# PENANGANAN TOLERANSI KESALAHAN (*FAULT TOLERANCE*) PADA SISTEM PEMBAYARAN ELEKTRONIS DALAM LINGKUP SISTEM TERDISTRIBUSI

**[1]Sunny Arief Sudiro, [2]Abdul Hakim**
*[1,2]STMIK Jakarta STI&K*
*Jl. BRI No. 17 Radio Dalam Kebayoran Baru Jakarta Selatan*
[1]sunnyariefsudiro@ieee.org, [2]hakiem09@gmail.com

**Abstrak**

*Kinerja suatu sistem tidak terlepas dari penanganan kesalahan pada sistem tersebut dan hal ini sangat mempengaruhi tingkat layanan sistem kepada penggunanya. Dari kecepatan layanan misalnya, sistem yang terlalu cepat belum tentu baik bagi pengguna dan sebaliknya. Seandainya terdapat kesalahan atau kegagalan fungsi dari sistem atau sub sistem sudah pasti mempengaruhi tingkat layanan tersebut. Dalam artikel ini dipaparkan mekanisme untuk menangani dan mentolerir serta antisipasi terhadap kesalahan pada suatu sistem atau subsistem pembayaran elektronis menggunakan mesin anjungan tunai mandiri (ATM) dalam suatu jaringan komputer.*

***Kata Kunci**: ATM, sistem, tingkat layanan, toleransi kesalahan*

**Abstract**

*The performance of a system is inseparable from handling errors in the system, and this greatly affects the level of system service to its users. From the speed of service, for example, a system that is too fast is not necessarily good for users and vice versa. If there is an error or malfunction of the system or sub-system it will certainly affect the level of service. In this article, the mechanism for handling and tolerating and anticipating errors in an electronic payment system or subsystem uses an automatic teller machine (ATM) in a computer network is presented.*

***Keywords**: ATM, fault tolerance, service level, system*

## PENDAHULUAN

Kemudahan sistem pembayaran saat ini sudah menjadi keharusan, paling tidak mengurangi sistem antrian pada suatu pempat pembayaran (*teller*). Bagi kepentingan nasabah pun hal ini merupakan daya tarik tersendiri. Berbagai macan teknologi dikembangkan untuk kemudahan ini antara lain adalah teknologi ATM (*Automated Teller Machine*) atau dikenal pula dengan Anjungan Tunai Mandiri.

Penanganan tingkat kesalahan pada sistem yang dapat ditoleransi merupakan faktor utama dalam menentukan tingkat layanan terhadap nasabah atau pengguna. Dengan demikian diperlukan metode dan algoritma yang dapat mendeteksi secara dini kesalahan atau penyimpangan yang terjadi pada sistem dan mengambil tindakan antisipasinya.

Penanganan tingkat kesalahan sangat tergantung pada masing-masing sistem. Mekanisme yang dipaparkan terbatas pada sistem pembayaran elektronis menggunakan mesin ATM NCR *Direct Connect* yang di-kendalikan oleh komputer PC dengan *interface* RS232-NRISO *Native Band* I dan terhubung dalam jaringan komputer berbasiskan TCP/IP.

## METODE PENELITIAN

Toleransi kesalahan adalah metode dinamis yang digunakan untuk menjaga sistem yang saling berhubungan bersama, mempertahankan keandalan, dan ketersediaan dalam sistem terdistribusi. Metode redundansi perangkat keras dan lunak adalah teknik toleransi kesalahan yang dikenal dalam sistem terdistribusi. Metode perangkat keras memastikan penambahan beberapa komponen perangkat keras seperti CPU, tautan komunikasi, memori, dan perangkat I/O sementara dalam metode toleransi kesalahan perangkat lunak, program spesifik dimasukkan untuk mengatasi kesalahan. Mekanisme toleransi kesalahan yang efisien membantu dalam mendeteksi kesalahan dan jika memungkinkan pulih dari itu [1]. Secara tradisional, *fault tolerance* mengacu pada pengembangan subsystem dari komponen *redundant* yang ditempatkan secara paralel [2].

