ISSN 1412 5641

MediaTeknika

Jurnal Teknologi

Vol. 12, No. 1, Juni 2017

Penyusunan Kebutuhan Perancangan Mesin Hemodialisis Menggunakan Kansei Engineering Serta Aplikasi QFD dan TRIZ
Khawarita Siregar, Rosnani Ginting, Syahrul Fauzi Siregar

Penjadwalan Tenaga Kerja untuk Menentukan *Regular Days Off* (RDOs) dengan Menggunakan Algoritma *Monroe*
Khalida Syahputri, Jelly Leviza, T. Keizerina Devi

Analisis Tegangan Roda Gigi Miring pada Transmisi Kendaraan Roda Empat berdasarkan AGMA dan ANSYS
Hadi Sutanto

Pengaruh *Hard Chrome Plating* pada Peningkatan Kekerasan Baja Komponen Kincir
Budi Setyahandana, Yohanes Eko Christianto

Karakteristik Kinerja Mesin Diesel Stasioner dengan Bahan Bakar Campuran Biodiesel dari Biji Kemiri Sunan
Farida Ariani, Elisabeth Ginting, Tulus Burhanuddin Sitorus

Visualisasi Aturan Asosiasi Berbasis *Graph* untuk Data Tindak Kejahatan
Eduardus Hardika Sandy Atmaja

Pemodelan Proses Pemilihan Rute pada Protokol Babel dengan *Activity Diagram* dan *Transition System*
Vittalis Ayu

Media Teknika Jurnal Teknologi
Vol. 12, No. 1, Juni 2017

Pemodelan Proses Pemilihan Rute pada Protokol Babel dengan *Activity Diagram* dan *Transition System*

Vittalis Ayu
Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Sanata Dharma, Yogyakarta, Indonesia
e-mail: vittalis.ayu@usd.ac.id

Abstract

Babel Routing Protocol was designed by Juliuz Chrobozcek and its written specification was published in RFC 6126 in April 2014. Babel is one of distance vector routing protocols that has its own distinctive features. Although it sends updates periodically to its neighbours, it also sends triggered updates whenever it detects its neighbour's changes. The changes can include broken links and new link detections. Babel route selection process is also unique, because there are feasibility condition requirements to ensure that the routing loop does not exist during the routing process. Unified Modelling Language (UML) is a modelling language widely used in software engineering. Its popularity is based on its capability to express and describe different aspects of a software. Activity diagram is one of the UML models to describe the flow activity of a software execution. It answers and explains situations like the different states that the flow can reach and the prerequisite to reach the next states. On the other hand, transition systems describe the transition state of the running software. When a software is executed, we have to keep track of the state of the software. We need to describe how state can jump from one to anotherstate and what requirements to do the transition from one state to another one.It will be helpful inthe debugging process laterwhen some errors occur. However, the transition system can be derived from the activity diagram that has described the software flow. This researchis conducted to give more understanding about the flow of the route selection process in Babel Routing Protocol and describe the flow in the activity diagram and further define it with a transition system.

Keywords*: babel, uml, transition system, routing protocol*

1. Pendahuluan

Optimisasi dan inovasi untuk mekanisme perutean data terus dilakukan baik secara *hardware* maupun *software* untuk mempercepat pengiriman paket dari sumber ke tujuan. Salah satu kreasi modifikasi yang dilakukan adalah dengan menyusun protokol routing baru yang diyakini mampu memberikan performa yang lebih daripada pendahulunya. Protokol routing merupakan suatu peraturan tentang tata cara pertukaran informasi yang dilakukan oleh perangkat jaringan. Contoh dari protokol routing antara lain adalah protokol routing*Optimized Link State Routing Protocol* (OLSR) [1], *Ad-Hoc On Demand Distance Vector* (AODV) [2], *Border Gateway Protocol* (BGP) [3], *Routing Information Protocol* (RIP) [4], *Open Shortest Path First* (OSPF) [5], *Enhanced Interior Gateway Routing Protocol* (EIGRP) [6], *Destination-Sequenced Distance Vector* (DSDV) [7], *Better Approach To Mobile Ad-hoc Networking* (BATMAN) [8],dan Babel [9]. Protokol routing terbagi menjadi dua kelompok berdasarkan cara penyampaian informasinya yaitu protokol routing berbasis *distance vector* dan protokol routing berbasis *link state.* Protokol routing berbasis *distance vector* menyampaikan informasinya hanya ke router tetangganya saja dan mendapatkan informasi tentang router lain juga hanya melalui router tetangganya saja sedangkan protokol routing berbasis *linkstate* menyampaikan informasinya ke seluruh router dalam jaringan tersebut dan mendapatkan informasi dari semua router di jaringan tersebut. AODV, BGP, EIGRP, dan DSDV merupakan contoh protokol routing berbasis *distance vector* sedangkan OLSR dan OSPF

