Pemanfaatan Prinsip Dasar Logika dan Pohon dalam Pembuatan Sistem Penghubung Fungsi Antar Perangkat Lunak pada *Smartphone*

Ahmad Darmawan (NIM : 13513096)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
adarwawan@s.itb.ac.id

***Abstrak* — Pada zaman sekarang, banyak orang yang mengandalkan smartphone sebagai penunjang kesibukannya sehari-hari dengan memanfaatkan perangkat lunak yang ada di *playstore* atau *appstore*. Namun, terkadang perangkat lunak yang kita gunakan tidak mencukupi semua yang dibutuhkan sehingga memerlukan perangkat lunak lain yang harus diakses secara manual setelah membuka perangkat lunak utama yang seharusnya dapat dilakukan secara bersamaan. Hal itu yang justru dapat membuang waktu kita. Pada makalah ini, penulis akan memanfaatkan prinsip dasar logika dan pohon. Prinsip tersebut digunakan untuk meningkatkan produktivitas pengguna dengan mengoptimalisasi hubungan fungsi antar perangkat agar *smartphone* dapat menjalankan perangkat lunak pendukung secara otomatis dengan mempelajari perilaku pengguna ketika menjalankan sebuah perangkat lunak utama.**

***Kata Kunci* — logika, implikasi, pohon, resep, *trigger*, *action*.**

I. PENDAHULUAN

Dalam abad ke-21 ini, dunia berada dalam era informasi. Hampir semua masyarakat di seluruh dunia memerlukan teknologi digital dalam memenuhi kebutuhannya. Mulai dari kebutuhan industri, bisnis, pendidikan, bahkan sampai hiburan.

Pada tahun 2008, mulai muncul secara komersil teknologi *smartphone* yang dicanangkan oleh *Google* dan *Android Inc.* menggunakan sistem operasi *Android*. Smartphone dapat dikatakan merupakan *desktop PC* yang dibuat dalam versi telepon seluler [1].

Kini hampir semua kalangan di dunia memiliki *smartphone*. Dengan *smartphone*, pengguna mendapatkan lebih banyak kemudahan dalam mendapatkan informasi atau kebutuhan lain daripada di komputer atau laptop karena segi fleksibilitasnya.

Smartphone juga didukung oleh banyaknya perangkat lunak dalam *playstore*. Mulai dari media sosial seperti *Facebook*, aplikasi penyimpanan file secara online seperti *Dropbox*, penyimpan catatan dan pengingat kegiatan seperti *Evernote*, aplikasi ramalan cuaca, aplikasi pengontrol kesehatan, dll. Perangkat lunak tersebut membuat kini smartphone diperlakukan layaknya asisten pribadi pengguna.

Namun, perangkat lunak yang digunakan memiliki keterbatasan dan tidak dapat memenuhi semua yang diperlukan oleh pengguna. Hal ini membuat pengguna perlu menjalankan aplikasi lain secara bergantian. Misal, ketika kamu menyukai foto di *Instagram*, kamu ingin otomatis menyimpannya di *Dropbox*. Karena tidak dapat dilakukan secara otomatis maka pengguna harus membuka *Dropbox* secara manual. Tentu itu akan membuang banyak waktu apalagi jika ternyata pengguna melakukan itu secara rutin.

Oleh karena itu, muncul ide agar pengguna dapat mengatur sendiri pola dan perilaku *smartphone* sesuai keinginan mereka menggunakan prinsip logika sederhana yaitu implikasi serta dengan menggunakan prinsip pohon.

II. LANDASAN TEORI

2.1. Teori Logika

2.1.1 Logika dalam Informatika

Logika adalah suatu cabang ilmu matematika yang membahas penalaran (*reasoning*). Menurut Kamus Besar Bahasa Indonesia, Logika merupakan cara berpikir dengan mengembahkan sesuatu berdasarkan akal budi dan bukan dengan perasaan atau pengalaman [2].

Dalam bidang Ilmu Komputer atau Informatika, banyak masalah yang dapat diselesaikan menggunakan teori logika. Misal dalam pembuatan sistem pakar untuk mobil, sistem klasifikasi hewan, dsb. Bahkan, logika sangat berperan penting sebagai pondasi dasar algoritma dan pemrograman. Sebagai contoh algoritma analisis bilangan bulat seperti berikut:

```
if a > 0 then
        output(a bilangan positif)
else if a < 0 then
        output(a bilangan negatif)
else { a = 0 }
        output(a bilangan nol)
```

Gambar 1 : Logika dalam pemrograman

2.1.2 Logika Preposisi

Logika preposisi didasarkan pada hubungan antara pernyataan (*statements*) atau kalimat yang memiliki arti penuh dan utuh. Hal ini membuat pernyataan tersebut dapat ditinjau bernilai salah atau benar tetapi tidak keduanya.

