu, *novice user* lebih menyukai *func-based model* untuk semua kasus interaksi. Hal ini menunjukkan bahwa, dalam kasus-kasus ini, *func-based model* mempunyai strategi yang lebih baik dalam memandu pengguna dalam mengungkapkan kebutuhan dan mengambil keputusan daripada *tech-based model*.



Gambar 7.4 Preferensi Novice User Terhadap Func-based Model vs Tech-based Model

Hal yang menarik adalah, pada kasus *empty_profile* (*user profile model* kosong) dan kasus *less_specific* (kebutuhan pengguna belum cukup untuk rekomendasi produk), preferensi dari *expert users* berbeda dengan *novice users*. Dalam kasus ini, mayoritas *expert users* menyukai *tech-based model*, sedangkan sebagian besar *novice users* menyukai *func-based model*. Hal ini dapat dipahami, karena *expert users* lebih familiar dengan fitur teknis, sehingga mereka lebih cenderung ingin secara langsung mengungkapkan kebutuhannya berdasarkan fitur teknis, baik pada awal interaksi maupun pada saat memberikan kebutuhan secara spesifik.

7.4 Efektifitas Model Interaksi dalam Mempengaruhi Persepsi Pengguna

Evaluasi ini terkait dengan persepsi pengguna terhadap model interaksi. Evaluasi dilakukan dengan mengadopsi evaluasi persepsi pengguna terhadap *recommender system* yang dilakukan Zanker dkk. (2012). Terdapat dua tujuan dalam evaluasi ini:

1. Menganalisis pengaruh *functional requirements – based interaction model* terhadap beberapa faktor, meliputi PU, EOU, BI, PE dan TR

(dijelaskan pada halaman 94), dibandingkan dengan model interaksi yang berbasis pada fitur teknis produk

2. Menganalisis faktor-faktor yang mempengaruhi adopsi pengguna terhadap model interaksi ini (dengan mempertimbangkan semua faktor/*construct* yang telah disebutkan di atas)
   Evaluasi dilakukan dengan memanfaatkan metode *Technology Acceptance Model* (TAM), khususnya untuk point butir ke-2 di atas. *Technology Acceptance Model* (TAM) adalah sebuah model untuk menjelaskan atau memprediksi penerimaan dan adopsi pengguna dan terhadap teknologi informasi baru, berdasarkan persepsi pengguna (Davis, 1993). Dalam TAM, faktor-faktor ini disebut sebagai *construct*, yang juga merupakan suatu bentuk dari persepsi pengguna terhadap sebuah sistem. Dengan suatu model hipotesis, dapat dilihat pengaruh antar *constructs* ini.

Metodologi untuk melakukan evaluasi adalah sebagai berikut,

1. Ditentukan *construct* yang akan dilibatkan dalam pengujian. *Construct* ini menentukan kuesioner yang akan dibuat
2. Dibuat kuosioner berdasarkan *constructs* yang akan diuji
3. Dilakukan *user study*. Responden mencoba sistem dan mengisi kuosioner
4. Pengujian validasi terhadap item-item pertanyaan dalam kueosioner berdasarkan hasil *user study* (menggunakan *principal component analysis*)
5. Disusun model hipotesis, dengan uji kesesuain model dengan data.
6. Dianalisa hasil uji hipotesis

Dalam pengujian ini, *construct* yang diobservasi meliputi,

- *Perceived Usefulness* (PU): sejauh mana pengguna merasa bahwa model interaksi ini cukup berguna dalam menyelesaikan permasalahan (Davis, 1993)
- *Perceived ease of use* (EOU): sejauh mana pengguna merasa bahwa menggunakan sistem ini akan free of effort (Davis, 1993)
- *Behavioral Intention* (BI): sejauh mana pengguna berniat untuk menggunakan model interaksi ini nantinya, khususnya unuk permasalahan pencarian produk (Davis, 1993)

Tabel 7.2 *Constructs* dan Item-itemnya

Kode	Perceived Usefulness (PU)	
PU1	Model Interaksi ini mampu meningkatkan kualitas pencarian produk yang saya lakukan	Zanker dkk. (2012)
PU2	Model interaksi ini menawarkan banyak keunggulan fitur dalam proses pencarian produk	
PU3	Model interaksi ini lebih menghemat waktu saya dalam pencarian produk	
PU4	Model interaksi ini mempermudah saya untuk menemukan produk yang sesuai	
PU5	Model interaksi dalam sistem ini mampu memberikan hasil pencarian yang lebih baik	
PU6	Sistem dengan model interaksi ini membantu dalam proses pencarian produk	
PU7	Secara keseluruhan, model interaksi ini sangat berguna dalam proses pencarian produk	
	Perceived ease of use (EOU)	
EOU1	Sistem dengan model interaksi ini mudah digunakan	Brooke (1996)
EOU2	Model interaksi ini tidak membutuhkan banyak upaya yang berat	
EOU3	Menurut saya, sistem dengan model interaksi ini juga mudah digunakan orang lain	
EOU4	Setiap tahap interaksi cukup jelas dan mudah dipahami	
EOU5	Saya cepat menyesuaikan dengan menu-menu yang ada dalam sistem	
	Trust (TR)	
TR1	Saya percaya dengan rekomendasi produk yang diberikan oleh sistem	Zanker dkk. (2012)
TR2	Saya percaya dengan penjelasan yang diberikan CRS, tentang mengapa sebuah produk direkomendasikan	

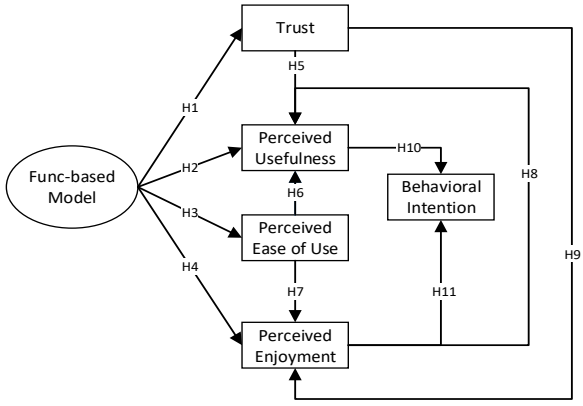
TR3	Saya percaya bahwa urutan produk yang direkomendasikan memang sesuai dengan kebutuhan saya	
TR4	Fasilitas penjelasan yang diberikan oleh sistem membuat saya percaya bahwa produk yang direkomendasikan sesuai dengan kebutuhan saya	
	Perceived Enjoyment (PE)	
PE1	Interaksi dalam sistem ini menarik	Liao dkk. (2008)
PE2	Saya merasa nyaman dengan interaksi dalam sistem ini	
PE3	Proses interaksi dalam sistem cukup menyenangkan	
PE4	Interaksi dalam CRS ini membuat saya merasa terpandu	
	Behavioral Intention (BI)	
BI1	Jika suatu saat saya penya akses terhadap sistem ini, saya berniat untuk menggunakan sistem ini untuk pencarian produk handphone	Liao dkk. (2008)
BI2	Saya akan menggunakan sistem dengan model interaksi ini di kemudian hari	
BI3	Saya akan merekomendasikan sistem ini ke orang lain juga	

- *Perceived Enjoyment* (PE): sejauh mana pengguna merasa tertarik, nyaman, dan terpandu dengan model interaksi yang ditawarkan (Liao dkk., 2008)
- *Trust* (TR): sejauh mana pengguna percaya terhadap rekomendasi yang diberikan sistem, dengan fasilitas penjelasan memegang peranan dalam hal ini (Pavlou, 2003)

Setiap *construct* terdiri dari beberapa item *construct*, yang akan menjadi daftar pertanyaan untuk pengguna, seperti disajikan dalam Tabel 7.2. Jawaban-jawaban dikodekan dalam 5-point *Likert scale* yang memiliki rentang dari sangat tidak setuju (1) sampai sangat setuju (5). Demi *fairness*,

pertanyaan-pertanyaan 1-23 tersebut disusun secara acak, dan tidak dikelompokkan berdasarkan masing-masing *construct*-nya. Kuesioner disajikan secara lebih lengkap dalam Lampiran F.