Diterima 18 Mei 2017; Direvisi 29 Agustus 2017; Disetujui 30 September 2017

merupakan contoh protokol routing berbasis *link state*. Babel merupakan protokol routing yang pembentukannya terinspirasi dari mekanisme protokol routing OLSR, DSDV, AODV dan EIGRP. *Routing loop* merupakan salah satu masalah yang terjadi saat penyebaran informasi routing ke router yang lain. Ketika ada jalur antar router yang putus, maka informasi tentang terputusnya jalur ini harus disebarkan ke seluruh router agar semua router tidak memakai jalur yang putus tersebut. Jika peralihan rutenya terlambat, maka data yang menuju ke jalur yang putus akan selalu berputar-putar di sekitar jalur yang putus. Penelitian yang dilakukan oleh Abolhasan [10] memberikan hasil bahwa protokol routing Babel memiliki keunggulan dalam *recovery time* setelah ada jalur yang putus dibandingkan dengan protokol routing Batman dan OLSR. Babel memiliki beberapa mekanisme untuk menjamin bahwa *routing loop* tidak akan terjadi dan mempercepat terjadinya konvergensi dengan memanfaatkan mekanisme pemilihan rute yang dimilikinya [9]. Pemahaman tentang mekanisme proses pemilihan rute pada Babel perlu dilakukan untuk mendalami proses apa saja yang terjadi di dalamnya.

Transition systems merupakan suatu model untuk menjelaskan perilaku dari sebuah sistem. Penjelasan yang diperoleh dengan *transition diagram* diharapkan memberikan deskripsi mengenai syarat yang harus dipenuhi untuk berpindah dari satu *state* ke *state* lainnya. *Activity diagram* yang merupakan bagian dari *Unified Modelling Language* adalah salah satu *tools* yang bisa kita gunakan untuk memberikan pemahaman aktivitas apa saja yang terjadi pada proses pemilihan rute, pilihan yang ada, dan letak perulangan yang dapat terjadi[11]. *Activity diagram* juga digunakan dalam penelitian yang dilakukan oleh Thramboulidis untuk menjelaskan desain dari *Transmission Control Protocol* (TCP) [12].

2. Metode Penelitian

Penelitian ini memodelkan proses pemilihan rute pada protokol routing Babel akan dengan dua cara yaitu pemodelan dengan *activity diagram* dan pemodelan dengan *transition system.*

2.1. Transition System

Transition system merupakan suatu graf berarah yang terdiri dari *state* dan *edge*. *State* memberikan informasi tentang keadaan sistem pada waktu tertentu[13]. Misalnya pada suatu sistem lampu lalu lintas, maka *state* sekarang adalah kondisi bahwa lampu lalu lintas berwarna hijau menyala, lampu merah tidak menyala, dan lampu kuning tidak menyala. *Edge* memberikan gambaran bagaimana *state* dapat berubah menjadi *state* lainnya. *Edge* menjelaskan transisi *state* yang terjadi di dalam sebuah sistem. Misalnya ketika pada sistem lampu lalu lintas, *edge* menjelaskan bahwa untuk berubah dari *state* (lampu hijau menyala, lampu kuning tidak menyala, lampu merah tidak menyala) ke *state* (lampu hijau tidak menyala, lampu kuning menyala, lampu merah tidak menyala)adalah *constraint* waktu 20 detik. Setelah 20 detik, *state* (lampu hijau menyala, lampu kuning tidak menyala, dan lampu merah tidak menyala) akan berubah ke*state* (lampu kuning menyala, lampu merah tidak menyala, lampu hijau tidak menyala).