Sebagai contoh kalimat "19 adalah bilangan prima" merupakan kalimat preposisi. Kalimat tersebut bernilai benar.

Pada umumnya, kalimat preposisi diwakili dengan simbol huruf kecil p,q,r,…. Contohnya,

p : Hewan yang memiliki bulu adalah burung.

q : Hari ini adalah hari selasa.

r : Jika hari esok hujan, maka ia akan membatalkan jadwal kegiatan di hari esok.

2.1.3 Operator dalam Logika

Sebuah kalimat preposisi dapat dikombinasikan dengan kalimat lain dengan menggunakan operator. Operator dalam logika adalah sebagai berikut:

a. Konjungsi

Gabungan dua pernyataan tunggal yang menggunakan kata penghubung "dan" sehingga terbentuk pernyataan majemuk disebut konjungsi. Konjungsi disimbolkan dengan "$\wedge$". Syarat agar penyataan konjungsi benar adalah semua penyusunnya penyusunnya harus benar.

Salah satu contoh pernyataan konjungsi adalah "p $\wedge$ q : 7 merupakan bilangan ganjil dan prima." kalimat tersebut terdiri dari dua pernyataan yaitu "p : 7 merupakan bilangan ganjil." dan "q : 7 merupakan bilangan prima.".

b. Disjungsi

Disjungsi adalah proposisi majemuk yang menggunakan kata hubung "atau".

Poposisi "p atau q" dinotasikan p $\vee$ q. Tidak seperti pernyataan berperangkai "dan" yang mempersyaratkan terpenuhinya kebenaran semua unsurnya, pernyataan berperangkai "atau" menawarkan suatu pilihan, artinya jika paling tidak salah satu dari kedua unsur proposisinya terpenuhi maka hal ini sudah cukup untuk pernyataan tersebut dikatakan benar.

Contoh, "p $\vee$ q : Graf semi-Euler memiliki dua simpul berderajat ganjil atau tidak ada simpul berderajat ganjil sama sekali.".

c. Negasi

Dari sebuah pernyataan tunggal (atau majemuk), kita dapat membuat sebuah pernyataan baru berupa "ingkaran" dari pernyataan itu. "ingkaran" disebut juga "negasi" atau "penyangkalan". Ingkaran menggunakan operasi "~".

Jika suatu pernyataan p benar, maka negasinya p salah, dan jika sebaliknya pernyataan p salah, maka negasinya p benar.

Contoh, "~p : Hari ini adalah hari libur." adalah negasi dari "p : Hari ini bukan hari libur.".

d. Implikasi

Implikasi adalah proposisi majemuk memiliki pola "jika p maka q ". Simbol p dapat dikatakan sebagai hipotesis, antesenden, premis, atau kondisi. Sedangkan q adalah konsekuen atau konklusi.

Perhatikan bahwa dalam implikasi yang dipentingkan nilai kebenaran premis dan konsekuen, bukan hubungan sebab dan akibat diantara keduanya [3]. Contoh implikasi yang tidak ada hubungan sebab-akibat yaitu "Jika PERSIB menjadi juara *ISL*, maka 3 dan 5 relatif prima.

Tabel kebenaran dari implikasi adalah sebagai berikut:

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Gambar 2. Tabel Kebenaran Implikasi [3]

2.1.4 Interferensi

Interferensi adalah proses penarikan kesimpulan dari sekumpulan pernyataan. Ada beberapa kaidah interferensi antara lain:

(a) Modus Ponen

(b) Modus Tolen

(c) Silogisme

Silogisme adalah kaidah penarikan kesimpulan dari dua pernyataan implikasi. Pola dari silogisme adalah:

```
p -> q
q -> r
------+
p -> r
```

2.2. Pohon

Pohon adalah salah satu jenis graf yang bersifat tak-berarah, terhubung, dan tidak mengandung sirkuit [4].