Untuk menganalisis sejauh mana model interaksi berbasis kebutuhan fungsional dapat meningkatkan persepsi positif pengguna dan pengaruhnya terhadap ketertarikan pengguna untuk mengadopsi model interaksi ini, perlu dibuat sebuah model hipotesis. Namun, sebelumnya perlu dilakukan pengujian kesesuaian model hipotesis dengan data. Untuk menguji kesesuaian model hipotesis ini, digunakan beberapa parameter yang dihasilkan oleh LISREL *path analysis*. Liao, dkk. (2008) menyatakan bahwa model dapat dikatakan sesuai dengan data jika *goodness of fit index* (GFI)>0.8 dan *root mean square error of approximate* (*RMSEA*) < 0.08. Sementara itu, Al-Maghrabi dkk. (2011) menyatakan bahwa *chi-square/df* (*CMIN*/*DF*) <= 5 mengindikasikan model sesuai dengan data.



Gambar 7.5 Model hipotesis

Dari parameter-parameter uji kesesuaian model ini, dan juga berdasarkan studi literatur terkait pembentukan model hipotesis, didefinsikan 11 hipotesis sebagai berikut,

- H1 : *Functional requirements-based interaction meningkatkan trust*
- H2 : *Functional requirements-based interaction meningkatkan perceived usefulness*

- H3 : *Functional requirements-based interaction meningkatkan ease of use*
- H4 : *Functional requirements-based interaction meningkatkan trust*
- H5 : *Perceived trust mempengaruhi perceived usefulness*
- H6 : *Perceived ease of use mempengaruhi perceived usefulness*
- H7 : *Perceived ease of use mempengaruhi perceived enjoyment*
- H8 : *Perceived enjoyment mempengaruhi usefulness*
- H9 : *Perceived ease of use mempengaruhi perceived usefulness*
- H10 : *Perceived usefilness of use mempengaruhi behavioral intention*
- H11 : *Perceived ease of use mempengaruhi behavioral intention*

Model hipotesis ini ditunjukkan pada Gambar 7.5. Hasil dari *user study*, untuk beberapa parameter kesesuaian model dengan data, didapatkan GFI = 0.93, RMSEA = 0.076, *CMIN*/*DF* = 1.58. Hasil ini menunjukkan bahwa model hipotesis sesuai dengan data.

Zanker (2012) telah membuktikan bahwa fasilitas penjelasan pada *recommender system* dapat meningkatkan *perceived usefulness*, *perceived ease of use* dan *trust*. Tintarev and Masthof (2007) juga mengungkapkan, bahwa fasilitas penjelasan dalam *recommender system*, harus mampu meningkatkan *trust* pengguna terhadap produk yang direkomendasikan maupun sistem itu sendiri. Untuk itu, untuk beberapa pertanyaan terkait dengan *trust*, kami mengaitkannya dengan fasilitas penjelasan yang disajikan oleh sistem. Dalam hal ini, akan dievaluasi pengaruh *func-based model* untuk peningkatan *perceived usefulness*, *perceived ease of use*, *trust* dan *perceived enjoyment* (H1 – H4).

Beberapa penelitian telah menunjukkan terdapat pengaruh positif antara *trust* dan *perceived usefulness* (Pavlou 2003; Al-Maghrabi dkk., 2011; Wu dkk., 2011), sehingga akan dievaluasi pengaruh antara dua *construct* ini (H5). Sementara itu, *perceived usefulness*, *perceived ease of use* dan *behavioral intention*, merupakan *constructs* ini dalam TAM, dan hubungan antara ketiganya juga telah dipelajari dalam banyak penelitian tentang TAM, sejak diperkenalkan oleh Davis dan Warshaw (1989). Dengan demikian, hipotesis H6 dan H10 perlu dibuat. *Perceived enjoyment* mempunyai pengaruh besar terhadap behavioral intention, dalam area *e-commerce* (Al-Maghrabi dkk., 2011; Koufaris, 2002; Wen dkk., 2011). Dengan demikian, dalam evaluasi ini, *perceived enjoyment* terlibat dalam analisis (H7 – H11).

Untuk menguji H1 – H11, terlibat sebanyak 100 pengguna, yang dikelompokkan dalam kategori familiar (*expert user*) sebesar 51% dan tidak familiar dengan fitur teknis produk (*novice user*) sebesar 49%. Semua responden adalah pengguna smartphone dan familiar dengan komputer. Kuesioner untuk penentuan kategori pengguna ini, selengkapnya terdapat pada lampiran C. Dalam kuesioner tersebut, kategori pengguna didefinisikan dalam 4 kelompok, untuk lebih mempermudah pengguna dalam mengkategorikan dirinya, yaitu:

1. *Familiar*: Familiar dengan semua spesifikasi teknis *smartphone*
2. *Cukup Familiar*: Familiar dengan sebagian besar spesifikasi teknis *smartphone*
3. *Kurang Familiar*: Familiar dengan sebagian kecil spesifikasi teknis *smartphone*
4. *Tidak Familiar*: Tidak familiar sama sekali dengan spesifikasi teknis *smartphone*

   Empat definisi kategori pengguna merupakan revisi dari 3 definisi kategori pengguna yang ada dalam *user study* sebelumnya. Selanjutnya, untuk keperluan analisa, kategori *kurang familiar* dan *tidak familiar* digabung sebagai kategori *novice user*, sedangkan *cukup familiar* dengan *familiar* digabung ke dalam kategori *expert user*.

Dalam pengujian H1-H4, responden diminta untuk mencoba interaksi baik dalam *func-based model*, maupun *tech-based model*. Kemudian responden diminta untuk mengisi kuesioner yang berisi pertanyaan-pertanyaan pada Tabel VII.2. Untuk mengevaluasi H5-H11, evaluasi yang dilakukan hanya fokus pada jawaban-jawaban pengguna terhadap *func-based model*, serta memanfaatkan analisis regresi linier dengan formula sebagai berikut :

$$PU = a + b_5(TR) + b_6(EOU) + b_8(PE) + \varepsilon \qquad (7.1)$$

$$PE = a + b_9(TR) + b_7(EOU) + \varepsilon \qquad (7.2)$$

$$BI = a + b_{10}(PU) + b_{11}(PE) + \varepsilon \qquad (7.3)$$

Pada tahap awal, dilakukan pengujian validitas terhadap 23 *construct items* yang ditanyakan kepada responden. Dengan menggunakan *principal component analysis* (metode *varimax rotation* dengan Kaiser *normalization*), 23 *construct item* tereduksi menjadi 11 item, seperti ditunjukkan pada Tabel 7.3. Masing-masing faktor berkorespondensi dengan masing-masing construct. Faktor 4 (TR/*trust*), berkorelasi dengan item TR1, TR2 dan TR3. Namun TR2 juga berkorelasi dengan faktor 1 (PE/*perceived enjoyment*), dengan demikian, item TR2 dapat juga dinterpretasikan

mendukung PU. Karena nilai *factor loading* TR2 untuk faktor TR dan PU tidak berselisih besar, jadi dalam evaluasi ini, TR2 dimasukkan dalam kelompok TR (*Trust*) (Zanker, 2012). Zanker (2012) menyatakan bahwa nilai *factor loading* lebih besar dari 0.40 sudah cukup merepresentasikan "*acceptable*" untuk dalam konteks analisis faktor.

Seperti dapat dilihat pada Tabel 7.3, semua nilai *factor loading* lebih besar daripada 0.40. Hasil dari uji reliabilitas dinyatakan sebagai nilai *Cronbach Alpha* dari masing-masing *construct*. O'Rourke dkk. (2013) [28] menyatakan bahwa nilai *Cronbach Alpha* lebih besar dari 0.50 adalah *sufficient* untuk keperluan riset, sedangkan 0.70 adalah *recommended*, sedangkan 0.80 adalah *desirable*. Dalam Tabel 7.3, semua nilai *Cronbach Alpha* lebih besar dari 0.60, bahkan sebagian besar di atas 0.70 maupun 0.80.

Tabel 7.3 Item-Item *construct* yang ter-ekstrak

	Component				
	Factor 1 (PE)	Factor 2(PU)	Factor 3(BI)	Factor 4 (TR)	Factor 5 (EOU)
PU1	0,32	**0,72**	0,20	0,17	0,19
PU3	0,26	**0,64**	0,49	0,03	0,23
PU7	0,49	**0,55**	0,29	0,13	0,22
Crobanch Alpha of PU = 0,83					
EOU2	0,20	0,24	0,24	0,06	**0,75**
EOU5	0,22	0,36	0,32	0,35	**0,44**
Crobanch Alpha of EOU = 0,60					
TR1	0,34	0,03	0,16	**0,80**	0,12
TR2	0,09	**0,57**	0,01	**0,50**	0,42
TR3	0,25	0,34	0,26	**0,58**	0,16
Crobanch Alpha of TR = 0,77					
PE1	**0,69**	0,32	0,32	0,11	0,22
PE4	**0,60**	0,28	0,43	0,24	0,09
Crobanch Alpha of PE = 0,78					
BI2	0,18	0,28	**0,69**	0,35	0,20

- **Pengaruh Model Interaksi Dalam Meningkatkan Persepsi Positif Pengguna (PU, EOU, BI, TR, PE)**

Evaluasi dari pengaruh model interaksi terhadap persepsi pengguna, dapat dilihat dari hasil dari pengujian H1 – H4, yang ditunjukkan pada Tabel 7.4. Dari hasil ini dapat dilihat bahwa untuk *expert user*, rata-rata rating pengguna untuk *func-based model* lebih besar daripada *tech-based model*,

untuk faktor *perceived usefulness*, *perceived ease of use*, *trust* dan *perceived enjoyment*. Untuk menguji perbedaan rata-rata *rating* dari dua model ini, digunakan *T-test* dengan melibatkan *Levene's Test for Equality* of *Variances*.

