Penerapan dan Pengembangan Metode Analisis Keamanan Perangkat Lunak Lintas Udara Berdasarkan ARP4761

Rachmat Iskandar [1], Dahlan [2]

[1,2] *Program Studi Teknik Informatika- Politeknik TEDC Bandung*

Jl. Politeknik-Pesantren KM2 Cibabat Cimahi Utara – Cimahi Jawa Barat - Indonesia

rachmat@poltektedc.ac.id, dahlan@poltektedc.ac.id

***Abstrak*—Sebagian besar fungsi pesawat sipil dicapai melalui perangkat lunak saat ini. Oleh karena itu, keamanan perangkat lunak udara telah menjadi bagian penting dari analisis keselamatan. Namun, untuk mencapai fungsi yang kompleks, perangkat lunak onboard menjadi sangat rumit, dan biasanya sulit untuk menjamin keamanannya. Pedoman dan Metode untuk Melakukan Proses Penilaian Keselamatan pada Sistem dan Peralatan Lintas Udara Sipil adalah penilaian keselamatan untuk pesawat sipil tersebut, namun metode yang digunakan belum bisa memenuhi regulasi keamanan tersebut. Pada tahun 2012 SAE ARP 4671 mengembangkan suatu Pedoman dan Metode Pelaksanaan Proses Penilaian Keselamatan pada Sistem dan Peralatan Lintas Udara Sipil. ARP4761 percaya bahwa pengembangan perangkat lunak dapat memenuhi keselamatan selama proses pengembangan perangkat lunak sesuai dengan pengembangan perangkat lunak. Standar dengan memasukkan perangkat lunak ke dalam proses analisis keselamatan. Untuk meningkatkan keamanan perangkat lunak udara, penulis mencoba memaparkan metode analisis yang digunakan untuk menerapakan SAE ARP4671. Artikel ini mengusulkan metode analisis keamanan berdasarkan proses ARP4761 yang disesuaikan untuk beradaptasi dengan perangkat lunak. Dalam persyaratan dan desain perangkat lunak, penulis memaparkan metode untuk menganalisis keamanan perangkat lunak untuk keamanan penerbangan sipil dengan menggunakan *Functional Hazard Assessment (FHA)* yang berfungsi untuk menganalisis bahaya perangkat lunak dan menentukan tingkat bahaya, kemudian menggunakan *Fault Tree Analysis (FTA)* untuk membangun pohon kesalahan perangkat lunak, dan melaksanakan *Common Cause Analysis (CCA)* analisis berdasarkan pohon kesalahan, lalu kemudian setelah pengembangan perangkat lunak selesai maka menggunakan metode menggunakan *Failure Modes and Effects Analysis (FMEA)* dan *Failure Modes and Effects Summary (FMES)* untuk meringkas kegagalan perangkat lunak dan mengembalikan kegagalan ini ke pohon kesalahan untuk menentukan apakah perangkat lunak dapat memenuhi yang ditentukan persyaratan. Kesimpulan hasilnya menunjukkan bahwa ada cacat pada perangkat lunak anti-icing, dan cacat tersebut akan menyebabkan tingkat kegagalan seluruh sistem kontrol anti-icing lebih tinggi dari persyaratan, jadi harus dimodifikasi.perangkat lunak anti-icing yang digunakan sebagai contoh untuk menjelaskan metode keefektifan dan proses dapat sesuai dengan ketentuan regulasi SAE ARP4671.**

***Kata Kunci*—*FHA*, *FTA*, *CCA*, *FMEA*, *FMES*, ARP4761.**

I. PENDAHULUAN

Sebagian besar fungsi pesawat sipil dicapai melalui perangkat lunak saat ini (M. Takahashi, R. Kosaka, R. Nanba, Y. Anang and Y. Watanabe,2016). Oleh karena itu, keamanan perangkat lunak udara telah menjadi bagian penting dari analisis keselamatan. Namun, untuk mencapai fungsi yang kompleks, perangkat lunak onboard menjadi sangat rumit, dan biasanya sulit untuk menjamin keamanannya. Panduan dan Metode untuk Melakukan Proses Penilaian Keselamatan pada Sistem dan Peralatan Lintas Udara sipil adalah pedoman penilaian keselamatan yang penting untuk pesawat sipil, tetapi diyakini bahwa pengembangan perangkat lunak dapat memenuhi keselamatan selama proses pengembangan perangkat lunak sesuai dengan standar pengembangan perangkat lunak (DO-178C) (RTCA/DO-178C,2019) dengan memasukkan perangkat lunak ke dalam proses analisis. Masalah keselamatan yang serius dapat terjadi jika seorang insinyur hanya menganalisis keselamatan pesawat berdasarkan ARP4761 dan mengabaikan dampak perangkat lunak. Misalnya, salah satu penyebab kecelakaan Boeing 737MAX-8 adalah logika perangkat lunak yang salah dalam sistem peningkatan daya dorong, (Aircraft Accident Investigation Report of PT. Lion Mentari Airlines Boeing737-8,2020).