Gambar 3 : Contoh Pohon [4]

2.2.1. Sifat-sifat Pohon

P = (V,E) adalah graf tak-berarah sederhana dan jumlah simpulnya n. Memiliki sifat-sifat [4]:

(a) P adalah pohon.
(b) Setiap pasang simpul di dalam P terhubung dengan lintasan tunggal.
(c) P terhubung dan memiliki m = n-1 buah sisi.
(d) P tidak mengandung sirkuit dan memiliki m = n-1 buah sisi.
(e) P tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
(f) P terhubung dan semua sisinya adalah jembatan.

2.2.2. Istilah-istilah pada Pohon

(a) Orang tua (parent) dan Anak (child)
(b) Lintasan (path)
(c) Saudara kandung (sibling)
(d) Upapohon
(e) Derajat
(f) Daun

Daun adalah istilah untuk simpul yang tidak mempunyai anak.

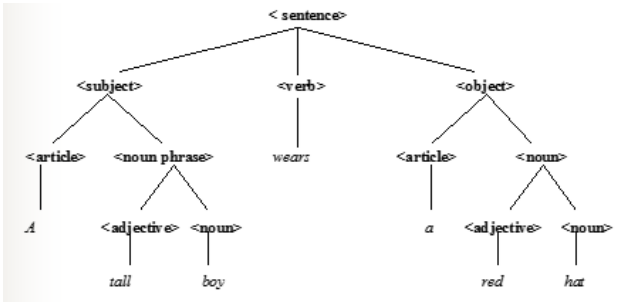
(g) Simpul dalam
(h) Aras (level)

Akar dari pohon merupakan aras ke-0. Sedangkan anaknya adalah aras ke-1. Begitupun dengan simpul dengan aras ke-n mempunyai anak dengan aras ke-n+1.

(i) Tinggi (height) atau Kedalaman (depth)

2.2.3. Pohon *n-ary*

Pohon berakar yang setiap simpul cabangnya mempunyai maksimal *n* buah anak disebut sebagai pohon *n-ary*.



Gambar 4 : Contoh Pohon Biner, Pohon dengan maksimal 2 anak [4].

Pohon n-ary dikatakan teratur dan penuh jika setiap cabang mempunyai tepat *n* buah anak.

III. KEGUNAAN TEORI LOGIKA PADA SISTEM PENGHUBUNG FUNGSI ANTAR PERANGKAT LUNAK

3.1. Pernyataan Preposisi sebagai Resep

Di dalam membuat sistem penghubung fungsi antar perangkat lunak, hal yang paling utama adalah pernyataan hubungan fungsi antar aplikasi. Pernyataan kita sebut sebagai resep (*recipe*). Resep secara sederhana terdiri dari kalimat implikasi "Jika a maka b".

Simbol "a" didefinisikan sebagai sebuah pemicu (*trigger*) dari sistem. Pemicu inilah yang harus dibuka secara manual oleh pengguna. Pemicu bisa juga merupakan sebuah keadaan (*event*) unik dari program pada *smartphone*. Sebagai contoh program prediksi cuaca mendeteksi esok hari akan hujan, program pengingat waktu sholat mendeteksi waktu telah memasuki subuh, dll. Kemudian, pemicu akan memanggil sistem untuk mencari resep yang berkaitan dengan simbol "a" sebagai pemicu.

Jika terdapat simbol "a" pada resep sebagai sebuah pemicu, maka sistem akan melihat bagian yang kedua dari satu resep tersebut yaitu bagian aksi (*action*) yang disimbolkan dengan "b". Sistem akan membuka secara otomatis aksi yang sudah terdapat dalam resep pada sistem.

Gambar 5 : Pola resep (recipe) sistem [5]

3.2. Klasifikasi Resep

Untuk menggunakan sistem tersebut, kita dapat membuat sendiri resep sesuai yang kita inginkan. Dalam pembuatan resep tersebut, ada dua klasifikasi resep yaitu: Resep Sederhana dan Resep Kompleks.

3.2.1. Resep Sederhana

Resep sederhana adalah resep yang terdiri dari satu pemicu dan satu aksi.

Salah satu ontoh dari resep sederhana adalah misal pengguna ingin agar ketika dia menyukai sebuah foto di *Instagram*, maka sistem secara otomatis akan mengunggah foto tersebut kedalam *Dropbox*. Jika kita tinjau dari contoh tersebut maka pemicu didefinisikan dengan "menyukai foto di *Instagram*" dan aksi didefinisikan dengan "unggah foto tersebut dalam *Dropbox*".