2.2. Activity Diagram

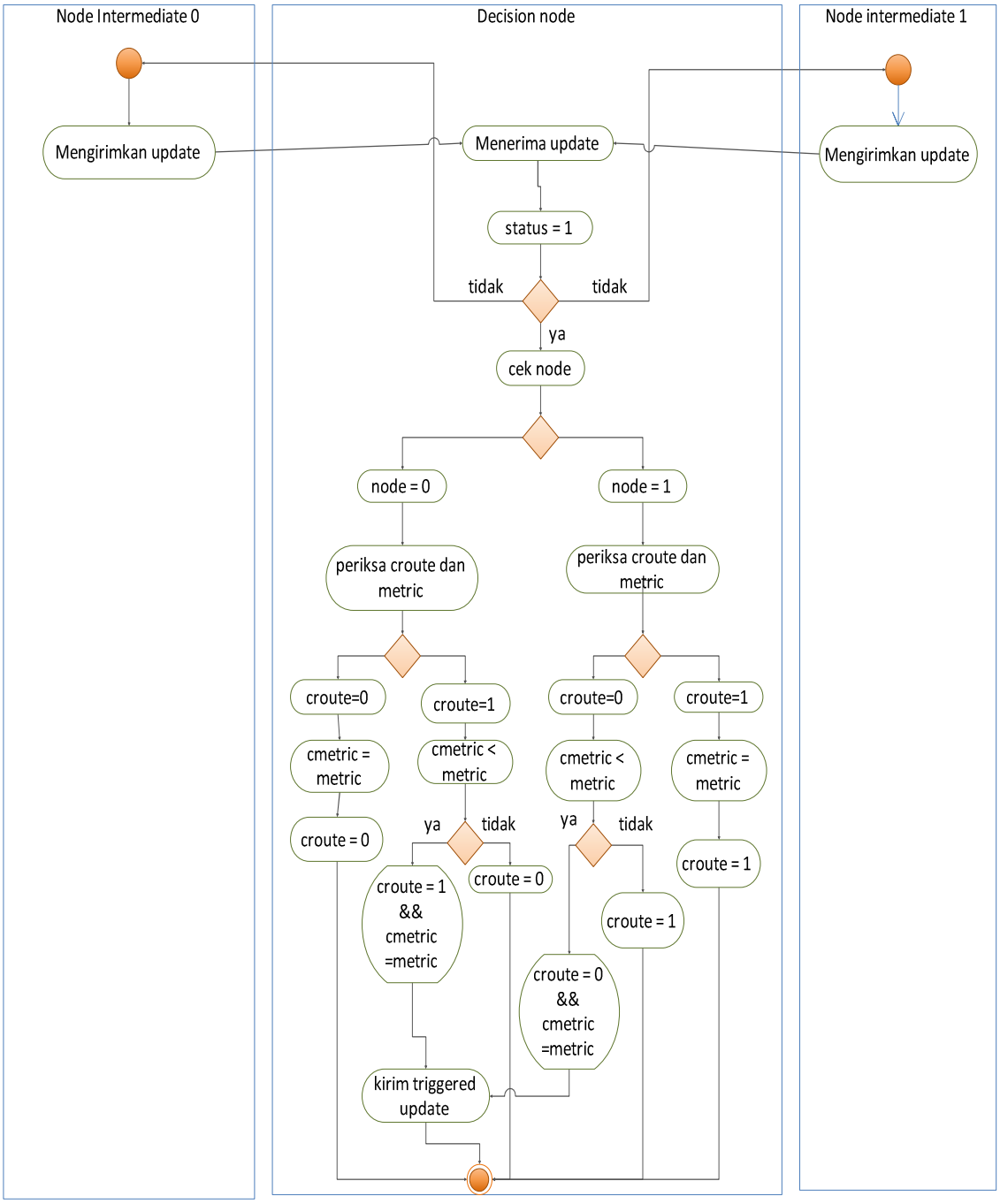
Activity diagram merupakan diagram yang mampu menjelaskan secara prosedural alur proses dari sebuah sistem [11]. Dalam diagram ini dimungkinkan untuk mengevaluasi kemungkinan adanya lebih dari satu jalur yang terbentuk dan berjalan secara bersamaan. Penggambaran dari *activity diagram* dimulai dari *initial node* hingga berakhir di *end node*. Yang dapat menjadi catatan bahwa *initial node* dalam sebuah *activity diagram* diperbolehkan lebih dari satu. Hal ini dilakukan untuk mengakomodasi jika sistem yang dimodelkan memiliki lebih dari satu input.