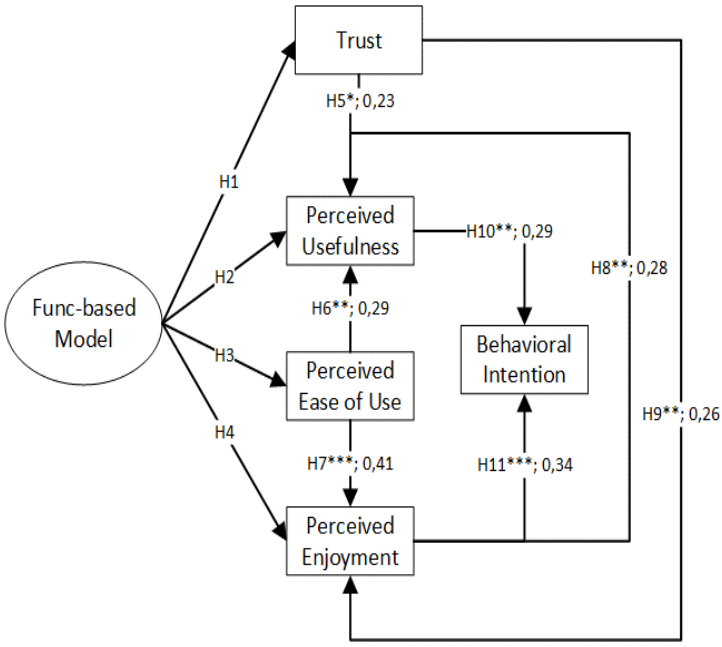
Hasil *T-test* dalam Tabel 7.4 menunjukkan bahwa *functional based interaction model* dalam *func-based model* berpengaruh secara signifikan untuk meningkatkan persepsi pengguna dari keempat faktor ini ($p < 0.001$). Hal ini menunjukkan bahwa H1, H2, H3 dan H4 diterima. Dari Tabel 7.4 dapat dilihat bahwa rata-rata dari rating pengguna untuk *func-based model* lebih tinggi daripada *tech-based model*, untuk keempat faktor ini. *T-test* membuktikan bahwa rata-rata rating pengguna ini berbeda secara siginifikan untuk keempat faktor, sehingga H1, H2, H3 dan H4. Dengan demikian dapat disimpulkan, bahwa *func-based model* mampu meningkatkan persepsi pengguna dari sisi *perceived usefulness* (PU), *perceived ease of use* (EOU), *trust* (TR) and *perceived enjoyment* (PE), baik untuk pengguna yang familiar dengan fitur teknis maupun yang tidak familiar dengan fitur teknis.

Tabel 7.4 Rata-rata rating pengguna and hasil *T-Test*

Factor	***Model***	***Expert user***		***Novice user***	
		Mean	**Sig (2-tailed)**	***Mean***	***Sig (2-tailed)***
PU	Func-based Model	**4,63**	0,00	**4,49**	0,00
	Tech-based Model	3,69		3,42	
EOU	Func-based Model	**4,30**	0,00	**4,04**	0,00
	Tech-based Model	3,34		3,19	
TR	Func-based Model	**4,21**	0,00	**4,14**	0,00
	Tech-based Model	3,68		3,40	
PE	Func-based Model	**4,27**	0,00	**4,17**	0,00
	Tech-based Model	3,59		3,25	

- **Faktor-Faktor yang Mempengaruhi Adopsi Pengguna Terhadap Model Interaksi**

Pertama-tama, dilakukan analisis *hypotheses paths* dengan tanpa melihat kategori pengguna. Hasil dari uji-uji hipotesis dapat dilihat pada Gambar 7.6. Masing-masing *edge* menunjukkan koefisen regresi (*b*) dari masing-masing variabel bebas (lihat Persamaan 7.1 – 7.3). Nilai dari *b* menyatakan derajat dari pengaruh sebuah faktor ke faktor lain. *Trust* dan *perceive ease of use* mempengaruhi *perceived enjoyment* dengan derajat yang hampir sama (H7 dan H9). Dengan demikian, fasilitas penjelasan (*trust*) dan kemudahan dalam *func-based model* mampu membuat pengguna merasa terpandu dan tertarik. Sementara itu, *trust*, *perceived ease of use* dan *perceived enjoyment* juga merupakan faktor yang mempengaruhi *perceived usefulness* (H5, H6 dan H8). Selain itu, dapat dilihat bahwa *perceived ease of use* merupakan faktor yang lebih kuat mempengaruhi *perceived enjoyment* daripada *trust* (H7, H9).



* : signifikan dengan p < 0,05

** : signifikan dengan p < 0,01

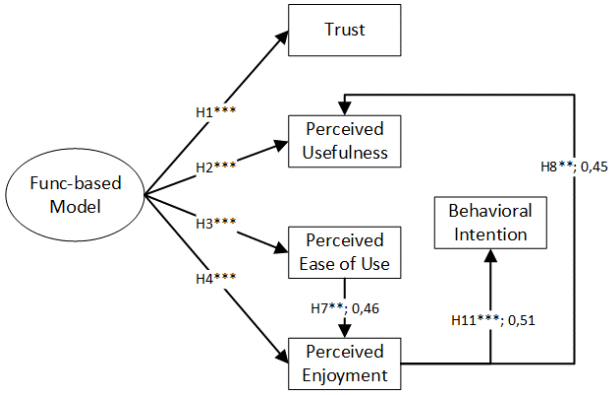
*** : signifikan dengan < 0,001

Gambar 7.6 *Paths* dari Hipotesis yang diterima untuk keseluruhan pengguna

Behavioral intention (niat pengguna untuk mengadopsi model interaksi) dipengaruhi oleh *perceived usefulness* (H10) dan *perceived enjoyment* (H11). Dalam kasus ini, *perceived enjoyment* mempunyai pengaruh yang lebih kuat daripada *perceived usefulness*. Dapat dilihat juga, *perceived ease of use* mempunyai peranan besar dalam meningkatkan *perceived enjoyment*, sementara *perceived enjoyment* mempunyai pengaruh yang kuat dalam meningkatkan niat pengguna untuk mengadopsi model interaksi. Secara umum dapat disimpulkan, bahwa *trust*, *perceived ease of use* dan *perceived enjoyment* berpengaruh secara langsung untuk meningkatkan niat pengguna untuk mengadopsi model interaksi, sedangkan *perceived usefulness* dan *perceived enjoyment* mempunyai pengaruh secara langsung, namun *peceived enjoyment* mempunyai pengaruh yang lebih kuat.

Selanjutnya akan dilakukan uji hipotesis untuk *expert user*. *Path* uji hipotesis yang diterima untuk kategori pengguna ini, disajikan oleh gambar 7.7.a. Parameter uji kesesuaian model hipotesis dengan data, menunjukkan *CMIN/DF* = 0.93, RMSEA = 0.00 dan GFI = 0.961. Hasil ini menunjukkan bahwa model hipotesis sesuai dengan data. Sebagaimana ditunjukkan pada gambar ini, *perceived ease of use* berpengaruh besar pada *perceived enjoyment* (H7). Hal ini menunjukkan bahwa kemudahan yang disajikan *func-based model* mampu membuat pengguna merasa tertarik dan terpandu (*perceived enjoyment*).

