

**APLIKASI PETRI NET PADA SISTEM PEMBAYARAN TAGIHAN LISTRIK  
PT. PLN (Persero) RAYON AMBON TIMUR**  
(*The Application of Petri Net in Electricity Bill Payment System of  
PT. PLN (Persero) Rayon Ambon Timur*)

**FREYA N. WATTIMENA<sup>1</sup>, THOMAS PENTURY<sup>2</sup>, YOPI A. LESNUSSA<sup>3</sup>**  
<sup>1,2,3</sup> *Staf Jurusan Matematika FMIPA UNPATTI*  
Jl. Ir. M. Putuhena, Kampus Unpatti, Poka-Ambon  
e-mail: ya.lesnussa@staff.unpatti.ac.id

**ABSTRACT**

Petri Net is a model of Mathematics used in the system event discri to illustrate certain event. One application of Petri Nets is to sketch the entrance occurs in a place like the public service system. This reserce will simply describe the use of Petri Net to sketch the entrance occurs in electricity bill payment system of PT.PLN (Persero) Rayon Ambon Timur by the number of place is 7 and the transition is 8, and to determine the matrix representation and the *coverability tree* of the model.

**Keywords:** *Eigenvalues, Equilibrium point, Jacobian-matrix, Rabies, SIR-models.*

**PENDAHULUAN**

Adanya keterbatasan sumber daya dalam suatu sistem ekonomi dan dunia usaha (bisnis) menyebabkan orang-orang, barang-barang maupun komponen-komponen harus menunggu untuk mendapatkan jasa pelayanan. Dalam kasus tersebut barisan tunggu yang terjadi sering disebut dengan antrian (*queues*).

Antrian terjadi bilamana banyaknya pelanggan yang akan dilayani melebihi kapasitas layanan yang tersedia. Sistem tersebut seringkali dan hampir terlihat setiap hari, seperti antrian kendaraan mobil-mobil yang memasuki tempat usaha pencucian mobil, antrian para nasabah di bank untuk mendapatkan pelayanan dari para teller, antrian para pelanggan di suatu swalayan untuk melakukan pembayaran di kasir.

Penulis mengambil salah satu contoh sistem antrian yaitu sistem antrian pada loket pembayaran tagihan listrik. Diketahui bahwa listrik merupakan salah satu kebutuhan pokok masyarakat untuk mempermudah dalam menjalani kegiatan sehari-hari mulai dari instansi yang paling kecil yaitu rumah tangga sampai yang paling besar meliputi industri-industri terbesar sekalipun.

PT. PLN (Persero) adalah perusahaan penyedia jasa listrik yang berusaha melayani konsumen dengan sebaik-baiknya. Untuk itu masyarakat sebagai konsumen diharuskan membayar tagihan listrik setiap bulannya. Proses pembayaran tagihan listrik ini dapat dilakukan di

kantor-kantor PLN terdekat atau koperasi-koperasi yang telah ditunjuk, atau bisa dilakukan di bank-bank yang telah menjamin kerjasama dengan PT. PLN bahkan sekarang pelanggan dapat melakukan pembayaran melalui ATM (Anjungan Tunai Mandiri).

Sistem pembayaran tagihan listrik pada kantor-kantor PLN tersebut mempunyai tahapan-tahapan yang dimulai dari input sampai dengan output. Sehingga penulis ingin mengubah sistem tersebut ke dalam model matematika dengan menggunakan Petri Net dan mengamati prosesnya pada PT. PLN (Persero) Rayon Ambon Timur.

**TINJAUAN PUSTAKA**

Petri Net pertama kali dikembangkan oleh Carl Adam Petri pada tahun 1962. Ia adalah seorang matematikawan asal Jerman. Pada awal 1960-an ia mendefinisikan tujuan umum model matematika untuk menggambarkan hubungan antara kondisi dan peristiwa (*R.David & H.Alla,1965*).

Petri Net merupakan suatu alat bantu untuk mempelajari sistem. Dengan menggunakan teori Petri Net maka suatu sistem dapat dimodelkan menjadi suatu Jaringan Petri, yang merupakan representasi matematika dari sistem tersebut. Dengan melakukan analisis dari Jaringan Petri tersebut diharapkan dapat diperoleh

informasi penting tentang struktur dan perilaku yang dinamis dari sistem yang dimodelkan dan mengusulkan peningkatan-peningkatan serta perubahan-perubahan yang diperlukan. Jadi perkembangan dari teori Jaringan Petri didasarkan pada pemakaian Jaringan Petri dalam memodelkan dan merancang sistem (J.L.Peterson,1981).

Petri Net merupakan salah satu alat untuk memodelkan *system event discrete*. *System event discrete* adalah sistem dimana ruang keadaan dari sistem tersebut diuraikan oleh himpunan diskrit  $\{0,1,2,\dots\}$  dan transisi keadaan hanya diamati pada titik diskrit dalam waktu. Jadi, berubahnya keadaan pada *system event discrete* diakibatkan oleh terjadinya *event* (Cassandras, 1993).

Petri Net merupakan *direct bipartite graph* (graph berarah) yang memiliki 2 *node* yang dinamakan sebagai *Place* dan *Transisi*. *Arc* dilambangkan dengan anak panah (*arrow*), *Place* dilambangkan sebagai lingkaran (*circles*) dan *Transisi* dilambangkan sebagai persegi panjang (*bars*). *Arc* secara tidak langsung menghubungkan *place* dengan *place* atau transisi dengan transisi, tetapi menghubungkan *place* dengan transisi atau transisi dengan *place*. Tiap *place* dapat berisi satu atau beberapa *token* yang dilambangkan dengan bulatan kecil (*dots*) yang merupakan material atau bahan yang ditrasfer dalam satu sistem Petri Net.