3. Pemodelan

Asumsi topologi: topologi jaringan yang digunakan dalam penelitian ini merupakan sub-topologi dari penelitian Jonglez [14]. Topologi jaringan terdiri dari satu node sumber, node *intermediate* 0, node *intermediate* 1, dan satu node tujuan. Node *intermediate* merupakan node yang berada diantara node sumber dan node tujuan. Node *intermediate* 0 dan node *intermediate* 1 masing masing memliki jalur menuju node tujuan. Kedua node *intermediate* ini masing-masing mengirimkan paket *update* ke tetangganya termasuk ke node sumber. Node sumber disebut juga sebagai *decision* node karena setelah mendapatkan paket *update* dari node *intermediate* 0 dan node *intermediate* 1, node sumber akan memutuskan jalur paket yang dikirimkannya apakah akan melalui node *intermediate* 0 atau node *intermediate* 1. Pada pemodelan protokol routing Babel digunakan sebelas *state* untuk mengabstraksikan sistem. Perpindahan *state* dari proses pemilihan rute pada protokol routing Babel menunjukan adanya dependensi. Contohnya adalah paket *update* yang diterima akan dihitung *metric*nya jika *update*-nya adalah *feasible*. Jika *update*-nya tidak *feasible*, maka paket *update* tersebut akan diabaikan.

3.1. Pemodelan dengan*activity diagram* untuk proses pemilihan rute protokol routing Babel

Gambar 1 merupakan urutan proses yang terjadi pada proses pemilihan rute pada protokol routing Babel. Proses ini melibatkan dua *intermediate node* dan satu *decision node*. *Intermediate node* merupakan node tetangga dari *decision node*. *Decision node* merupakan node pengambil keputusan lewat node mana paket akan diteruskan. Secara detail, proses ini dimulai dengan *intermediate node* mengirimkan paket update ke *decision node*. *Decision node* akan menerima paket update dan akan memeriksa status dari update yang diterima. Jika status=0 maka updatenya tidak *feasible*. Jika paket tidak *feasible*, maka update akan diabaikan dan akan menunggu paket update selanjutnya. Jika status=1, maka update *feasible*. Jika paket *feasible*, maka akan diteruskan untuk mengecek isi update. Periksa apakah paket update memiliki jalur ke node tujuan lewat *node intermediate* 0 atau *node intermediate* 1. Jika *node intermediate* memiliki jalur ke node tujuan, maka periksa rute yang sekarang dipakai untuk meneruskan paket ke tujuan catat sebagai *current route*. Jika rute yang sekarang dipakai untuk meneruskan paket ke node tujuan adalah rute yang melewati *node intermediate* 0 maka set *current route*=0 begitu pula jika rute yang sekarang dipakai untuk meneruskan paket ke node tujuan adalah rute yang melewati *node intermediate* 1, maka set *current route*=1. Kemudian kita periksa *metric* yang dimiliki oleh *current route.Metric* ini disebut sebagai *cmetric*. Langkah berikutnya kita kombinasikanbeberapa hal yang sudah kita periksa tadi yaitu *node intermediate*, *current route*, *cmetric*, dan *metric* untuk menentukan apakah rute berubah dan berapa *metric* rute tersebut sekarang. Jika *node intermediate*-nya adalah *node intermediate* 0, *current route*=0 serta *metric* lebih kecil dari *cmetric* maka *set current route* = 0. Boolean *route_change* akan menjadi penanda apakah terjadi perubahan rute dengan adanya indikasi perubahan *node intermediate* sebagai *next hop*. Jika *node intermediate* = 0, croute = 1 dan *metric*<*cmetric* maka akan terjadi perubahan rute dari rute yang tadinya melewati *node intermediate* 0 menjadi rute yang melewati *node intermediate* 1 dan akan dikirimkan *triggered update* untuk mengumumkan kepada node lain bahwa ada perubahan rute yang diputuskan oleh *decision node*. Jika *node intermediate* = 0, croute = 1 dan *metric*>*cmetric* maka tidak terjadi perubahan rute sehingga croute tetap melalui *node intermediate* 1. Jika *node intermediate* = 1, croute = 0 dan *metric*>*cmetric* maka akan terjadi perubahan rute dari rute melewati *node intermediate* 1 ke rute yang melewati *intermediate node* 0 sehingga croute = 0. Jika *node intermediate* = 1, croute = 0 dan *metric*>*cmetric* maka tidak ada perubahan rute jadi croute = 1. Jika *node intermediate* 1 dan croute = 1 maka update nilai *metric* menjadi *cmetric* dan croute tidak berubah.