Untuk mengatasi masalah keamanan perangkat lunak, peneliti telah mengusulkan banyak metode analisis, seperti perangkat lunak *Fault Tree Analysis (FTA)* (M. Takahashi, R. Kosaka, R. Nanba, Y. Anang and Y. Watanabe,2016), perangkat lunak *Failure Modes and Effects Analysis (FMEA)(*Andrea Bondavalli, Francesco Brancati,2017), *Systems Theory Process Analysis (STPA*) (Asim Abdulkhaleq, Stefan Wagner,A, 2016), metode formal, dan metode terintegrasi antara metode ini, seperti metode terpadu STPA dan FMEA (Sulaman, S., Beer, A., Felderer, M. *et al.,2019)*. Namun, metode ini tidak digunakan dalam ARP4761, sehingga metode ini sulit untuk digabungkan ke dalam hasil analisis keamanan yang ada, yang pada akhirnya menyebabkan hasil analisis keamanan perangkat lunak tidak mencukupi. Untuk mengatasi masalah ini, artikel ini mengusulkan metode analisis keamanan berdasarkan proses ARP4761 yang disesuaikan dengan perangkat lunak, (Yuxin

Ding, Deming, Zhong, 2019). Dengan cara ini maka metode analisis keselamatan yang diusulkan dapat dihubungkan dengan ARP4761 sehingga proses dan hasil dari metode analisis keamanan yang berbeda akan kompatibel dengan sistem analisis keselamatan pesawat terbang sipil yang ada, (N. Snooke, Model-based Failure Modes and Eff, 2020).

II. METODE PENELITIAN

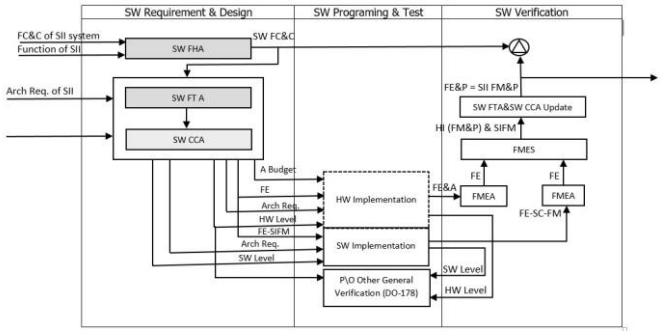
Berdasarkan proses pengembangan perangkat lunak, penulis membagi proses analisis keamanan perangkat lunak menjadi tiga bagian: fase persyaratan dan desain, fase implementasi, dan fase verifikasi. Oleh karena itu, penulis menyesuaikan proses ARP4761 dan membaginya menjadi tiga tahap sesuai dengan karakteristik perangkat lunak, seperti yang ditunjukkan pada Gambar 1.

1. Fase Persyaratan dan Desain Perangkat Lunak

Tujuan analisis keselamatan dalam persyaratan dan desain perangkat lunak adalah untuk memeriksa struktur perangkat lunak, menentukan fungsinya dan atribut bahaya yang sesuai, dan menilai dampak bahayanya pada sistem. Bagian ini terdiri dari tiga langkah: perangkat lunak FHA, perangkat lunak CCA, dan perangkat lunak FTA. Diantaranya, perangkat lunak FHA digunakan dalam persyaratan, sedangkan CCA dan FTA digunakan dalam desain. FHA perangkat lunak terutama digunakan untuk mengidentifikasi status kegagalan item intensif perangkat lunak, menentukan tingkat keparahan, dan mengklasifikasikannya berdasarkan dampak dari status kegagalan. Menurut klasifikasi keadaan kegagalan, persyaratan keselamatan yang sesuai dari keadaan kegagalan dapat ditentukan. Karena bahaya perangkat lunak ditentukan dari sistem maka FHA akan lebih intensif penggunaan perangkat lunak. FTA perangkat lunak adalah metode analisis top-down yang digunakan untuk menentukan penyebab kegagalan fungsi perangkat lunak. Dibutuhkan status kegagalan penting dalam FHA sebagai peristiwa teratas, yaitu dan peristiwa terbawah dari FTA adalah kegagalan komponen perangkat keras dan perangkat lunak Keluaran dari perangkat lunak FTA adalah bahaya dalam perangkat lunak. Software CCA digunakan untuk melengkapi dan meningkatkan software FTA dan hasilnya. Premis analisis FTA adalah sistemnya ketat, sehingga diperlukan CCA untuk menjamin independensi analisis FTA. CCA perangkat lunak yang didasarkan pada input kebutuhan perangkat lunak, desain, dan pohon kegagalan perangkat lunak, digunakan untuk memverifikasi independensi kejadian "dan gerbang" dalam FTA perangkat lunak, menganalisis secara kualitatif potensi penyebab umum status bahaya item intensif perangkat lunak , dan keluarkan perangkat lunak persyaratan dan bahaya desain yang terkait dengan penyebab umum, untuk memastikan bahwa bahaya penyebab umum terkait perangkat lunak dihilangkan atau diperbaiki atau dikendalikan.