**Definisi 1**

Petri Net terdiri dari 4-tuple  $(P, T, A, w)$  dengan (Cassandras, 1993) :

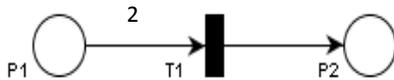
- $P$  : himpunan berhingga *Place*,  $P = \{p_1, p_2, \dots, p_n\}$
- $T$  : himpunan berhingga transisi,  $T = \{t_1, t_2, \dots, t_n\}$
- $A$  : himpunan *arc*,  $A \subseteq (PxT) \cup (TxP)$
- $w$  : fungsi bobot ,  $w : A \rightarrow \{1,2,3, \dots\}$

Dalam membahas representasi Petri Net secara grafik akan digunakan notasi  $I(t_j)$  dan  $O(t_j)$  yang masing-masing menyatakan himpunan *place* input dan *place* output ke transisi  $t_j$ . Secara matematis definisi tersebut dapat ditulis menjadi persamaan berikut (Cassandras, 1993).

$$I(t_j) = \{p_i : (p_i, t_j) \in A\}$$

$$O(t_j) = \{p_i : (t_j, p_i) \in A\}$$

**Contoh 1:**



Gambar 1. Petri Net Sederhana

Perhatikan gambar 1, terdapat dua *place* pada Petri Net tersebut yaitu  $p_1$  dan  $p_2$  ditulis  $P = \{p_1, p_2\}$ . Untuk menyatakan bahwa terdapat sebuah transisi yaitu  $t_1$  maka ditulis  $T = \{t_1\}$ . *Arc* dinyatakan dengan pasangan berurutan, elemen pertama menyatakan asal dan elemen kedua menyatakan tujuan misalnya *arc* dari *place*  $p_1$  ke transisi  $t_1$  ditulis  $(p_1, t_1)$  dan  $(t_1, p_2)$  menyatakan *arc* dari transisi  $t_1$  ke *place*  $p_2$  yang secara lengkap ditulis  $A = \{(p_1, t_1), (t_1, p_2)\}$ . Bobot *arc* dari *place*  $p_1$  ke transisi  $t_1$  adalah 2 yang digambarkan dengan dua buah *arc* sehingga dapat ditulis  $w(p_1, t_1) = 2$  dan bobot dari

transisi  $t_1$  ke *place*  $p_2$  adalah 1 yaitu  $w(t_1, p_2) = 1$ , pada contoh ini  $I(t_1) = \{p_1\}$  dan  $O(t_1) = \{p_2\}$ .

Perlu membedakan antara Petri Net yang *pure* dan *impure*. Petri Net disebut *pure* jika tidak ada *place* yang menjadi *input* dan *output* untuk suatu transisi. Jika terdapat *place* yang dapat menjadi input sekaligus output untuk transisi tertentu maka Petri Net dikatakan *impure*, yang secara formal ditulis

$$p_i \in P, t_j \in T \ni \{(p_i, t_j), (t_j, p_i)\} \subseteq A$$

Jelas bahwa Petri Net pada Gambar 1 adalah *pure* karena tidak ada *place* yang menjadi input sekaligus output untuk suatu transisi (Cassandras, 1993).

Transisi pada Petri Net menyatakan *event* pada *system event discrete* dan *place* merepresentasikan kondisi agar *event* dapat terjadi. Diperlukan mekanisme untuk mengindikasikan apakah kondisi telah terpenuhi. *Token* adalah sesuatu yang diletakan di *place* yang menyatakan terpenuhi tidaknya suatu kondisi. Secara grafik *token* digambarkan dengan *dot* dan diletakan di dalam *place*.

**Definisi 2.2.**

Penanda (*marking*)  $m$  pada Petri Net adalah fungsi  $m: P \rightarrow \{0,1,2, \dots\}$  (Cassandras, 1993).

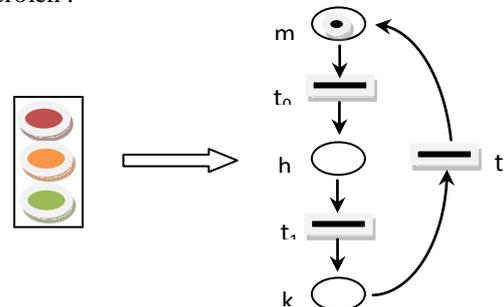
Penanda dinyatakan dengan vektor yang berisi bilangan bulat tak negatif yang menyatakan jumlah token yaitu  $m = [m(p_1), m(p_2), \dots, m(p_n)]^T$ . Jumlah elemen  $m$  sama dengan banyaknya *place*  $p_i$ ,  $m(p_i) \in \{0,1,2, \dots\}$ .

**Definisi 2.3**

Petri Net bertanda (*marked*) terdiri dari 5-tuple  $(P, T, A, w, m_0)$  dimana  $(P, T, A, w)$  adalah Petri Net dan  $m_0$  adalah penanda awal (Cassandras, 1993).

**Contoh 2:**

Lampu lalu lintas pada jalan raya yang dibuat dalam model Petri Net. Lampu berwarna Merah (m), Kuning (k), Hijau (h). Jika diubah dalam bentuk Petri Net maka diperoleh :

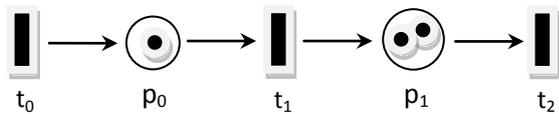


Gambar 2. Petri Net bertanda (*marked*)

**Transisi Sumber dan Transisi Akhir**

Suatu transisi  $T$  tanpa *input place* dinamakan transisi sumber (*source transition*). Suatu transisi  $T$  tanpa *output place* dinamakan dengan transisi akhir (*sink transition*). Suatu transisi dapat *difire* (diselesaikan prosesnya) jika sistem Petri Net mulai dijalankan. Jika suatu transisi *difire* maka ada sejumlah *token* yang dipindahkan dari *input place*

sesuai dengan aturan dari Petri Net sendiri, tapi tidak ada *token* yang diproduksi pada *output*. Jika suatu transisi *difire*, ada sejumlah *token* yang berpindah ke suatu *place* dan jika tidak ada token yang berpindah dari suatu *place* maka transisi tersebut disebut *source* (sumber).