Sementara itu, *perceived enjoyment* berpengaruh terhadap *behavioral intention* dan *perceived usefulness* (H11, H8). Hasil ini menunjukkan bahwa *expert user* tertarik untuk mengadopsi *func-based model* dipengaruhi oleh faktor kemudahannya (secara tidak langsung) dan kemampuan *func-based model* dalam memandu pengguna (*perceived enjoyment*). Namun, *trust* dan *perceived usefulness* bukan merupakan faktor yang mempengaruhi *expert user* untuk mengadopsi *func-based model*. Fasilitas penjelasan yang mengacu pada kebutuhan fungsional produk tidak begitu menarik bagi pengguna kategori ini, karena mereka sudah familiar dengan fitur teknis, sehingga sudah sangat paham terhadap kaitan antara produk dengan fitur teknisnya. Namun pengguna merasa terpandu dengan semua interaksi dalam *func-based model*, sehingga mereka tertarik untuk menggunakan model interaksi di waktu yang akan datang (*behavioral intention*)

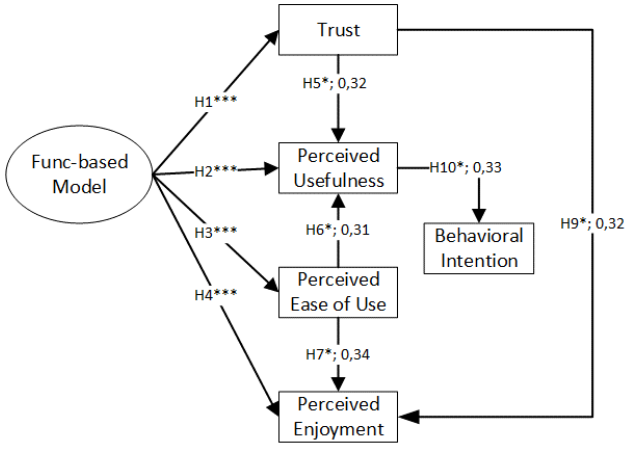


(a)

* : signifikan dgn p < 0.05 ** : signifikan dgn p < 0.01

*** : signifikan dgn p < 0.001

Gambar 7.7 *Paths* dari hipotesis yang diterima untuk *expert users*



* : signifikan dgn p < 0,05 ** : signifikan dgn p < 0,01

*** : signifikan dgn p < 0,001

Gambar 7.8 *Paths* dari hipotesis yang diterima untuk *novice users*

Hypotheses Path untuk *novice user*, ditunjukkan oleh gambar 7.7.b. Dari uji kesesuaian model hipotesis dan data, diperoleh *CMIN*/*DF* = 0.82, *RMSEA* = 0.00 dan GFI = 0.941. Hasil uji hipotesis menunjukkan bahwa *perceived ease of use* dan *trust* berpengaruh terhadap *perceived usefulness* (menerima H5 and H6), dan *perceived usefulness* ini berpengaruh terhadap *behavioral intention* (menerima H9). Dengan demikian, dengan *func-based model*, pengguna merasa terbantu dikarenakan faktor kemudahan serta *trust*. Kedua faktor ini secara tidak langsung mempengaruhi pengguna untuk tertarik menggunakan model interaksi ini di kemudian hari. Ini berarti, fasilitas penjelasan berdasarkan kebutuhan fungsional ini cukup membantu pengguna, dan membuat pengguna tertarik. Selain itu, *trust* dan *perceived ease of use* berpengaruh terhadap *perceived enjoyment* (menerima H7 dan H9). Dengan demikian, faktor fasilitas penjelasan serta kemudahan penggunaan dalam interaksi ini, yang membuat pengguna merasa terpandu (*perceived enjoyment*).

7.5 *Resume*

Evaluasi dari CRS meliputi pengujian yang bersifat obyektif (*query refinement*) dan subyektif (efektifitas model interaksi dalam memandu pengguna, pengaruh model interaksi terhadap persepsi pengguna). Dari hasil pengujian kemampuan sistem dalam *query refinement*, sistem mampu mengurangi jumlah sisa *record* secara signifikan dalam 4 interaksi. Dari hasil evaluasi efektifitas model interaksi, pengguna yang familiar dengan fitur produk (*expert user*), dalam sebagian besar kasus interaksi, lebih menyukai *func-based model* daripada *tech-based model*, namun secara keseluruhan interaksi, mereka menyukai lebih menyukai *func-based model*. Sedangkan sebagian besar pengguna yang kurang familiar dengan fitur teknis (*novice user*), lebih menyukai *func-based model* (daripada *tech-based model*) di setiap kasus interaksi, maupun keseluruhan interaksi.

Untuk evaluasi pengaruh model interaksi terhadap persepsi pengguna, secara umum, *func-based model* mampu meningkatkan *perceived ease of use*, *perceived enjoyment*, *trust* dan *perceived usefulness* baik untuk *expert user* maupun *novice user*. Sementara itu, terdapat perbedaan dalam hal faktor-faktor yang mempengaruhi adopsi sistem, antara kedua kategori pengguna. Untuk *expert user*, *trust* bukan merupakan faktor yang mempengaruhi adopsi pengguna terhadap model. Ini dikarenakan mereka

sudah faham dengan fitur teknis, sehingga penyajian fasilitas penjelasan berdasarkan kebutuhan fungsional tidak begitu menarik.

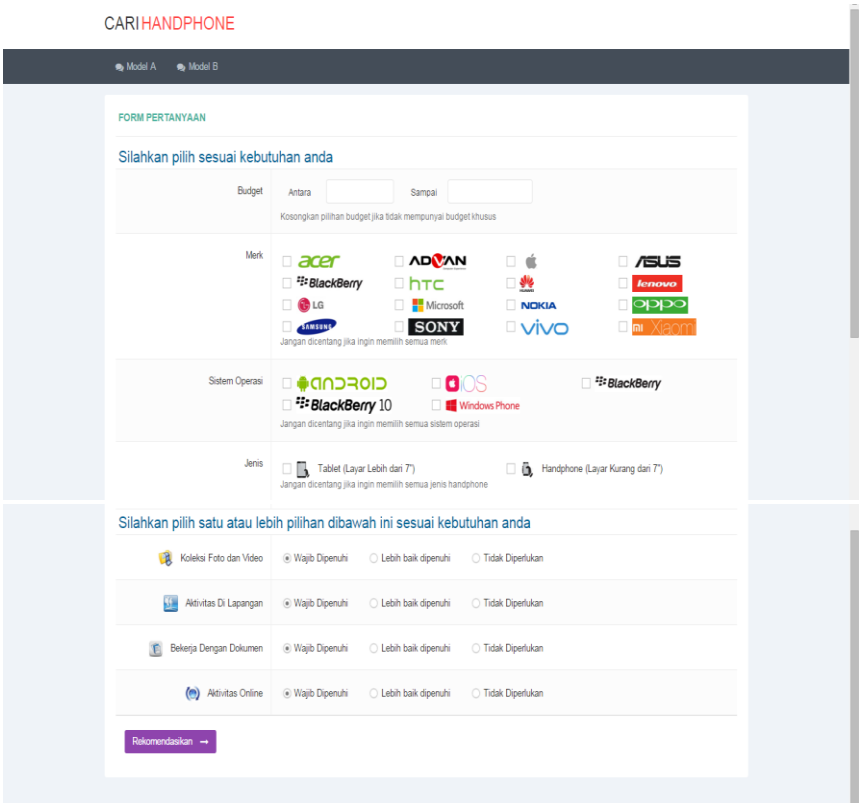
Expert user tertarik untuk mengadopsi sistem dari sisi *perceived enjoyment*, hal ini menunjukkan, mereka merasa terpandu dengan interaksi yang diberikan *func-based model*. Sementara itu, untuk *novice user*, faktor *trust* dan kemudahan (*perceived ease of use*) yang memegang peranan untuk adopsi *func-based model*. Dengan proses interaksi yang ada, dengan berbagai macam kasus yang telah ditangani oleh sistem, juga dengan fasilitas penjelasan yang mengacu pada kebutuhan fungsional, pengguna merasa bahwa ini merupakan model interaksi yang berguna dan sangat membantu (*perceived usefullness*), sehingga pengguna akan mengadopsi sistem ini nantinya untuk pencarian produk. Hal yang menarik adalah, *perceived enjoyment* bukan merupakan faktor yang mempengaruhi adopsi pengguna terhadap model interaksi. Dari pengujian sebelumnya, pengguna memang merasa bahwa *func-based model* ini lebih *enjoy* dan memandu daripada *tech-based model* (menerima H4), namun bukan faktor ini yang mempengaruhi pengguna untuk mengadopsi *func-based model*.