Pada sistem penerbangan terlihat adanya kombinasi komputer *redundant* dan versi *software redundant*, versi *software redundant* dengan spesifikasi yang sama, pada dasarnya mengacu pada pemrograman N versi. Pemrograman ini merupakan pengembangan paradigma *fault tolerance* yang mengeksekusi banyak program (yang dirancang/ditulis secara independen dan menerapkan fungsi yang sama) secara paralel dan mengambil keputusan dari sejumlah hasil yang nilai keluarannya sering berubah. *Software* berkemampuan *fault tolerance* jika dan hanya jika [3]:

1. Program mampu mengkomputasi *acceptable result* meskipun program itu sendiri meng-alami kekurangan dari logika yang tidak tepat, dan
2. Program apakah benar atau tidak, mampu mengkomputasi *acceptable result* meskipun program itu sendiri menerima data ter-korupsi selama eksekusi.

Kunci pokok ada pada *acceptable*, mencakup karakteristik seperti *correctness/ safety*, dan hal ini berdasarkan pada sistem. Interprestasi *software fault tolerance* dihasilkan dari kombinasi prinsip-prinsip *software safety* dan *robustness design*. Hal yang membedakan antara *robustness* dan *fault tolerance* didasarkan pada apakah kondisi yang tidak diharapkan tadi terduga atau tidak terduga. *Robustness* berkaitan dengan masalah yang terduga dan harus diantisipasi sedangkan *fault tolerance* berkaitan dengan masalah tak terduga yang juga harus diantisipasi.

Untuk *software* kritis, pada dasarnya terdapat tiga kondisi yang dihasilkan dari eksekusi program: (1) benar, (2) tidak benar tetapi dapat diterima dan tidak berbahaya, (3) berbahaya. *Software fault tolerance*, mengacu pada kemampuan *software* untuk menghasilkan keluaran yang dapat diterima *'acceptable'* berkaitan dengan status program yang terjadi selama eksekusi. *Software safety* mengacu pada kemampuan *software* menghasilkan keluaran tak berbahaya berkaitan dengan status program selama eksekusi. Keluaran tak berbahaya di-definisikan oleh persyaratan tingkat keamanan sistem. Untuk itu *software safety* menurut pandangan *fault tolerance* adalah tipe khusus dari *software fault tolerance. Fault tolerance* mengacu pada kelompok *output* yang dapat ditolerir sedangkan *software safety* mengacu pada kelompok *output* yang tak dapat ditolerir [3].

Dalam pengembangan *software fault tolerant* ini banyak menggunakan algoritma yang dikenal dengan *Byzantine Fault Tolerant* (*Byzntine General Algorithm*, oleh Lamport tahun 1982) dan banyak dibahas dalam berbagai tulisan dari jurnal sampai bahan tesis. Miquel Castro dan Barbara Liskov banyak melakukan penelitian baik teori maupun *practical*, sehingga terkenal dengan Castro & Liskov's BFT. *Protocols* [4,5]. Aplikasi dari BFT mencakup dari sistem operasi sampai aplikasi berbasiskan web (http *services*) seperti FARGOS/VISTA [6]. Baik komputer yang berdiri sendiri maupun pada suatu jaringan sistem terdistribusi yang saling ketergantungan [7].

Teknologi ATM pada dunia perbankan saat ini terdiri dari beberapa metode. Beberapa ATM dihubungkan melalui perangkat komunikasi dengan komputer pusat yang dikenal sebagai ATM *Controller/Switching* (biasanya komputer kelas mini atau *main frame*). Pendekatan lain adalah mesin-mesin ATM ini dihubungkan dan dikendalikan oleh *Personal Computer* (*PC Base*) [8,9].

## HASIL DAN PEMBAHASAN

Mekanisme pengamatan difokuskan pada komputer jaringan dengan protokol jaringan TCP/IP yang terdiri dari beberapa komputer *client* dan satu komputer *server*, dan pembuatan sistem yang terdiri dari:

a. Algoritma/metode penanganan toleransi kesalahan.