3.2.2. Resep Kompleks

Resep kompleks merupakan resep yang dapat terdiri dari minimal dua pemicu dan/atau minimal dua aksi.

Salah satu contoh dari resep kompleks dengan dua pemicu dan satu aksi adalah misal pengguna ingin ketika program prediksi cuaca memprediksi bahwa besok hujan dan pada program kalender menyatakan terdapat sebuah rapat di esok hari, maka sistem akan mengirimkan *SMS* berupa pengingat untuk membatalkan rapat. Pemicu dari resep tersebut adalah pada aplikasi prediksi cuaca dan kalender sedangkan aksinya adalah pada program *SMS*.

Sedangkan contoh dari resep kompleks dengan satu pemicu dan dua aksi adalah misal pengguna merupakan aktivis sosial media. Ia ingin memposting foto yang ia potret ke dalam *Facebook* dan *Twitter*. Maka, pemicu dari resep tersebut adalah program Kamera sedangkan aksinya adalah aplikasi *Facebook* dan *Twitter*.

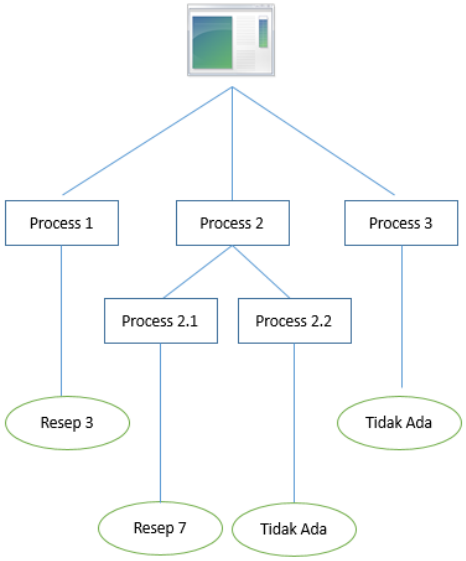
IV. KEGUNAAN POHON PADA SISTEM PENGHUBUNG FUNGSI ANTAR PERANGKAT LUNAK

Pada uraian sebelumnya, sistem menggunakan teori logika untuk membuat resep-resep untuk menghubungkan satu aplikasi dengan aplikasi lain. Sebuah aplikasi mungkin bisa terdapat dalam lebih dari satu resep sebagai pemicu. Hal yang membedakan antara satu pemicu dengan yang lainnya adalah proses yang sedang dilakukan. Sebagai contoh, aplikasi *Instagram* dapat memicu dua hal yang berbeda. Pertama, jika pengguna menyukai foto milik orang lain, maka sistem akan mengunggah foto tersebut dalam *Dropbox*. Kedua, jika pengguna sedang mengunduh foto baru di *Instagram*, maka sistem akan menggunakan foto tersebut sebagai *wallpaper smartphone* pengguna tersebut. Oleh karena itu, penulis memilih struktur pohon dalam mengimplementasikan kasus tersebut.

Setiap aplikasi yang terdapat pada resep mempunyai pohonnya masing-masing. Selain itu, sebuah aplikasi bisa mempunyai maksimal dua pohon apabila aplikasi menjadi pemicu dan aksi dalam resep-resep yang telah dibuat.

4.1. Pohon untuk Pemicu (*trigger*)

Pohon untuk pemicu (*trigger*) digunakan untuk mengidentifikasi apa yang sedang dilakukan oleh aplikasi pemicu. Dari identifikasi tersebut akan berakhir dengan menunjuk resep yang akan digunakan.

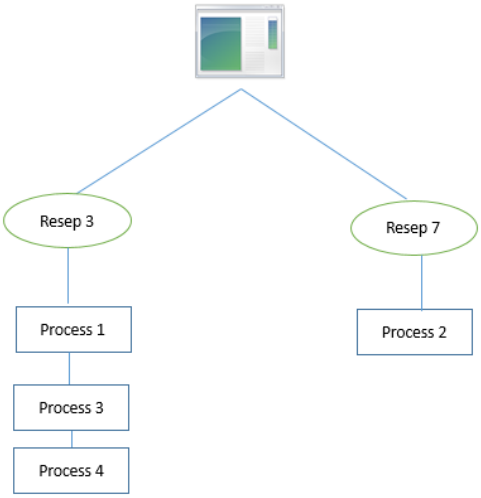


Gambar 4 : Contoh pohon untuk pemicu (trigger)

Pohon untuk pemicu ini mempunyai khas untuk setiap daunnya mewakili resep mana yang akan ditunjuk untuk memproses aksi.