Gambar 1. *Activity diagram* proses pemilihan rute pada Babel.

3.2. Pemodelan dengan *transition system* untuk proses pemilihan rute protokol routing Babel

Transition system direpresentasikan dengan *state* awal, *state* tujuan, *guard*, dan aksi. Pada pemodelan protokol routing Babel, penulis menggunakan sebelas *state* untuk mengabstraksikan sistem dalam model berhingga yaitu S_1,S_2, S_3, S_4, S_5,S_6,S_7, S_8, S_9,S_{10}, dan S_{11}. Perpindahan *state* dari proses pemilihan rute pada protokol routing Babel menunjukkan adanya dependensi. Contohnya adalah paket *update* yang diterima akan dihitung *metric*nya jika *update*-nya adalah *feasible*. Jika *update*-nya tidak *feasible*, maka paket *update* tersebut akan diabaikan. Transisi antara satu *state* dengan *state* lainnya dikatakan sebagai transisi yang kondisional karena untuk mencapai suatu *state* tertentu, syarat tertentu harus dipenuhi. Transisi tersebut dinamakan sebagai*conditional transition system. Conditonal transition*

*system*yang dijelaskan dalam buku *Principles of Model Checking* [13] dapat dijabarkan dalam*Transition System* (1) sebagai berikut:

$$S_i \xrightarrow{g:\alpha} S_i' \tag{1}$$

dengan, S_i merupakan *state* awal, g disebut *guard* yaitu syarat yang harus dipenuhi oleh suatu transisi, α merupakan aksi yang dilakukan ketika *guard* terpenuhi, S_i' merupakan *state* tujuan.

Perilaku dari transisi sistem yang ditunjukkan dalam diagram state pada Gambar 2 adalah sebagai berikut: Sistem transisi bermula dari suatu *state* awal S_i dan bertransisi ke *state* S_i' dengan syarat g. Jika syarat g terpenuhi, maka akan dieksekusi suatu aksi.

Penjelasan proses pemilihan rute pada protokol routing Babel dapat dijabarkan dalam beberapa *transition system* sebagai berikut:

1. S_0 merupakan *state* awal. Ketika ada paket *update* yang datang, maka *update* tersebut akan diperiksa terlebih dulu darimana *original node*-nya (onode) dan apakah update tersebut *feasible* atau tidak. Jika *update* tidak *feasible*, maka update akan diabaikan. Jika update diabaikan maka tidak ada perubahan *state* sehingga *state* tidak berubah, tetap di S_0 seperti diperlihatkan pada *Transition System* (2) dan (3) berikut:

$$S_0 \xrightarrow{onode=B \wedge status=notfeasible:ignoreupdate} S_0 \tag{2}$$

$$S_0 \xrightarrow{onode=B \wedge status=notfeasible:ignoreupdate} S_0 \tag{3}$$

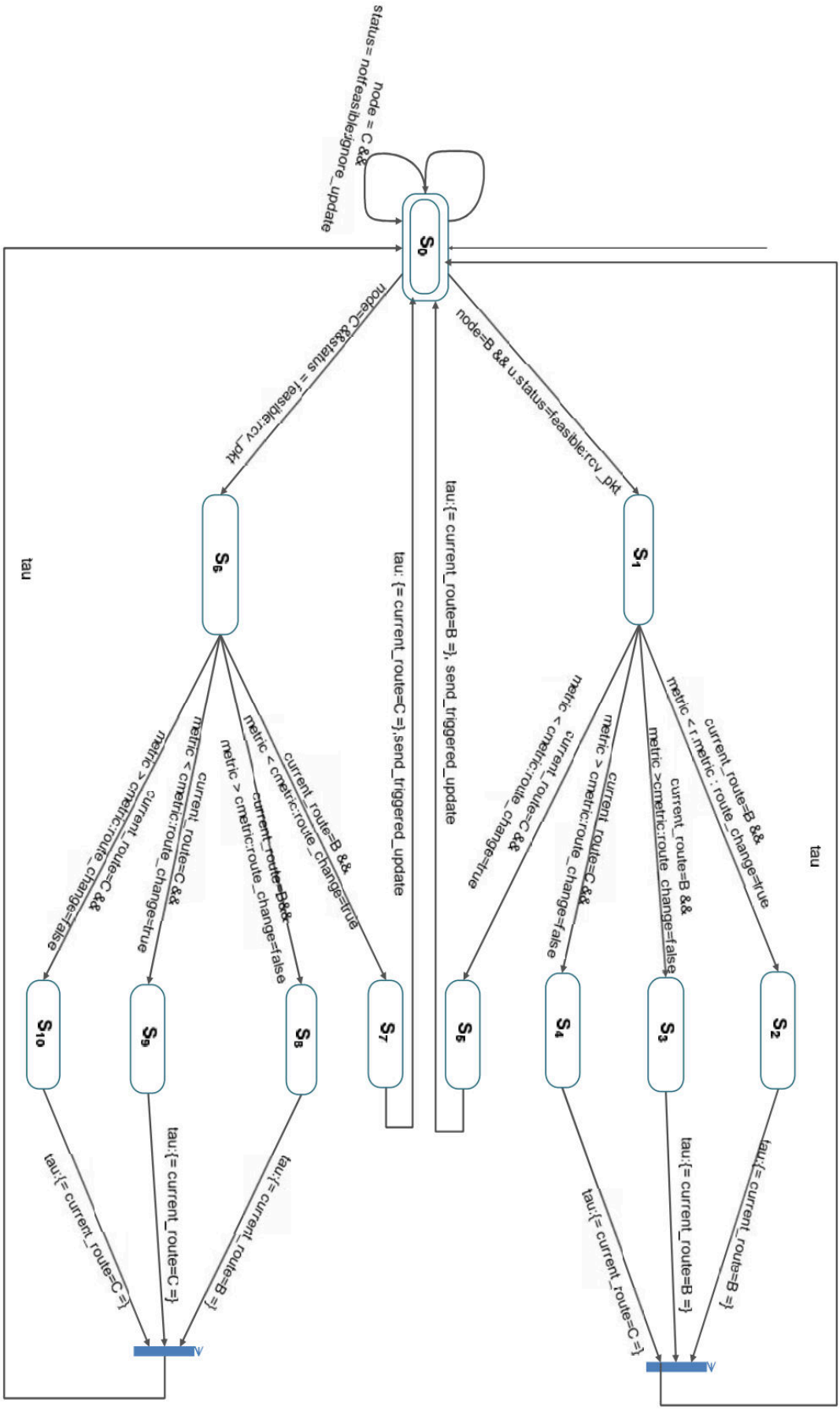
2. Pada *Transition System* (4) dan (5) diperlihatkan jika *update* tersebut *feasible* maka *state* akan berpindah tergantung pada *original node.* Apabila *original node* dari *update* yang *feasible* itu adalah *node intermediate 0* (B), maka akan terjadi perpindahan ke *state* S_2. Namun jika *original node* dari *update* yang *feasible* itu adalah *node intermediate 1* (C), maka akan terjadi perpindahan ke *state* S_6.

$$S_0 \xrightarrow{onode=B \wedge status=feasible:\tau} S_1 \tag{4}$$

$$S_0 \xrightarrow{onode=C \wedge status=feasible:\tau} S_6 \tag{5}$$

3. Jika sudah mencapai *state* S_1, maka dapat dipastikan bahwa paket *update* tersebut *feasible*. Untuk selanjutnya diperiksa *current route* apakah lewat B ataukah C dan dilakukan perbandingan antara *advertised metric* yang dibawa oleh paket *update* dengan *current metric* dari *current route.*
    a. Apabila *current route* adalah B dan *advertised metric*-nya kurang dari *current metric* maka syarat terjadinya perubahan rute terpenuhi, sehingga terjadi aksi *route change = true* dan *state* berpindah ke *state* S_2. Hal ini ditunjukkan dengan *Transition System* (6) berikut:

$$S_1 \xrightarrow{croute=B \wedge adv1metric<cmetric:routechange=true} S_2 \tag{6}$$