LAMPIRAN

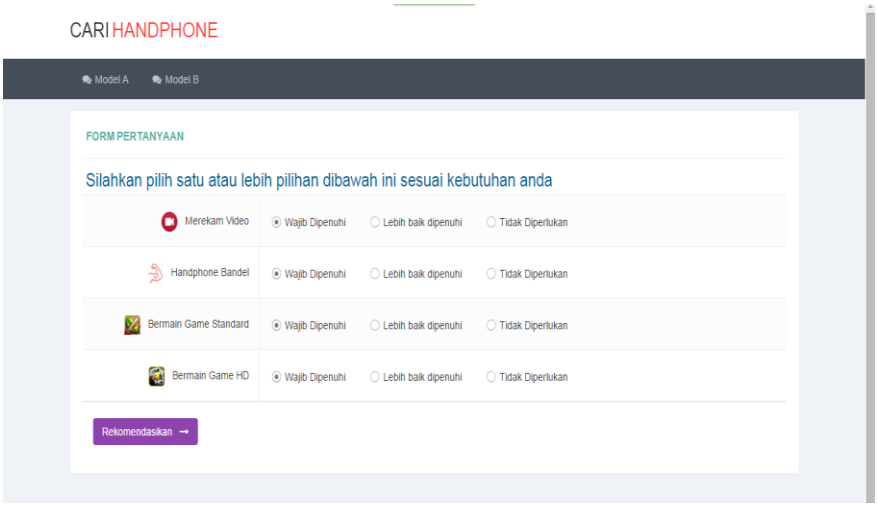
Screenshot Antarmuka Sistem

Func-based Model

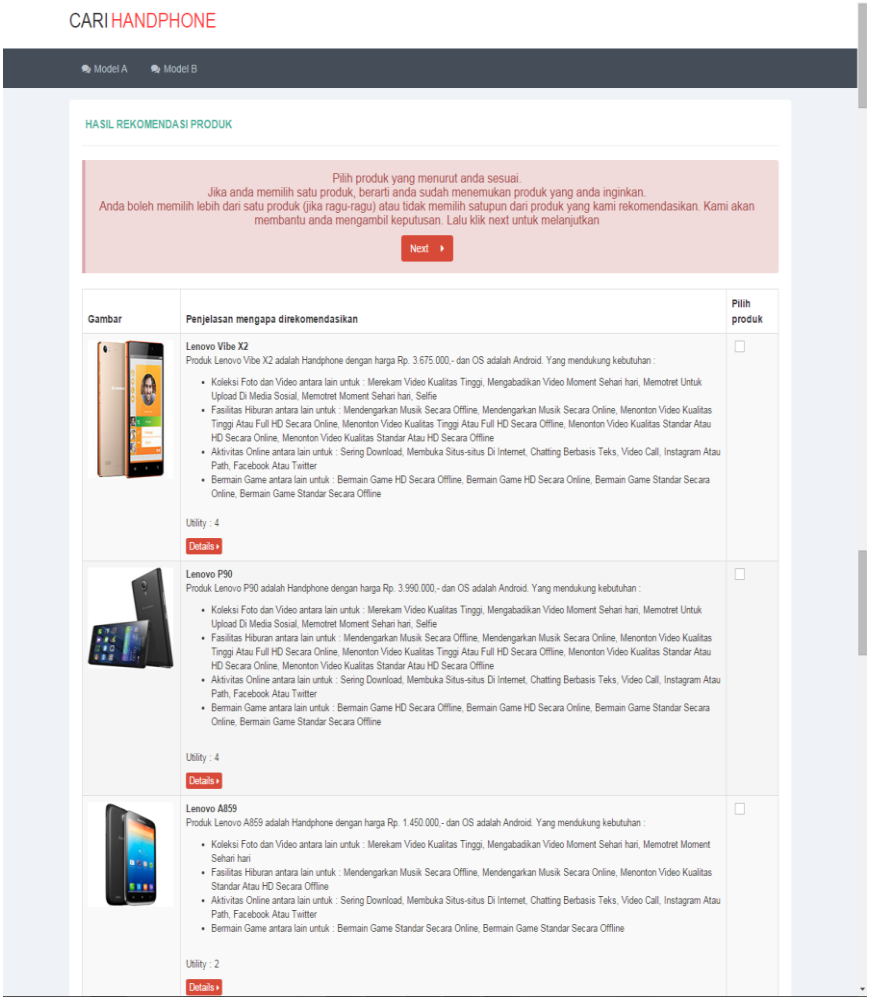
Tampilan awal interaksi atau ketika *user profile model* dalam keadaan kosong



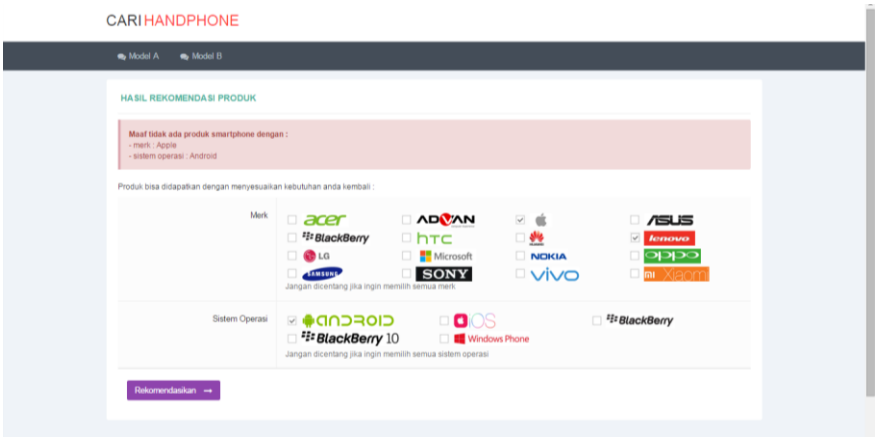
Tampilan Interaksi pemberian pertanyaan setelah awal interaksi

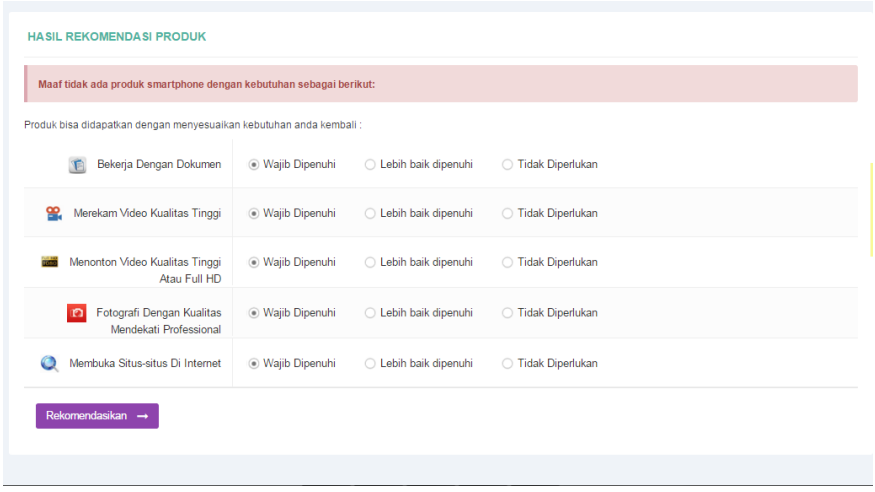


Tampilan rekomendasi produk serta penjelasan mengapa produk direkomendasikan

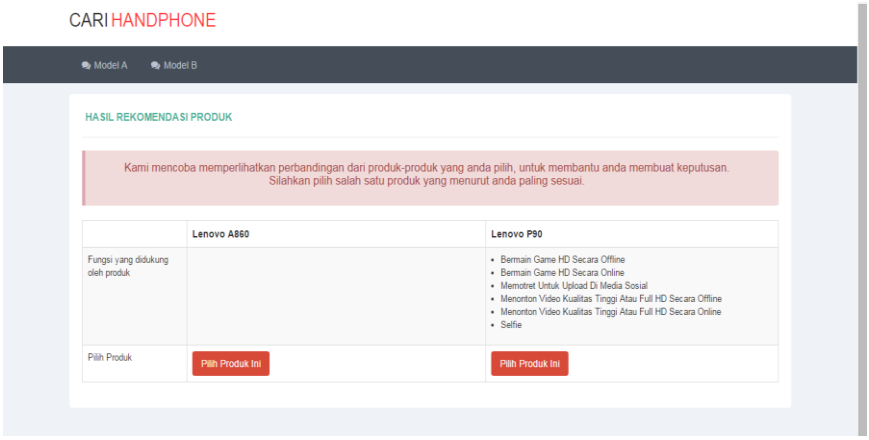


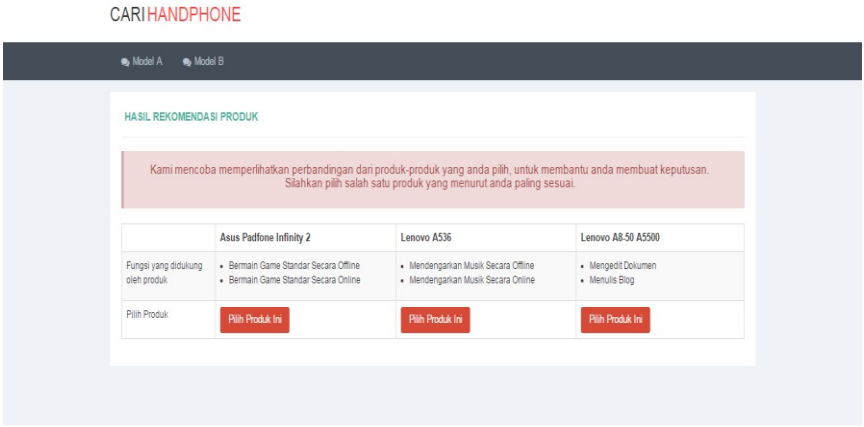
Tampilan pada saat sistem menampilkan kebutuhan pengguna yang kontradiktif (pada saat tidak ditemukan produk yang sesuai dengan kebutuhan)