Gambar 3. Transisi sumber ( $t_0$ ), transisi akhir ( $t_2$ ), dan  $m_0 = [1 \ 2]$

**Transisi Enable**

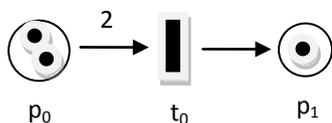
Jika semua keadaan yang diperlukan sudah terpenuhi maka transisi dapat terjadi. Dalam hal ini keadaan merupakan *place input* dari transisi. Bobot *arc* dari *place input* ke transisi menunjukkan *token* minimum di *place* agar transisi *enable*. Jika semua *place input* mempunyai *token* lebih dari atau sama dengan jumlah *token* minimum yang dibutuhkan maka transisi *enable*.

**Definisi 2.4.**

Transisi  $t_j \in T$  pada Petri Net bertanda *enable* jika,  $m(p_i) \geq w(p_i, t_j), \forall p_i \in I(t_j)$ . (Cassandras, 1993).



Gambar 4(a). Transisi tidak *enable*



Gambar 4(b). Transisi yang *enable*

Gambar 4(a). merupakan contoh transisi yang tak *enable*. Jelas bahwa  $I(t_0) = \{p_0\}$ .  $m(p_0) = 1$  dan  $w(p_0, t_0) = 2$ . Transisi  $t_0$  tidak *enable* karena  $1 = m(p_0) < w(p_0, t_0) = 2$ . Transisi  $t_0$  *enable* jika jumlah *token* pada *place*  $p_0$  lebih dari atau sama dengan 2. Pada Gambar 4(b) terlihat bahwa  $m(p_0) = 2$  sehingga transisi  $t_0$  pada Petri Net tersebut *enable*. Maka dapat disimpulkan bahwa agar transisi  $t_j$  *enable* maka jumlah *token* pada *place*  $p_i$  paling sedikit sebesar bobot *arc* yang menghubungkan  $p_i$  ke  $t_j$ . Kenyataan ini sesuai dengan definisi transisi *enable* yang telah dituliskan sebelumnya.

Perlu diperhatikan bahwa jika tidak ada *arc* dari *place*  $p_i$  ke  $t_j$  maka  $w(p_i, t_j) = 0$ . Dan ingat jumlah *token* pada *place* merupakan bilangan bulat tak negatif yaitu  $m(p_i) \geq 0$  sehingga berapapun nilai  $m(p_i)$  pernyataan  $m(p_i) \geq w(p_i, t_j)$  selalu benar.

**Dinamika Petri Net.**

Jika Petri Net digunakan untuk memodelkan sistem dinamik *event discrete*, seharusnya Petri Net dilengkapi dengan mekanisme yang mirip dengan transisi keadaan (*state transition*) pada automata. Mekanisme ini berupa

menjalankan *token* melewati jaringan (*net*) ketika transisi menjadi *enable* dan proses ini mengubah keadaan Petri Net.

Hanya transisi *enable* yang dapat *difire*. Transisi *difire* saat *event* yang dinyatakan oleh transisi terjadi. Berikut ini adalah proses yang terjadi pada pemfirean transisi. Semua *token* di *place input* dikurangi atau diambil sebanyak bobot *arc* yang menghubungkannya. Berdasarkan Definisi 2.4 Jumlah *token* di *place input* setelah dikurangi adalah bilangan bulat tak negatif. *Token* di *place* ditambahkan sebanyak bobot *arc* yang menghubungkannya.

**Representasi Petri Net dengan menggunakan matriks.**

Pada bagian ini akan dikaji representasi Petri Net dalam notasi matriks. Hal ini bertujuan untuk memudahkan dalam implementasi, Petri Net dapat direpresentasikan dalam dua matriks yang disebut *backward incidence* dan *forward incidence*. Kedua matriks ini masing-masing berukuran  $n \times m$  dengan  $n$  adalah jumlah *place* dan  $m$  adalah jumlah transisi. Elemen matriks ini adalah bilangan bulat tak negatif.

Elemen pada matriks *backward incidence* merupakan bobot *arc* yang menghubungkan *place* ke transisi sedangkan elemen pada matriks *forward incidence* merupakan bobot *arc* yang menghubungkan transisi ke *place*. Jika tidak ada *arc* yang menghubungkan *place* ke transisi maka *bobot arc* diisi nol.

**Definisi 2.5**

Matriks *backward* dan *forward incidence* yang mempresentasikan Petri Net adalah matriks berukuran  $n \times m$  dengan elemen baris ke- $i$ , kolom ke- $j$  adalah

$$A_b(i, j) \stackrel{\text{def}}{=} w(p_i, t_j), A_f(i, j) \stackrel{\text{def}}{=} w(t_j, p_i).$$

Salah satu kegunaan matriks *backward incidence* adalah menentukan transisi yang *enable*. Jika  $(p_i)$  bukan merupakan *place input* dari transisi ( $t_j$ ) yaitu  $p_i \notin I(t_j)$  maka *bobot arc* dari *place* ( $p_i$ ) ke transisi ( $t_j$ ) adalah nol karena tidak ada *arc* yang menghubungkannya, ditulis  $w(p_i, t_j) = 0$ . Sehingga dapat ditulis dalam notasi vektor berikut.