b. Jaringan komputer terbatas

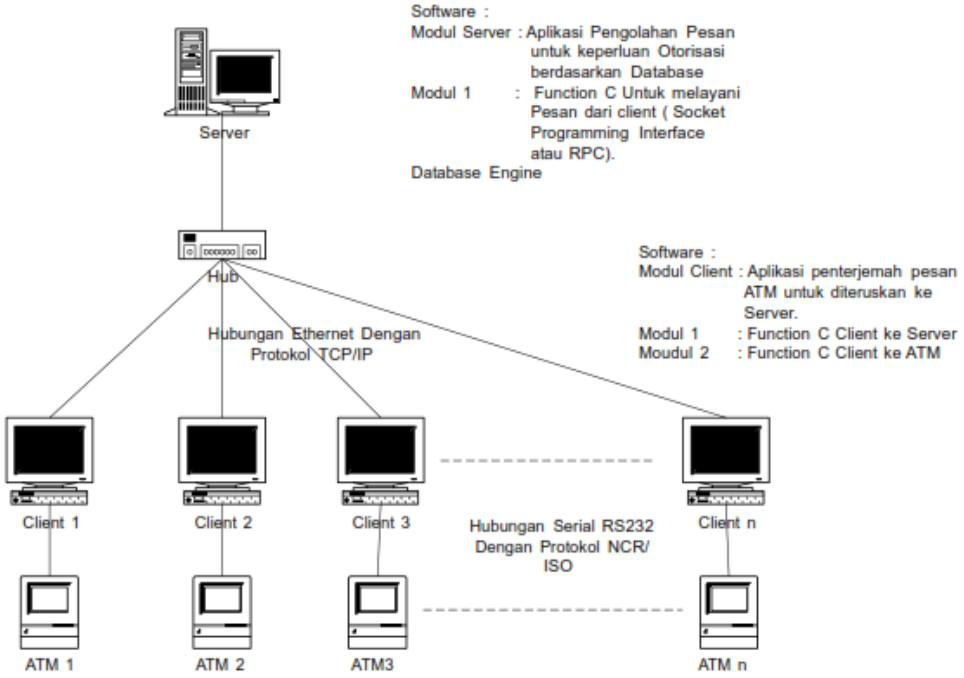
c. Rancangan *database*

d. Program aplikasi:

- Modul pada *Server*
- Modul pada *Client*

e. Analisis Kinerja: Yang berkaitan dengan response time dari sistem

Model yang dikembangkan adalah beberapa mesin ATM masing-masing dikendalikan oleh satu komputer PC yang seluruhnya terhubung dengan komputer pusat (PC) pada jaringan komputer dengan protokol TCP/IP seperti dapat dilihat pada Gambar 1.




Gambar 1. Model Jaringan Komputer Pengendali Multi ATM dengan Protokol TCP/IP

PC pengendali ATM disebut *client* dan berhubungan dengan ATM berdasarkan komunikasi serial RS232 dengan protokol NCR/ISO. Perangkat lunak yang diperlukan pada komputer *client* adalah modul *client*. Program ini akan menerima pesan dari ATM kemudian menerjemahkan pesan tersebut untuk diotorisasi ke komputer pusat. Program ini dapat dikembangkan dengan bahasa pemrogramman tingkat tinggi seperti 4GL, Delphi, dan VB. Untuk transfer data ke ATM diperlukan *function* C berdasarkan protokol NCR/ISO (Modul 1 *Client*) dan transfer data ke *server* diperlukan *function* C untuk meng-akses *server* berdasarkan IP tertentu demikian sebaliknya (Modul 2 *Client*).

Pada komputer *server*, disamping *database engine* juga diperlukan program utama yang akan menerima pesan dari *client* untuk diterjemahkan dan kemudian diotorisasi berdasarkan data yang ada pada *database*. Pada komputer ini perlu dibuatkan suatu program yang akan melayani seluruh *client* menggunakan bahasa C dan nantinya akan berada *resident* pada komputer (*daemon*) atau ditambatkan pada program utama. Yang perlu

diperhatikan disini adalah waktu tanggap dan prioritas pelayanan dari sistem jika permintaan pelayanan *client* terlalu banyak dan waktu respon jaringan/sistem.