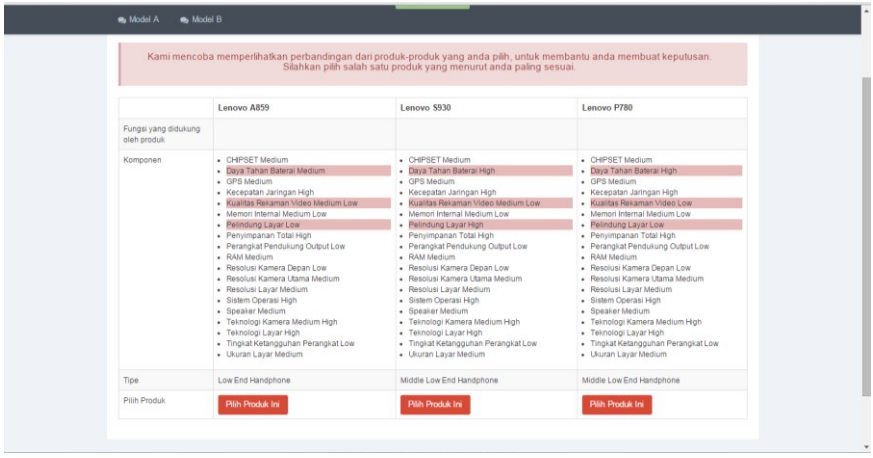


Tampilan pada saat sistem memberikan perbandingan *distinctive functional requirements* dari beberapa yang produk yang dipilih pengguna

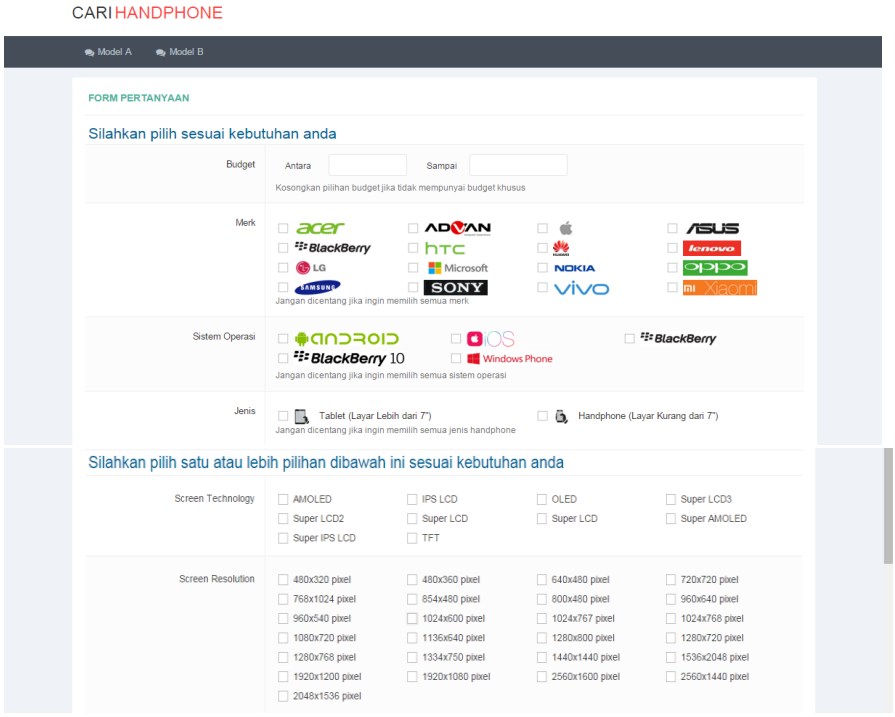


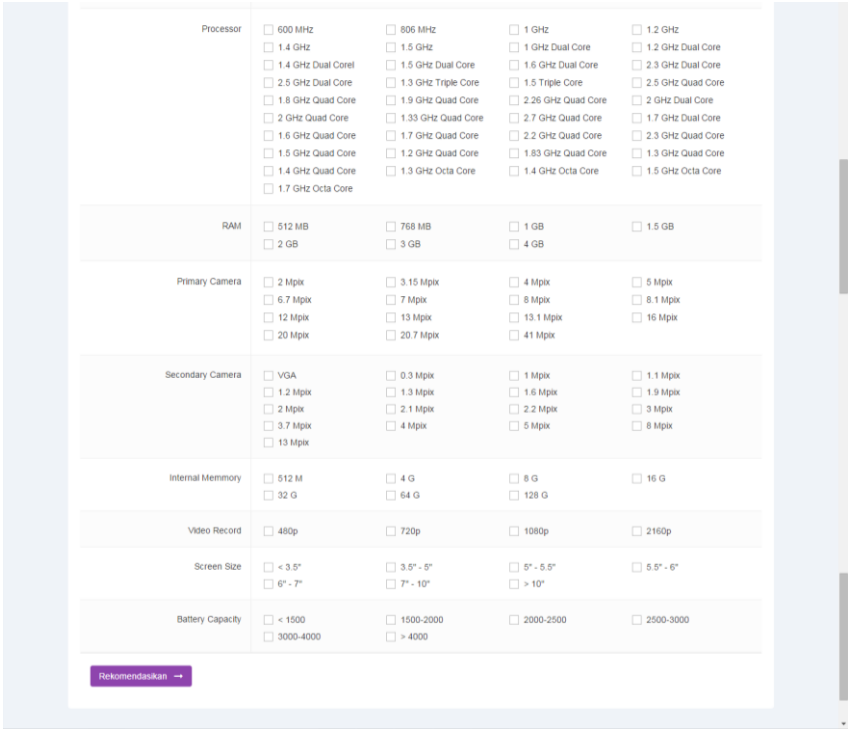


Tampilan pada saat sistem memberikan perbandingan gradasi spesifikasi dari beberapa produk yang dipilih pengguna, pada model A

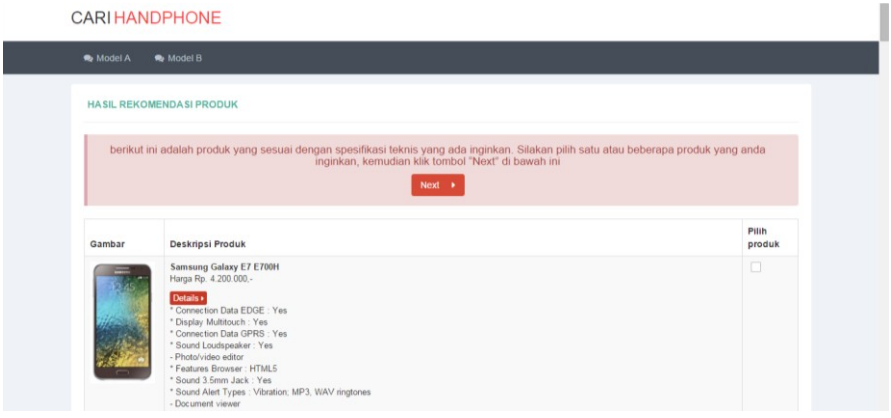


Tech-based Model (sistem pencarian produk berdasarkan fitur teknis)
Tampilan awal interaksi