$$\begin{aligned} m([p_1, \dots, p_n]^T) &\geq w([p_1, \dots, p_n]^T, t_j) \\ &= A_b(:, j) \\ &= A_b e_j \end{aligned} \tag{i}$$

Dengan  $A_b(:, j)$  menunjukkan kolom ke- $j$  dari matriks  $A_b$  dan  $e_j$  merupakan kolom ke- $j$  matriks identitas berorder  $m$ . Dari persamaan (i) maka transisi yang *enable* dapat dilakukan dengan mencari kolom dari matriks *backward incidence* yang kurang dari atau sama dengan vektor keadaan, sehingga dapat ditulis secara ringkas menjadi

$$m \geq A_b e_j \tag{i}$$

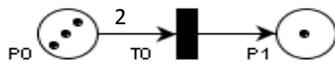
dan

$$m' = m + Au \tag{ii}$$

Dengan  $u$  merupakan vektor kolom yang mempunyai elemen sebanyak  $m$  yaitu diperoleh dari kolom identitas.

**Contoh 3:**

Menentukan transisi yang *enable* setelah transisi  $t_1$  *difire* !!



Gambar 5. Sebelum transisi  $t_1$  *difire*

Penyelesaian :

Diketahui :  $i = 2$  place,  $j = 1$  transisi.  
 $m_0 = [3 \ 1]^T$ ,  $A_b = [2 \ 0]^T$ ,  $A_f = [0 \ 1]^T$   
 $A = A_f - A_b$   
 $= [0 \ 1]^T - [2 \ 0]^T$   
 $= [-2 \ 1]^T$

Dengan menggunakan persamaan (iii) diperoleh  $m_1 = m_0 + Au$

$$m_1 = \begin{bmatrix} 3 \\ 1 \end{bmatrix} + \begin{bmatrix} -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

**Contoh 4 :**

Terdapat 5 place dan 3 transisi pada Petri Net sehingga  $n=5$  dan  $m=3$ . Matriks *backward incidence* dan *forward incidence* masing-masing berukuran  $5 \times 3$  sebagai berikut

$$A_b = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{dan} \quad A_f = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Keadaan awal Petri Net adalah  $m_0 = [2 \ 2 \ 0 \ 0 \ 1]^T$

Transisi  $t_1$  *enable* karena  $m_0 \geq A_b(:,1)$  sedangkan transisi  $t_2$  dan  $t_3$  tidak *enable* karena  $m_0 \not\geq A_b(:,j)$  untuk  $j=2,3$ . Selanjutnya, dihitung matriks *incidence* yang dapat digunakan untuk menentukan keadaan berikutnya.

$$A = A_f - A_b = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & -1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$

Untuk menentukan keadaan berikutnya, gunakan persamaan (iii). Transisi yang *difire* adalah  $t_1$  karena transisi tersebut yang *enable*.

$$m_1 = m_0 + Au = \begin{bmatrix} 2 \\ 2 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & -1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Sehingga diperoleh  $m_1 = [1 \ 1 \ 1 \ 1 \ 1]^T$ . Karena  $m_1 \geq A_b(:,j)$ ,  $1 \leq j \leq 3$ , jadi yang *enable* adalah  $t_1, t_2, t_3$ .

**Liveness dan Deadlocks**

Jika membahas mengenai Petri Net pasti sering dijumpai istilah *deadlock* yang secara mudah berarti keadaan dimana tidak ada transisi yang *enable*. *Deadlock* dapat disebabkan persaingan memperoleh *resource*. Ketika semua pihak tidak memperoleh *resource* yang dibutuhkan maka terjadi *deadlock*. *Resource* dalam Petri net biasanya dinyatakan dengan *token* dan pihak yang bersaing memperoleh *token* adalah transisi. Dengan demikian dapat disimpulkan bahwa *deadlock* terjadi ketika transisi tertentu atau himpunan transisi pada Petri net tidak dapat *difire*.

Transisi yang tidak berhubungan dengan *deadlock* disebut *live*. Perhatikan bahwa transisi yang *live* tidak harus *enable*. Istilah *liveness* dapat diartikan dengan transisi yang mungkin *enable*. Idealnya setiap transisi pada Petri net dapat *difire* setelah beberapa *pfifirean*, hal ini menjamin *deadlock* tidak terjadi.

**Coverability Tree**

*Coverability tree* merupakan teknik yang digunakan untuk menyelesaikan beberapa aspek analisis pada *system event discrete*. *Coverability tree* dapat dibangun dari Petri Net dengan keadaan awal. Keadaan awal Petri Net didefinisikan sebagai *node root*. Anak dari *node root* merupakan keadaan yang dapat dicapai dari keadaan awal dengan *memfire* sebuah transisi. Keadaan-keadaan ini dihubungkan ke *node root* dengan *edge*. Setiap *edge* pada *coverability tree* mempunyai bobot sebuah transisi yaitu transisi yang *difire* untuk mencapai keadaan tersebut.

*Coverability tree* dari Petri Net  $(P, T, A, w)$  dapat dinyatakan dengan 3-tuple yaitu  $(S, E, v)$  yang masing-masing menyatakan himpunan keadaan, himpunan *edge* dan fungsi bobot. Anggota dari himpunan keadaan merupakan *n-tuple* dengan  $n$  adalah jumlah *place* pada Petri Net. Selain berupa bilangan bulat tak negatif, keadaan pada *coverability tree* juga bisa berupa  $\omega$ . Untuk membedakan dengan Petri Net maka pada *coverability tree* digunakan istilah *edge*.