Gambar 2. Diagram stateperilaku dari transisi sistem

Apabila *current route* adalah B dan *advertised metric*-nya lebih dari *current metric* maka dikerjakan aksi mengubah nilai *route change = false* dan *state* berpindah ke *state* S_3. Hal ini ditunjukkan dengan *Transition System* (7) berikut:

$$S_1 \xrightarrow{croute=B \wedge adv1metric>cmetric:routechange=false} S_3 \quad (7)$$

b. Apabila *current route* adalah C dan *advertised metric*-nya kurang dari *current metric* maka syarat terjadinya perubahan rute terpenuhi, sehingga terjadi aksi mengubah nilai *route change* menjadi *true* dan *state* berpindah ke *state* S_5. Hal ini ditunjukkan dengan *Transition System* (8) berikut:

$$S_1 \xrightarrow{croute=C \wedge adv1metric<cmetric:routechange=true} S_5 \quad (8)$$

c. Apabila *current route* adalah C dan *advertised metric*-nya lebih dari *current metric* maka dikerjakan aksi mengubah nilai *route change* menjadi 0 dan *state* berpindah ke *state* S_4. Hal ini ditunjukkan dengan *Transition System* (9) berikut:

$$S_1 \xrightarrow{croute=C \wedge adv1metric>cmetric:routechange=false} S_4 \quad (9)$$

4. Ketika kita mencapai *state* S_2, S_3, dan S_4, tidak ada *guard* yang harus dipenuhi maka dengan otomatis kita akan berpindah ke *state* berikutnya yaitu kembali ke *state* S_0 dan tidak ada aksi yang dilakukan pada saat perpindahan ke *state* S_0. Hal ini ditunjukkan dengan *Transition System* (10), (11), dan (12) berikut:

$$S_2 \xrightarrow{true:croute=B} S_0 \quad (10)$$

$$S_3 \xrightarrow{true:croute=B} S_0 \quad (11)$$

$$S_4 \xrightarrow{true:croute=C} S_0 \quad (12)$$

5. Ketika kita mencapai *state* S_5, tidak ada *guard* yang harus dipenuhi maka secara otomatis kita akan berpindah ke *state* berikutnya yaitu kembali *state* S_0 dan ada aksi yang harus dilakukan sembari melakukan perpindahan ke *state* S_0 yaitu mengirim *triggered update*. Hal ini ditunjukkan dengan *Transition System* (13) berikut:

$$S_5 \xrightarrow{true:croute=B,sendtriggeredupdate} S_0 \quad (13)$$

6. Jika sudah mencapai *state* S_6, maka dapat dipastikan bahwa paket *update* tersebut *feasible*. Untuk selanjutnya diperiksa *current route* apakah B ataukah C dan dilakukan perbandingan antara *advertised metric* yang dibawa oleh paket *update* dengan *current metric* dari *current route*.
   a. Apabila *current route* adalah B dan *advertised metric*-nya kurang dari *current metric* maka syarat terjadinya perubahan rute terpenuhi, sehingga terjadi aksi mengubah nilai *route change* menjadi *true* dan *state* berpindah ke *state* S_7. Hal ini ditunjukkan dengan *Transition System* (14) berikut:

$$S_6 \xrightarrow{croute=B \wedge adv2metric<cmetric:routechange=true} S_7 \quad (14)$$

b. Apabila *current route* adalah B dan *advertised metric*-nya lebih dari *current metric* maka *route change=false* dan *state* berpindah ke *state* S_8. Hal ini ditunjukkan dengan *Transition System* (15) berikut:

$$S_6 \xrightarrow{croute=B \wedge adv2metric > cmetric : routechange=false} S_8 \quad (15)$$

c. Apabila *current route* adalah C dan *advertised metric*-nya kurang dari *current metric* maka syarat terjadinya perubahan rute terpenuhi, sehingga *route change=true* dan *state* berpindah ke *state* S_9. Hal ini ditunjukkan dengan *Transition System* (16) berikut:

$$S_6 \xrightarrow{croute=C \wedge adv2metric < cmetric : routechange=true} S_9 \quad (16)$$

d. Apabila *current route* adalah C dan *advertised metric*-nya lebih dari *current metric* maka dikerjakan aksi mengubah nilai *route change* menjadi 0 dan *state* berpindah ke *state* S_{10}. Hal ini ditunjukkan dengan *Transition System* (17) berikut:

$$S_6 \xrightarrow{croute=C \wedge adv2metric > cmetric : routechange=true} S_{10} \quad (17)$$