4.2. Pemicu untuk Aksi (*action*)

Pohon untuk aksi (*action*) digunakan sebagai *flowchart* yang akan menjalankan aplikasi sesuai pemicu dari resep yang dimaksud.



Gambar 5 : Contoh pohon untuk aksi (action)

Pohon untuk aksi ini mempunyai khas yaitu pada aras ke-1 dari pohon tersebut merupakan indeks resep yang dikembalikan oleh pohon untuk pemicu.

V. CONTOH APLIKASI SISTEM PENGHUBUNG FUNGSI ANTAR PERANGKAT LUNAK

Salah satu contoh aplikasi yang mengembangkan sistem penghubung fungsi antar perangkat lunak pada *smartphone* adalah *IFTTT*.

Gambar 7 : Tampilan *IFTTT* untuk *Android* [6]

IFTTT adalah sebuah layanan yang memberikan akses kepada pengguna untuk membuat hubungan aplikasi di smartphone sebagai pemicu (*trigger*) dengan layanan aplikasi berbasis web seperti *Gmail*, *Google Reader*, *Instagram*, dan *Craiglist* sebagai aksi (*action*). *IFTTT* sendiri merupakan singkatan yang berarti "*IF This Then That*".

Aplikasi ini dibuat pada September 2011 oleh Linden

Tibbets, Jesse Tane, dan Alexander Tibbets. Saat ini, aplikasi tersebut sudah berkembang dan tiap harinya ada sekitar 15 juta resep digunakan oleh aplikasi tersebut. [6]

Untuk informasi lebih lanjut mengenai *IFTTT*, Anda dapat mengakses laman https://ifttt.com/.

VI. Simpulan

Pada era teknologi ini, hampir setiap orang di dunia memiliki *smartphone*. Hal itu dikarenakan banyak aplikasi yang ditawarkan untuk membantu dalam menunjang kebutuhannya sehari-hari. Mulai dari media sosial, aplikasi *cloud storage*, pencatat kegiatan, dll.

Namun, beberapa aplikasi yang ada belum mampu menunjang semua yang pengguna harapkan. Sehingga, pengguna harus memasang aplikasi lain dan mengaksesnya secara manual. Itu membuat pengguna akan membuang waktu. Apalagi jika yang ia lakukan itu merupakan hal yang rutin.

Dengan menggunakan teori matematika distrik sederhana dan sedikit kreatifitas, kita dapat membuat sistem yang berguna meningkatkan produktifitas penggunanya. Salah satu contohnya memanfaatkan prinsip dasar logika dan pohon dalam sistem penghubung fungsi antar perangkat lunak pada smartphone.

VII. Ucapan Terima Kasih

Pertama-tama penulis mengucapkan syukur kepada Allah SWT, karena dengan rahmatnya penulis berkesempatan untuk menyelesaikan makalah ini. Penulis juga berterima kasih kepada dosen mata kuliah IF2120 Matematika Distrik, Dr. Ir. Rinaldi Munir yang telah berjasa mengajarkan materi Prinsip Logika dan membimbing dalam pembuatan makalah sehingga penulis mampu menulis makalah ini. Terima kasih juga kepada rekan-rekan yang senantiasa memberikan masukan dan semangat selama proses pembuatan makalah ini.

Referensi

[1] Ziegler, Chris. *Android: A visual history*. The Verge. Alamat : http://www.theverge.com/2011/12/7/2585779/android-history. Diakses pada 7 Desember 2014 pukul 23.59.

[2] *Kamus Besar Bahasa Indonesia Online*. Alamat : http://kbbi.web.id/. Diakses pada 8 Desember 2014 pukul 01.13.

[3] Munir, Rinaldi. *Slide Kuliah IF2120 Matematika Diskrit - Pengantar Logika*. 2014.

[4] Munir, Rinaldi. *Slide Kuliah IF2120 Matematika Diskrit – Pohon*. 2014.

[5] *About IFTTT*. Alamat : https://ifttt.com/wtf. Diakses pada 10 Desember 2014 pukul 12.08.

[6] *IFTTT on Playstore*. Google Play Store. Alamat : https:// play.google.com/store/apps/details?id=com.ifttt.ifttt. Diakses pada 10 Desember 2014 pukul 19.09.

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 November 2014

Ahmad Darmawan (NIM : 13513096)