**Definisi 2.6**

*Coverability tree* untuk Petri Net  $(P, T, A, w)$  dinyatakan dengan 4-tuple  $(S, E, T, v)$  dimana :

- $S$  : himpunan berhingga keadaan (*state*),  $S \in \{\omega, 0, 1, \dots\}^n$
- $E$  : himpunan berhingga *edge*,  $E \in S \times S$
- $T$  : himpunan berhingga transisi,  $T = \{t_1, t_2, \dots, t_m\}$
- $v$  : fungsi bobot,  $v : E \rightarrow T$

Beberapa istilah yang digunakan untuk menjelaskan langkah-langkah yang dilakukan membangun *coverability tree* dengan jumlah *node* berhingga.

1. *Node root*, adalah *node* pertama dari *tree* dan merupakan keadaan awal.
2. *Node terminal*, adalah *node* yang menyatakan keadaan *deadlock* yaitu tidak ada transisi yang *enable*.
3. *Node duplicated*, adalah *node* yang sama dengan *node* yang sudah ada di *tree*.
4. Hubungan dominasi *node*. Misalkan notasi  $x = [x(p_1), \dots, x(p_n)]^T$  dan notasi  $y = [y(p_1), \dots, y(p_n)]^T$  menyatakan dua keadaan, yaitu *node* pada *coverability*

tree. Node  $x$  mendominasi  $y$  yang dinotasikan  $x > dy$  jika kedua kondisi berikut terpenuhi.

- (a).  $x(p_i) \geq y(p_i)$  untuk semua  $i = 1, \dots, n$
- (b).  $x(p_i) > y(p_i)$  untuk satu atau lebih  $i = 1, \dots, n$

Untuk membangun *converability tree* dengan jumlah node berhingga dapat dituliskan sebagai berikut :

1. Inisialisasi  $x = x_0$  (keadaan awal)
2. Untuk setiap *node*  $x$ , evaluasi nilai fungsi transisi  $f(x, t_j)$  untuk semua  $t_j \in T$  seperti dibawah ini :
  - (a). Jika  $f(x, t_j)$  tidak terdefinisi untuk semua  $t_j \in T$  (tidak ada transisi yang enable pada keadaan  $x$ ) maka  $x$  adalah node terminal.
  - (b). Jika  $f(x, t_j)$  terdefinisi untuk satu atau lebih  $t_j \in T$ , maka hitung keadaan  $x'$  sebagai node baru dengan persamaan  $x' = f(x, t_j)$ .

Jika terdapat *node*  $y$  pada path dari *node* *root*  $x_0$  (termasuk) ke  $x$  sedemikian hingga  $x' > dy$  maka  $x'(p_i) = \omega$  untuk semua  $p_i$  yang memenuhi  $x'(p_i) > y(p_i)$ .

3. Jika semua *node* baru merupakan *node* terminal atau *duplicated* maka berhenti.

**HASIL DAN PEMBAHASAN**

Pada bagian ini akan dijelaskan mengenai model Petri Net untuk salah satu layanan publik yang melibatkan antrian. Salah satu contoh layanan publik yang akan dibuat sistemnya dengan menggunakan model Petri Net adalah sistem pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur.

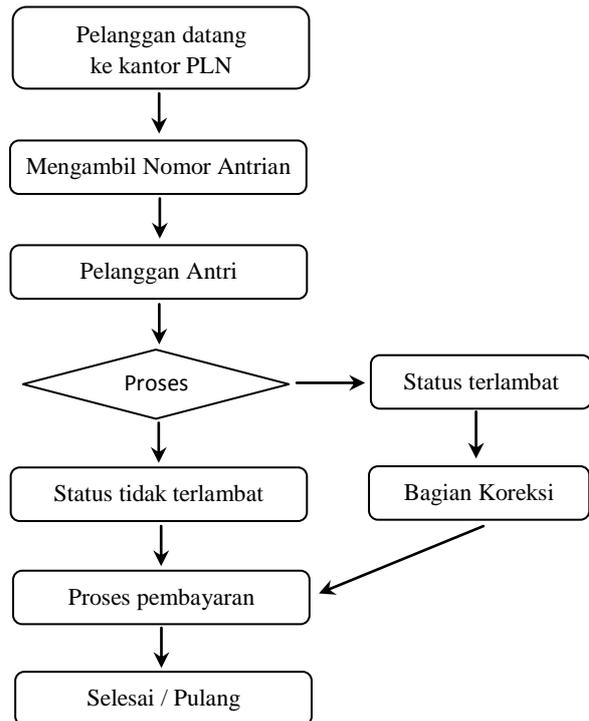
**Analisis sistem pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur**

Salah satu organisasi yang bergerak dalam bidang pelayanan publik adalah Perusahaan Listrik Negara atau PLN. PLN adalah perusahaan penyedia jasa listrik yang berusaha melayani konsumen sebaik-baiknya. Untuk itu pembayaran tagihan listrik setiap bulannya merupakan tanggung jawab atau kewajiban dari setiap pelanggan atau pengguna listrik PLN yang harus dipenuhi.

Tidak semua pelanggan yang mengetahui bagaimana membayar tagihan listrik di kantor-kantor PLN terdekat, bank atau tempat-tempat pembayar tagihan listrik lainnya. Dalam penulisan ini akan diambil salah satu contoh tempat pembayaran tagihan listrik yaitu kantor PLN (Persero) Rayon Ambon Timur. Loker pembayaran tagihan listrik yang ada pada kantor PLN (Persero) Rayon Ambon Timur berjumlah 2 loket, dimana setiap loket terdapat satu petugas (server).

Kondisi yang terjadi di kantor PLN (Persero) Rayon Ambon Timur adalah sering terjadi antrian yang panjang pada tanggal-tanggal tertentu dan besarnya jumlah pelanggan yang datang untuk melakukan pembayaran listrik bisa mencapai ratusan perharinya. Dalam membayar tagihan listrik ada beberapa *event* yang terjadi diantaranya antrian, pengecekan meteran, pengecekan denda, sampai proses membayar. Berikut adalah ilustrasi pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur.