Tampilan rekomendasi produk serta penjelasan mengapa produk direkomendasikan



* 2G Network : GSM 850 / 900 / 1800 / 1900
* Features Sensors : Accelerometer, gyro, proximity, compass
* Camera Primary : 8 MP, 3264 x 2448 pixels, autofocus, LED flash
* Features Java : Yes, via Java MIDP emulator
* Features Operating System : Android OS, v4.4.2 (KitKat)
* Features Colors : White, Black
* Camera Video : 720p@30fps
* Battery Standby :
* Features GPU : -
* Camera Features : Geo-tagging
* Camera Secondary : 2 MP
* Display Type : TFT capacitive touchscreen, 16M colors
* Sound Loudspeaker : Yes, with stereo speaker
* Connection WLAN : Wi-Fi 802.11 a/b/g/n, dual-band, Wi-Fi Direct, hotspot
- Photo viewer
* Features Messaging : SMS Email, Push Email,IM
* Connection Bluetooth : v4.0, A2DP, LE
- MP4/H.264
* Features GPS : Yes, with GLONASS
* Battery Talk Time : Up to 10 h (multimedia)
* Memory Internal : 32 GB, 2 GB RAM
* Connection Data Speed : HSPA, LTE
* Body Weight : 162.5 g (5.71 oz)
* Body Dimensions : 208.3 x 137.9 x 7.4 mm (8.20 x 5.43 x 0.29 in)
* Battery Music Play : Up to 64 h
* Battery Type : Non-removable Li-Ion 4200 mAh battery
* 3G Network : HSDPA 850 / 900 / 1900 / 2100 SM-N7505
* Features CHIPSET : Exynos 5260
* Display Size : 768 x 1024 pixels (~160 ppi pixel density), 8.0 inches (~69.0% screen-to-body ratio)
* Features CPU : Quad-core 1.2 GHz Cortex A7

Samsung Galaxy Tab A 8.0 3G
Harga Rp. 4.268.000,-

Details ›

* Connection Data EDGE : Yes
* Features Radio : No
* Display Multitouch : Yes
* Connection Data GPRS : Yes
* Memory Card Slot : microSD, up to 128 GB
* Connection USB : microUSB v2.0, USB Host
- MP3/WAV/eAAC+/Flac player
* SIM : Micro-SIM
* Features Browser : HTML5
* Sound 3.5mm Jack : Yes
* Sound Alert Types : Vibration; MP3, WAV ringtones
- Document viewer

* 2G Network : GSM 850 / 900 / 1800 / 1900
* Features Sensors : Accelerometer, gyro, proximity, compass
* Camera Primary : 8 MP, 3264 x 2448 pixels, autofocus, LED flash
* Features Java : Yes, via Java MIDP emulator
* Features Operating System : Android OS, v4.4.2 (KitKat)
* Features Colors : White, Black
* Camera Video : 720p@30fps
* Battery Standby :
* Features GPU : -
* Camera Features : Geo-tagging
* Camera Secondary : 2 MP
* Display Type : TFT capacitive touchscreen, 16M colors
* Sound Loudspeaker : Yes, with stereo speaker
* Connection WLAN : Wi-Fi 802.11 a/b/g/n, dual-band, Wi-Fi Direct, hotspot
- Photo viewer
* Features Messaging : SMS Email, Push Email,IM
* Connection Bluetooth : v4.0, A2DP, LE
- MP4/H.264
* Features GPS : Yes, with GLONASS
* Battery Talk Time : Up to 10 h (multimedia)
* Memory Internal : 32 GB, 2 GB RAM
* Connection Data Speed : HSPA, LTE
* Body Weight : 162.5 g (5.71 oz)
* Body Dimensions : 208.3 x 137.9 x 7.4 mm (8.20 x 5.43 x 0.29 in)
* Battery Music Play : Up to 64 h
* Battery Type : Non-removable Li-Ion 4200 mAh battery

DAFTAR PUSTAKA

Adomavicius, G. dan Tuzhilin, A. (2005): Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, *IEEE Trans. Knowl. Data Eng*., **17** (6), 734–749

Al-Maghrabi, T., Dennis, C., Halliday, S. V., dan BinAli, A. (2011): Determinants of customer continuance intention of online shopping, International Journal of Business Science and Applied Management, 6(1), 41–65.

Aubrecht, P., Záková, M, dan Kouba, Z. (2005): Ontologi transformation using generalised formalism, dalam Znalosti, roèniku conference, 154-161

Bridge, D., Goker, M. H., McGinty, L., dan Smyth, B., (2006): Case-based recommender systems, The Knowledge Engineering Review, Cambridge University Press, 20(3), 315–320

Blanco-Fernandez, Y., Pazos-Arias, J., Gil-Solla, A., Ramos-Cabrer, M., Lopez-Nores, M., Garcıa-Duque, J., Ferna´ndez-Vilas, A., Dıaz-Redondo R., dan Bermejo-Munoz, J. (2008a): A flexible semantic inference methodology to reason about user preferences in knowledge-based recommender systems, Knowledge-Based Systems, 21, 305–320

Blanco-Fernández, Y., Pazos-Arias, J., Gil-Solla, A., Ramos-Cabrer, M., dan López-Nores, M., (2008b): Providing entertainment by content-based filtering and semantic reasoning in intelligent recommender systems, IEEE Transactions on Consumer Electronics, MAY 2008, 54 (2), , 727-735

Brooke J. SUS: A Quick and Dirty Usability Scale, (1996): dalam Usability Evaluation in Industry, Taylor & Francis, 189–194.

Burke, RD., Hammond, KJ., dan Young, BC. (1997): The FindMe approach to assisted browsing, IEEE Expert, 12(5), 32–40

Burke, R. (2002): Interactive critiquing for catalog navigation in e-commerce, Artificial Intelligence Review, 18(3–4), 245–267

Burke, R. (2012): Recommender Systems: An Introduction, oleh Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich, Cambridge University Press, 2011, 336 hal, ISBN: 978-0-521-49336-9.

Cao, Y., dan Li, Y., (2007): An intelligent fuzzy-based Recommendation System for Consumer Electronic Products, Journal of Expert Systems with Applications, 33, 230-240

Crnkovic, I., Hnich, B., Jonsson, T., dan Kiziltan, Z. (2002): Specification, implementation, and deployment of components, Communications of the ACM, 45(10), 35-40.

Davis F. D. (1993): User acceptance of information technology: system characteristics, User perceptions and behavioral impacts, International Journal of Man-Machine Studies, 38(3), 475–487

Davis F. D., Bagozzi R. P., dan Warshaw P. R. (1989): User acceptance of computer technology: a comparison of two theoretical models, Management Science, 35(8), 982–1003.

Doyle, M., dan Cunningham, P. (2000): A Dynamic approach to reducing dialog in on-line decision guides, dalam Blanzieri, E and Portinale, L (eds). Proceedings of the 5th European Workshop on Case-Based Reasoning., 49–60, Berlin: Springer

Felfernig, A., Friedrich, G., Jannach, D., dan Zanker, M. (2015): Constraint-based recommender systems, dalam Recommender Systems Handbook, Springer US, 161–190.

Felfernig, A., Friedrich, G., Jannach, D., dan Zanker, M. (2007): An integrated environment for the development of knowledge-based recommender applications, International Journal of Electronic Commerce, 11(2), 11–34

Felfernig, A., Burke, dan R. (2008): Constraint-based recommender systems: technologies and research issues, Proceedings of the 10th international conference on Electronic commerce, ACM

Gu, M dan Aamodt, A. (2004): Explanation-boosted question selection in conversational CBR, dalam conversational CBR, ECCBR04 workshop on Explanation in CBR

Guarino, N., Giaretta, P. (1995): Ontologies and knowledge bases: towards a terminological clarification, chapter Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing, IOS Press, Amsterdam, 25-32

Ha, S. H. (2002): Helping online customers decide through web personalization, IEEE Intelligent System, 17(6), 34–43

Hammond, K. J., Burke, R., dan Schmitt, K. (1996): A case-based approach to knowledge navigation, dalam Leake, DB (ed.) Case-Based Reasoning: Experiences, Lessons, & Future Directions, Menlo Park, CA, AAAI Press, 125–136

Hu, B. dan Aufaure, M. (2013): A query refinement mechanism for mobile conversational search in smart environments. dalam Proceeding of IUI 2013 on Second Workshop on Interacting With smart Objects, 1-6

Hsief, S. M., Huang, S. J., Hsu, C. C., dan Chang, H. C. (2004): Personal documents recommendation system based on data mining techniques, dalam Proc. IEEE/WIC/ACM International Conference of Web Intelligent, 51–57

Ibáñez, M. J., Álvarez, P., Bhiri, S., dan Ezpeleta, J. (2009): Unary RDF-annotated petri nets: a formalism for the modeling and validation of business processes with semantic information, dalam Proceedings of the 4th International Workshop on Semantic Business Process Management, ACM, 1-4