7. Ketika kita mencapai *state* S_8, S_9, dan S_{10}, tidak ada *guard* yang harus dipenuhi maka dengan otomatis kita akan berpindah ke *state* berikutnya yaitu kembali ke *state* S_0 dan tidak ada aksi yang dilakukan pada saat perpindahan ke *state* S_0. Hal ini ditunjukkan dengan *Transition System* (18), (19), dan (20) berikut:

$$S_8 \xrightarrow{true : croute=B} S_0 \quad (18)$$

$$S_9 \xrightarrow{true : croute=C} S_0 \quad (19)$$

$$S_{10} \xrightarrow{true : croute=C} S_0 \quad (20)$$

8. Ketika kita mencapai *state* S_7, tidak ada *guard* yang harus dipenuhi maka secara otomatis kita akan berpindah ke *state* berikutnya yaitu kembali *state* S_0 dan ada aksi yang harus dilakukan sembari melakukan perpindahan ke *state* S_0 yaitu mengirim *triggered update.* Hal ini ditunjukkan dengan *Transition System* (21) berikut:

$$S_7 \xrightarrow{true : croute=C, sendtriggerdeupdate} S_0 \quad (21)$$

5. Kesimpulan

Deskripsi menggunakan *activity diagram* dapat menggambarkan alur proses pemilihan rute dari proses pemilihan rute dari protokol routing Babel. *Penggunaan transition system* dapat lebih merunut atau melihat satu persatu kemungkinan *state* yang dapat dilalui melalui pelacakan secara berurut. Penelitian ini masih hanya menggunakan input dari dua *intermediate node*, satu *decision node*, dan syarat perubahan rute adalah nilai *metric* saja. Pada penelitian selanjutnya bisa ditambahkan*node intermediate*. Penelitian selanjutnya juga bisa menambahkan syarat lain sebagai syarat perubahan rutenya.

Daftar Pustaka

[1] Jacquet P, Clausen T. *Optimized Link State Routing Protocol (OLSR)*. RFC 3626. 2003.
[2] Perkins C, Belding-Royer E. *Ad hoc On-Demand Distance Vector (AODV) Routing*. RFC 3561. Juli 2003.
[3] Lougheed K, Rekhter Y. *Border Gateway Protocol (BGP)*. RFC 1105. Juni 1989.
[4] Hedrick C. *Routing Information Protocol (RIP).* RFC 1058. Juni 1988.
[5] Moy J. OSPF Version 2. RFC 1247. Juli 1991.
[6] Savage D, Ng J, Moore S, Slice D, Paluch P and White R. *Cisco's Enhanced Interior Gateway Routing Protocol (EIGRP)*. RFC 7868. Mei 2016.
[7] Perkins Charles E, Bhagwat Pravin. *Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers*. In Proceedings of the conference on Communications architectures, protocols and applications (SIGCOMM '94). 1994. ACM. New York, NY, USA.
[8] Kulla Elis, Hiyama Masahiro, Ikeda Makoto, Barolli Leonard. *Performance comparison of OLSR and BATMAN routing protocols by a MANET testbed in stairs environment*. In Proceedings of Computers & Mathematics with Applications, Volume 63, Issue 2, 2012: 339-349.
[9] Chroboczek J. *The Babel Routing Protocol*. RFC 6126. April 2011.
[10] Abolhasan M, Hagelstein B, Wang J. *Real-world Performance of Current Proactive Multi-hop Mesh Protocols*. In Proceedings of the 15th Asia-Pacific Conference on Communications, APCC. 2009. IEEE Press, Piscataway, NJ, USA.
[11] Seidl M, Scholz M, Huimer C, Kappel G. UML@Classroom:An Introduction to Object-Oriented Modeling. 2015.
[12] Thramboulidis K, Mikroyannidis A. *Using UML for the Design of Communication Protocols: The TCP case study*. International Conference on Software, Telecommunications and Computer Networks, SoftCOM. Januari 2003.
[13] Baier C, Katoen J. *Principles of Model Checking*. 2008. MIT Press. London, England.
[14] Jonglez B, Boutier M, Chroboczek J. *A delay-based routing metric*. Computing Research Repository (CoRR). arXiv:1403.3488v1. Maret 2014.