Alur :

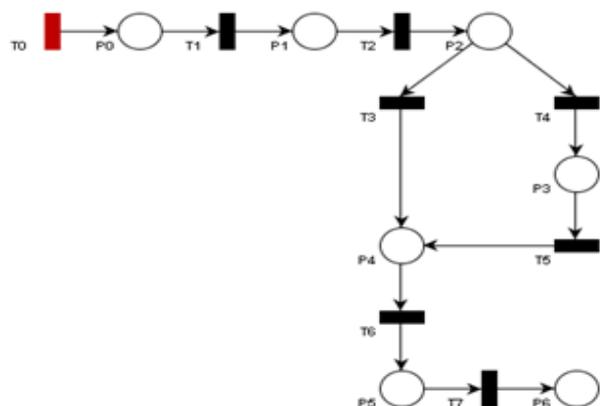


Gambar 7. Alur pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur

**Model jaringan Petri Net sistem pembayaran tagihan listrik PT.PLN (Persero) Rayon Ambon Timur**

Dari hasil analisis sistem pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur, maka permasalahan pembayaran tagihan listrik dapat dimodelkan dengan *software pipe* versi 2.5cr5 untuk memperoleh jaringan Petri Net.

Setiap *event* yang terjadi adalah berkaitan dengan transisi dan *place* yang merupakan kondisi yang harus dipenuhi agar transisi dapat terjadi. Model Petri Net dari sistem pembayaran tagihan listrik pada PLN (Persero) Rayon Ambon Timur seperti pada Gambar 8 yang terdiri dari 7 *place* dan 8 transisi.



Gambar 8. Model Petri Net sistem pembayaran tagihan listrik

Dari Gambar 8 didefinisikan bahwa pembayaran tagihan listrik dilayani pelanggan yang berstatus terlambat dan yang tidak terlambat dalam membayar tagihan listrik. Adapun penjelasan yang diberikan untuk setiap notasi *place* dan transisi sebagai berikut:

**Place:**

- $p_0$  : Pelanggan datang ke kantor PLN
- $p_1$  : Pelanggan mengantri dan siap dilayani
- $p_2$  : Mengecek nomor tagihan
- $p_3$  : Ke bagian koreksi
- $p_4$  : Pengecekan tagihan bulan lalu
- $p_5$  : Proses pembayaran
- $p_6$  : Pulang

**Transisi:**

- $t_0$  : Sumber (*source*)
- $t_1$  : Pelanggan mengambil nomor antrian
- $t_2$  : Pelayanan dimulai
- $t_3$  : Pelanggan tidak berstatus terlambat
- $t_4$  : Pelanggan berstatus terlambat
- $t_5$  : Hitung denda
- $t_6$  : Siap membayar
- $t_7$  : Proses pembayaran selesai

Penentuan keadaan awal Petri Net pada sistem pembayaran tagihan listrik dimulai dengan penempatan satu token pada *place*  $p_0$  sebagai inisialisasi *marking*. Berdasarkan Gambar 8, setiap *place* tidak terisi token ini menyatakan bahwa loket pembayaran tagihan listrik belum dilakukan sampai waktu yang ditentukan. Dari model Petri Net yang diperlihatkan terdapat transisi  $t_0$  yang selalu *enable*, sehingga apabila transisi  $t_0$  *difire* maka token akan selalu bertambah pada *place*  $p_0$ . Berikut prosedur *pfire*-an yang akan dilakukan [ $t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7$ ].

Jika proses dilanjutkan maka transisi yang *enable* adalah transisi  $t_0$  dan  $t_1$ , yang menyatakan bahwa pelanggan datang ke kantor PLN dan mengambil nomor antrian. Setelah transisi  $t_1$  *difire*, maka token dari *place*  $p_0$  akan dipindahkan ke *place*  $p_1$  yaitu suatu keadaan dimana pelanggan sudah mengantri dan siap dilayani. *Place*  $p_0$  akan tetap terisi minimal satu token yang menyatakan terjadinya proses kedatangan yang berlangsung hingga batas waktu yang disediakan. Pada keadaan *place*  $p_1$  yang berisi satu token, maka transisi yang *enable* adalah transisi  $t_0, t_1$  dan  $t_2$ . Setelah transisi  $t_2$  *difire* maka token akan berpindah ke *place*  $p_2$  yang menyatakan suatu keadaan dimana *server* akan mengecek nomor tagihan pelanggan. Keadaan *place*  $p_2$  akan mengakibatkan transisi yang *enable* adalah  $t_0, t_3$  dan  $t_4$ . Transisi  $t_3$  menyatakan bahwa status tidak terlambat yang berarti pelanggan membayar tagihan tepat waktu dan tidak dikenai denda. Transisi  $t_4$  menyatakan status terlambat yang berarti pelanggan terlambat membayar tagihan dan akan dikenai denda.

Pada saat  $t_3$  *difire* maka akan menyebabkan transisi  $t_4$  menjadi tidak *enable* dan *place*  $p_4$  akan berisi satu token. Hal ini berarti mekanisme denda pada *place*  $p_3$  tidak akan dilalui, *place*  $p_4$  yang berisi satu token menyatakan keadaan pengecekan tagihan bulan lalu. Sebaliknya jika transisi  $t_4$  *difire* maka akan menyebabkan

transisi  $t_3$  tidak *enable*, sehingga *place*  $p_3$  akan berisi satu token yang menyatakan keadaan pelanggan harus menuju ruang koreksi. Pada *place*  $p_4$  akan tetap kosong sampai transisi  $t_3$  atau  $t_5$  *difire*. Ini berarti proses perhitungan tagihan bulan lalu akan siap jika denda keterlambatan selesai dihitung. Jika transisi  $t_6$  *difire* maka token akan berpindah dari *place*  $p_4$  ke *place*  $p_5$ .