Jacobson, I., Griss, M.L., dan Jonsson, P. (1997): Software reuse, architecture, process and organization for business success, Addison Wesley and ACM Press

Jannach, D. (2004): Advisor suite - A knowledge-based sales advisory system. dalam Lopez de Mantaras L.S. (ed.), 16th European Conference on Artificial Intelligence - Prestigious Applications of AI (PAIS), IOS Press, 720–724

Jannach, D. dan Kreutler, G. (2004): A knowledge-based framework for the rapid development of conversational recommenders, Web Information System-WISE 2004, 3306, 390-402

Jannach, D. dan Kreutler, G., (2007): Rapid development of knowledge-based conversational recommender applications with advisor suite, Journal of Web Engineering, 1(1), 000-000

Jiang, B., Wang, W., dan Benbasat, I. (2005): Multimedia-based interactive advising technology for online consumer decision support, Communications of the ACM, 48 (9), 93–98

Johnson, R. E. (1997), Frameworks = (components+ patterns), Communications of the ACM, 40(10), 39-42

Kazienko, P., dan Kołodziejski, P. (2006): Personalized integration of recommendation methods for e-commerce, International Journal of Computer Science and Application, 3(3), 12–26

Kohlmaier, A., Schmitt, S., dan Bergmann, R. (2001): A Similarity-based approach to selection in user-adaptive sales dialogs, dalam Aha, D.W. dan Watson, I. (eds.) Case-Based Reasoning Research and Development, Proc. of the 4th International Conference on Case-Based Reasoning, ICCBR-2001, Vancouver, Canada. LNAI 2080, Springer

Koufaris, M. (2002): Applying the technology acceptance model and flow theory to online consumer behavior, Information Systems Research, 13(2), 205–223

Lee, C., Chang, Y., dan Wang, M. (2009): Ontological recommendation multi-agent for Tainan City travel, Expert Systems with Applications, 36, 6740–6753

Liao, C. H., Tsou, C. W., dan Shu, Y. (2008): The Roles of perceived enjoyment and price perception in determining acceptance of multimedia-on-demand, International Journal of Business and Information, 3(1), 27–52

McCarthy, K., Reilly, J., McGinty, L., dan Smyth, B, (2004): On the dynamic generation of compound critiques in conversational recommender systems. Proceeding of Adaptive Hypermedia of Compound Critiques in Conversational Recommender System, Third International Conference

McCarthy, K., Reilly, J., McGinty, L. dan Smyth, B. (2004): Thinking positively – explanatory feedback for conversational recommender systems, dalam Proceedings of the Workshop on Explanation in CBR at the Seventh European Conference on Case-Based Reasoning (ECCB'04), 115-124

Mirzadeh, N., Ricci, F., dan Bansal, M. (2005): Feature selection methods for conversational recommender system, IEEE International Conference on e-Technology, e-Commerce, and e-Service (EEE'05), Hongkong, 772-777

Moreno, A., Valls, A., Isern, David., Marin, Lucas., dan Borr, Joan. (2013): SigTur/E-Destination: Ontologi-based personalized recommendation of Tourism and Leisure Activities, Engineering Applications of Artificial. Intelligence, 26, 633–651

Nguyen, P. H., Kaneiwa, K., Corbett, D. R., dan Nguyen, M. Q. (2008): An ontologi formalization of relation type hierarchy in conceptual structure theory. In Australasian Joint Conference on Artificial Intelligence, 79-85. Springer Berlin Heidelberg.

Nielsen, J., (1994), Usability Engineering, Academic Press Inc, 165

O'Rourke, N., Psych, R., Hatcher L. (2013): A Step-By-Step Approach to Using SAS For Factor Analysis and Structural Equation Modeling, Sas Institute.

Pavlou P. A, (2003): Consumer Acceptance of Electronic Commerce: Integrating Trust and Risk with the Technology Acceptance Model, International Journal of Electronic Commerce, 7(3), 69–103.

Park, Deuk Hee., Kim, Hyea Kfindyeong., Choi, Il Young., dan Kim, Jae Kyeong. (2012): A literature review and classification of recommender systems research, Jounal of Expert Systems with Applications, 39, 10059-10072

Reilly, J., K. McCarthy, L. McGinty, dan Smyth, B., (2005): Incremental critiquing, Knowledge-Based Systems,18(4-5), 143–151

Resnick P., dan Varian. H.R. (1997): Recommender systems, Communications of the ACM, 40(3), 56–58

Ricci, F., Venturini, A., Cavada, D., Mirzadeh, N., Blaas, D., dan Nones, M., (2003): Product recommendation with interactive query management and two folds similarity. dalam 5th International Conference on Case-Based Reasoning, Springer, Trondheim, Norway, 479–493.

Ricci, F. dan Nguyen. Q. (2007), Acquiring and revising preferences in a critique-based mobile recommender system, IEEE Intelligent Systems Special Issue: Recommender Systems, 22(3), 22-29

Ricci, F., Nguyen, Q., dan Averjanova, O., (2009): Exploiting a map-based interface in conversational recommender systems for mobile travelers, dalam Sharda, N., editor, Tourism Informatics: Visual Travel Recommender Systems, Social Communities, and User Interface Design. Information Science Reference

Schmitt, S., (2002): simVar: a similarity-influenced question selection criterion for e-sales dialogs, Artificial Intelligence Review 18(3–4), 195–221

Schmitt, S., Dopichaj, P, dan Domínguez-Marín, P. (2002): Entropy-based vs similarity-influenced attribute selection methods for dialogs tested on different electronic commerce domains. In Craw, S and Preece, A (eds), Proceedings of the 6th European Conference on Case-Based Reasoning, 380–394. Springer

Smyth, B., McGinty, L., Reilly, J., dan McCarthy, K. (2004): Compound critiques for conversational recommender systems, IEEE/WIC/ACM International Conference on Web Intelligence (WI'04), Beijing, China, 145 – 151

Tintarev, N. dan Masthoff, J. (2007): Effective explanations of recommendations: user-centered design, Proceedings of the ACM conference on Recommender systems, 153-156

Yang, S. (2010): Developing an ontologi-supported information integration and recommendation system for scholars, Expert Systems with Applications, 37, 7065–7079

Shelly G. dan Woods, D. (2010): HTML, XHTML, and CSS: Introductory, 6th Edition, Course Technology, Boston, USA, 22

Shimazu, H. (2001): ExpertClerk: navigating shoppers' buying process with the combination of asking and proposing, dalam Bernhard Nebel, editor, Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01), Springer, 1443-1448

Shimazu, H. (2002): ExpertClerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops, Artificial Intelligence Review, Kluwer Academic Publishers, Printed in the Netherlands, 18, 223-244

Smyth, B. dan Cunningham, P. (1994): A Comparison of model-based and incremental casebased approaches to electronic fault diagnosis, dalam proceedings of the Case-Based Reasoning Workshop, AAAI-1994

Smyth, B., McGinty, L., Reilly, J., dan McCarthy, K. (2004): Compound Critiques for Conversational Recommender Systems, IEEE/WIC/ACM International Conference on Web Intelligence (WI'04), 145 – 151

Symeonidis, P., Nanopoulos, A., dan Manolopoulos, Y. (2008): Providing justifications in recommender systems, IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 38 (6), 1262–1272

Thompson, C., Göker, M., dan Langley, P. (2004): A Personalized system for conversational recommendations, Journal of Artificial Intelligence Research, 21, 393–428.

Ullah, F., Sarwar, G., Lee, S, Park, Y., Moon, K., Deok, K., dan Jin, T. (2012): Hybrid recommender system with temporal information, The

International Conference on Information Network (ICOIN), IEEE, 421-425

Vesin, B., Ivanovic, M., Klasnja-Millicevic, A., dan Budimac Z. (2012): Protus 2.0: ontologi-based semantic recommendation in programming tutoring system, Expert Systems with Applications, 38(15), 12229-12246

Wen, C., Prybutok, V. R., Xu, C. (2011): An integrated model for customer online repurchase intention, Journal of Computer Information Systems; 52(1), 14-23.

Weng, S., dan Chang, H. (2008): Using ontologi network analysis for research document recommendation, Expert Systems with Applications, 34, 1857–1869

Wu, K., Zhao, Y., Zhu, Q., Tan, X., dan Zheng, H. (2011): A meta-analysis of the impact of trust on technology acceptance model: Investigation of moderating influence of subject and context type, International Journal of Information Management, 31(6), 572–581.

Zanker, M. (2012): The Influence of knowledgeable explanations on users' perception of a recommender system. Proceedings of the sixth ACM conference on Recommender systems, 269-272