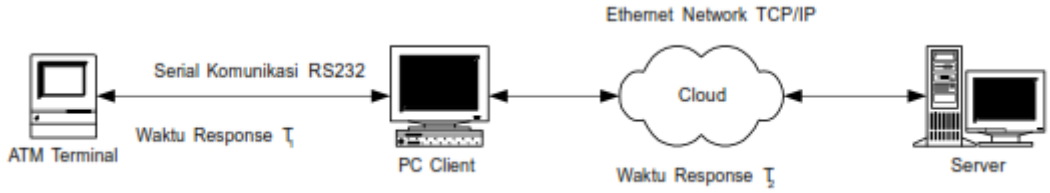
Sistem operasi yang digunakan akan tergantung pada *database engine* dan pengembang program aplikasi (modul *client/server*) yang digunakan. Dapat saja sistem operasi *client* dan *server* berbeda namun perlu dicari sistem yang stabil dan handal. Diharapkan pada *server* menggunakan sistem operasi Unix mengingat teknik pemrogramman yang akan digunakan adalah *Socket Programming Interface* atau *Remote Procedure Call*.

Waktu yang diperlukan untuk transfer data antara ATM terminal dengan *server* diasumsikan:

1. Waktu transfer data antara ATM dan komputer *client* adalah T1.
2. Waktu modul *client* untuk memproses data kemudian dikirimkan ke *server* adalah TMC.
3. Waktu transfer data antara komputer *client* ke *server* melalui *cloud* Jaringan adalah T2.
4. Waktu modul *server* untuk memproses data dan memberikan jawaban adalah TMS.

Jadi, total waktu respon adalah :

$$T_{respon} = 2T_1 + 2T_2 + T_{MC} + T_{MS} \text{ detik}$$




Gambar 2. Waktu *Response* Transfer Data antara ATM Terminal dan *Server*

Beberapa algoritma penanganan toleransi kesalahan sistem (SFT, *system fault tolerance*) berbentuk subprogram atau pemrogramman agen (*agent programming*) harus dikembangankan dan ditanamkan pada aplikasi utama:

1. Pembuatan subprogram (*agent programming*) penanganan kesalahan sistem.
2. Pembuatan program *interface* antara ATM dan PC melalui serial *port*.
3. Pembuatan program pengiriman paket data antar PC dengan protokol TCP/IP.
4. Pembuatan program pengendali beberapa ATM pada jaringan komputer berbasiskan TCP/IP.
5. Analisis kinerja sistem dan jaringan komputer terbatas.

## KESIMPULAN DAN SARAN

Rancangan mekanisme untuk mendapatkan kecepatan proses (*response time*) dari sistem untuk menangani beberapa titik yang

dapat diterima dan metode antisipasi kesalahan yang dapat ditolerir (*fault torance).*

**DAFTAR PUSTAKA**


[1] A. Sari dan M. Akkaya, "Fault tolerance mechanisms in distributed systems," *International Journal Communications, Network and System Sciences (IJCNS)*, vol. 8, hal. 471 – 482, 2015.

[2] J. Yin, J. P. Martin, A. Venkataramani, L. Alvisi, dan M. Dahlin, "Separating agreement form execution for byzantine fault tolerant services," In Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003, hal. 253 – 267.

[3] M. Aliouat dan Z. Aliouat, "Recovery in distributed systems from transient and permanent faults," *Journal of Computer Science*, vol. 3, no. 8, hal. 617 – 623, 2007.

[4] M. Castro and B. Liskov, "Byzantine fault tolerance can be fast," In Proceedings of International Conference on Dependable Systems and Networks, 2001, hal. 513 – 518.

[5] J. Yin, J. P. Martin, A. Venkataramani, L. Alvisi, dan M. Dahlin, "Byzantine fault tolerant confidentiality," In Proceedings of the International Workshop on Future Directions in Distributed Computing, hal. 12 – 15, 2002.

[6] G. C. Carpenter, "Byazantine fault-tolerant HTTP services using FARGOS/ VISTA," Fargos Development, New York, 2001.

[7] A. Postma, Th. Krol, dan E. Molenkamp, "Optimized authenticated self-synchronizing Byzantine agreement protocols," In Proceedings Pacific Rim International Symposium on Fault-Tolerant Systems, hal. 122 – 129, 1997.

[8] T. M. Kusuma, "Program aplikasi ATM dengan piranti lunak NCR Direct Connect (NDC) TM Native Band 1 yang berbasis personal computer," Skripsi, STMIK Gunadarma, Depok, 1994.

[9] T. N. Rachmat, " Program Komunikasi ATM Berperangkat Lunak NDC yang berbasis IBM PC dengan Protocol NCR/ISO," Skripsi, STMIK Gunadarma, Depok, 1994.