*Place*  $p_5$  yang berisi satu token ini berarti pelanggan akan melakukan proses pembayaran, dan transisi yang *enable* adalah  $t_6$ . Jika transisi  $t_6$  *difire* maka satu token akan berpindah dari *place*  $p_5$  ke *place*  $p_6$  yang menyatakan bahwa proses pembayaran telah selesai dan pelanggan boleh pulang. Proses yang serupa akan berulang untuk pelanggan berikutnya.

**Model matematika dalam bentuk representasi matriks.**

Untuk merepresentasikan Petri Net dalam bentuk matriks yang dinyatakan dalam matriks *incidence*. Ada dua matriks *incidence* yaitu matriks *forwards incidence* dan matriks *backward incidence*. Elemen matriks *forwards incidence* adalah bobot *arc* yang menghubungkan transisi ke *place*, dan elemen matriks *backwards incidence* adalah bobot *arc* yang menghubungkan *place* ke transisi.

Berdasarkan model Petri Net pada Gambar 8 terdapat 7 *place* dan 8 transisi, sehingga banyak baris ( $n$ ) = 7 dan banyaknya kolom ( $m$ ) = 8. Dengan demikian akan terbentuk matriks *forward* dan *backwards incidence* dengan ordo 7x8. Representasi dalam bentuk matriks *incidence* adalah sebagai berikut:

- o Matriks *forwards incidence* ( $A_f$ )

$$A_f = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- o Matriks *backwards incidence* ( $A_b$ )

$$A_b = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Keadaan awal Petri Net dari Gambar 8 adalah  $m_0 = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$ . Transisi  $t_0$  *enable* karena memenuhi  $m_0 \geq A_b(:,0)$ . Untuk matriks *incidence* akan dihasilkan dari persamaan  $A = A_f - A_b$  dan menghasilkan matriks *incidencenya* sebagai berikut :

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Untuk menentukan keadaan berikutnya maka digunakan persamaan :  $m_1 = m_0 + Ae_1$  dengan  $e_1$  menyatakan vektor kolom yang mempunyai elemen sebanyak  $m$  yaitu diperoleh dari kolom matriks identitas.

Petri Net pada Gambar 8 keadaan awalnya adalah

$$m_0 = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T,$$

sehingga akan dihasilkan keadaan setelah transisi  $t_1$  difire sebagai berikut :

$$m_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Dari perhitungan tersebut diperoleh

$$m_1 = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T.$$

Hal ini berarti bahwa keadaan yang baru transisi yang *enable* adalah  $t_0$  dan  $t_1$ . Untuk penentuan keadaan yang baru selanjutnya, maka  $m_1$  menjadi keadaan awal untuk  $m_2$ , demikian seterusnya.

**Model Coverability Tree**

Untuk membangun *coverability tree* pada Gambar 8, terlebih dulu ditentukan *node root* dari Petri Net yaitu keadaan awal

$$m_0 = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T.$$

Keadaan ini menandakan bahwa *place*  $p_0$  belum terisi oleh token sampai transisi  $t_0$  difire. Untuk menentukan *child* dari *node* tersebut berdasarkan keadaan yang dihasilkan setelah transisi sumber  $t_0$  difire, yaitu pada saat transisi  $t_0$  dan  $t_1$  *enable*. Dari bentuk *coverability tree* yang dihasilkan setelah transisi  $t_0$  difire, maka *place*  $p_0$  akan terisi satu token sehingga transisi  $t_1$  menjadi *enable* dan diperoleh

$$m_1 = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T.$$

Pada saat transisi  $t_1$  difire maka token akan berpindah dari *place*  $p_0$  ke *place*  $p_1$  sehingga transisi yang *enable* adalah  $t_0$  dan  $t_2$  dan diperoleh

$$m_2 = [0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T,$$

ketika transisi  $t_2$  difire maka token akan berpindah dari *place*  $p_1$  ke  $p_2$  dan diperoleh

$$m_3 = [0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0]^T$$

hal ini berarti *node root* mempunyai satu *child*. Pada keadaan *place*  $p_2$  berisi satu token maka transisi  $t_3$  dan  $t_4$

menjadi *enable*,  $m_2$  menjadi keadaan awal untuk menghitung keadan

$$m_4 = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0]^T$$

dan

$$m_5 = [0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0]^T$$

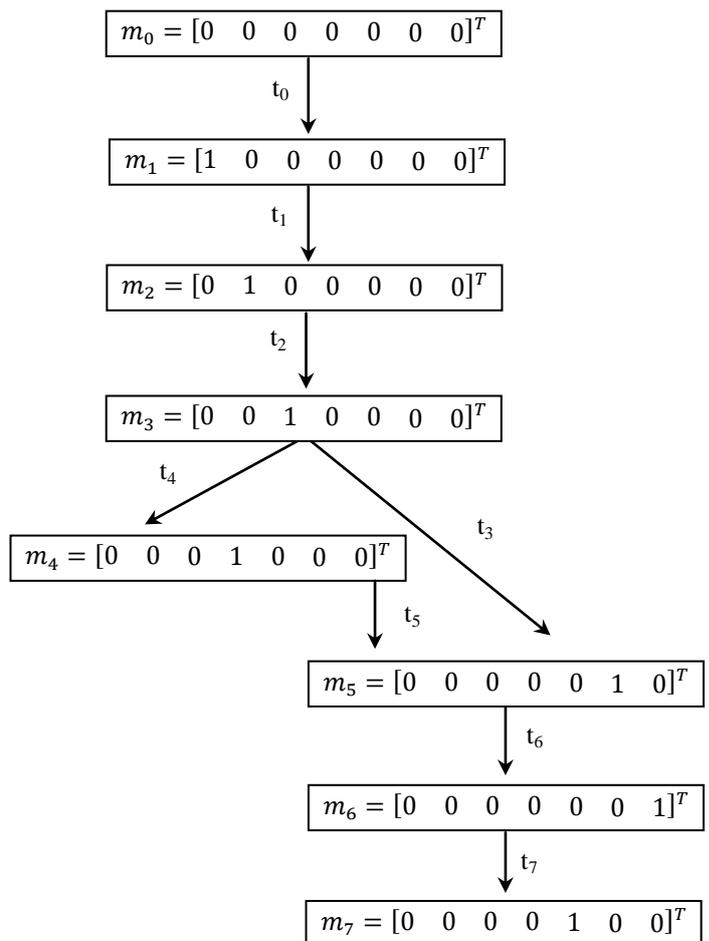
dengan demikian  $m_2$  mempunyai dua *child*. Untuk keadaan  $m_4$  akan menyebabkan transisi  $t_5$  menjadi *enable*, sedangkan pada  $m_5$  menyebabkan transisi  $t_6$  *enable* dan jika transi  $t_5$  difire maka akan menghasilkan  $m_5$ . Jadi  $m_5$  merupakan *child* dari  $m_3$  sekaligus  $m_4$ . Selanjutnya  $m_5$  digunakan sebagai keadaan awal untuk memperoleh  $m_6$  dan diperoleh

$$m_6 = [0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0]^T$$

yaitu keadaan dimana transisi  $t_6$  difire. Dalam hal ini  $m_5$  mempunyai satu *child* yaitu  $m_6$ . Pada saat transisi  $t_7$  *enable* diperoleh dengan memasukkan  $m_6$  sebagai keadaan awal, sehingga diperoleh

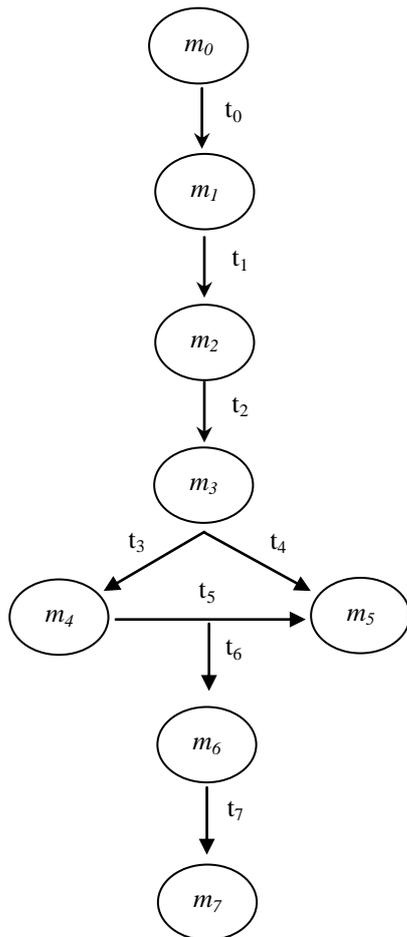
$$m_7 = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0]^T.$$

Dengan demikian akan muncul keadaan awal yang membuat  $t_0$  kembali *enable* dan prosesnya akan berulang seterusnya. Secara lengkap *coverability tree* dari Petri Net pada gambar 7 adalah sebagai berikut:



Gambar 9. *Coverability Tree* dari model jaringan Petri Net Sistem pembayaran tagihan listrik

Atau dapat juga digambarkan sebagai berikut :



Gambar 10. Bentuk lain dari *Coverability Tree*.

## DAFTAR PUSTAKA

- Cassandras, C.G. 1993. *Discret Event System : Modeling and Performance Analysis*, Aksen Associates Incorporated Publishers, Boston.
- David, R dan H. Alla. 1992. *Discret, Continuous and Hybrid Petri Nets*. Springer. Germany.
- Kordache, M dan P.J.Antsalis. 2006. *Supervisory Control of Concurrent System*. Birkhauser. Amerika.
- Moen. A. 2003. *Introduction to Petri Nets-Part I*. Berlin.
- Murata, T. 1989. *Petri Nets : Properties, Analysis and Application*. Proceeding of the IEEE.
- Nurwan dan Subiono. 2008. *Model Petri Net Antrian Klinik Kesehatan Serta Kajian Dalam Aljabar Max Plus*. ITS, Surabaya.
- Peterson.J.L. 1981. *Teori Petri Net dan Pemodelan Sistem*. Prentice Hall. New York.
- Tjahyono, H. 1998. *Aplikasi Jaringan Petri dan Manufaktur Sistem; Pemodelan, Pengendalian dan Analisis Kinerja*. IEEE press.  
<http://daimi.au.dk/Petrinets>, Tgl 30 Mei 2011, Pkl.23.15.

## KESIMPULAN

Berdasarkan hasil pembahasan pada bab 4 di atas, maka dapat disimpulkan beberapa hal sebagai berikut :

1. Penggunaan model Petri Net sangat baik untuk menganalisis sistem antrian pada sistem pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur sehingga dapat diperoleh model matematikanya.
2. Petri Net merupakan suatu alat bantu untuk memodelkan suatu *system event discrete* dalam kasus sistem pembayaran tagihan listrik yang dapat dinyatakan secara matematis dalam bentuk jaringan Petri Net dan Matriks *incidence*.
3. *Coverability tree* merupakan suatu teknik untuk menganalisis beberapa aspek *system even discrete* dan dapat dibangun dari Petri Net untuk kasus sistem pembayaran tagihan listrik PT. PLN (Persero) Rayon Ambon Timur.