2. Fase Implementasi

Pada bagian implementasi, Penulis mengimplementasikan perangkat lunak yang sesuai sesuai dengan persyaratan keamanan perangkat lunak yang diperoleh dari identifikasi kebutuhan perangkat lunak dan analisis bahaya. Implementasinya tidak memerlukan analisis keamanan, ini hanya proses pengembangan perangkat lunak. Perlu dicatat bahwa beberapa persyaratan keamanan perangkat lunak pada akhirnya akan diterapkan di perangkat keras (seperti waktu respons perangkat keras). Oleh karena itu, beberapa implementasi terkait perangkat keras juga akan muncul.



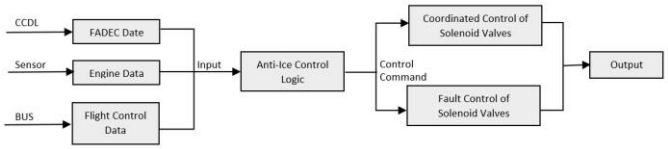
Gbr.1. Metode Analisis Keamanan Perangkat Lunak Berdasarkan ARP4761 (Sumber: ARP4761 SAE International)

3. Fase Verifikasi Perangkat Lunak

Tujuan analisis bahaya pada bagian verifikasi perangkat lunak adalah untuk menganalisis dan memverifikasi apakah persyaratan keselamatan yang ditentukan pada bagian persyaratan dan desain telah terpenuhi sesuai dengan hasil desain dan implementasi item perangkat lunak dan perangkat keras. Bagian ini terdiri dari tiga langkah: perangkat lunak FMEA dan FMES, pembaruan pohon kesalahan perangkat lunak, dan pembaruan analisis penyebab umum perangkat lunak. Karena beberapa kegagalan perangkat keras akan mempengaruhi kegagalan perangkat lunak, kegagalan ini perlu dipertimbangkan oleh perangkat lunak. Oleh karena itu, perangkat lunak FMES mungkin perlu menyertakan beberapa hasil perangkat keras FMEA, tetapi FMEA perangkat keras tidak termasuk dalam proses. Perangkat lunak FMEA mengacu pada daftar periksa mode kegagalan kode perangkat lunak, menganalisis mode kegagalan kode perangkat lunak yang dirancang dan diimplementasikan, (A. Abdulkhaleq, S. Wagner,2015). Menurut pekerjaan sebenarnya, mode kegagalan FMEA dapat berupa level komponen perangkat lunak atau level unit perangkat lunak. FMES adalah ringkasan dari FMEA, tujuannya adalah untuk menentukan dampak kegagalan kode pada perangkat lunak. Pembaruan pohon kesalahan perangkat lunak adalah proses pembaruan yang didasarkan pada perangkat lunak yang diimplementasikan, struktur dan data pohon kesalahan yang dibangun pada tahap identifikasi persyaratan perangkat lunak diperbarui melalui hasil FMES perangkat lunak, untuk memverifikasi apakah keamanan perangkat lunak yang diimplementasikan memenuhi standar awal. persyaratan Pembaruan CCA perangkat lunak adalah proses pembaruan berdasarkan pohon kesalahan perangkat lunak yang diperbarui. Tujuan dari langkah tersebut adalah untuk memverifikasi apakah perangkat lunak yang dirancang masih memenuhi persyaratan independensi. Jika analisis pohon kesalahan perangkat lunak sebelumnya dan persyaratan independensi tidak diperbarui, langkah ini tidak diperlukan.

4. HASIL PEMBAHASAN

Metode ini diterapkan pada software anti-icing pesawat terbang. Perangkat lunak anti-icing adalah bagian dari sistem anti-icing, yang mencakup anti-icing manual, anti-icing otomatis, alarm kegagalan anti-icing, dan pemeriksaan mandiri anti-icing, prosesnya ditunjukkan dalam gambar 2.



Gbr.2. Proses Anti-Ice

1. Persyaratan dan Desain Perangkat Lunak Anti-es

A. FHA dari Perangkat Lunak

Tujuan FHA adalah untuk mengidentifikasi fungsi sistem kontrol anti-icing, mode bahaya fungsional, dan atribut bahaya terkait. Menurut pengaruh keadaan kegagalan, efeknya ditentukan dan diklasifikasikan, dan hasil FHA diberikan, yang mencakup tiga bagian: analisis status kegagalan fungsi, analisis tahapan kerja, dan klasifikasi bahaya, kemudian menentukan persyaratan tingkat kegagalan dari kondisi kegagalan berdasarkan hasil FHA.

Pertama, status kegagalan perangkat lunak anti-icing dianalisis, termasuk kegagalan sistem kontrol anti-icing, mematikan fungsi anti-ice, dan pembukaan fungsi anti-es yang salah.

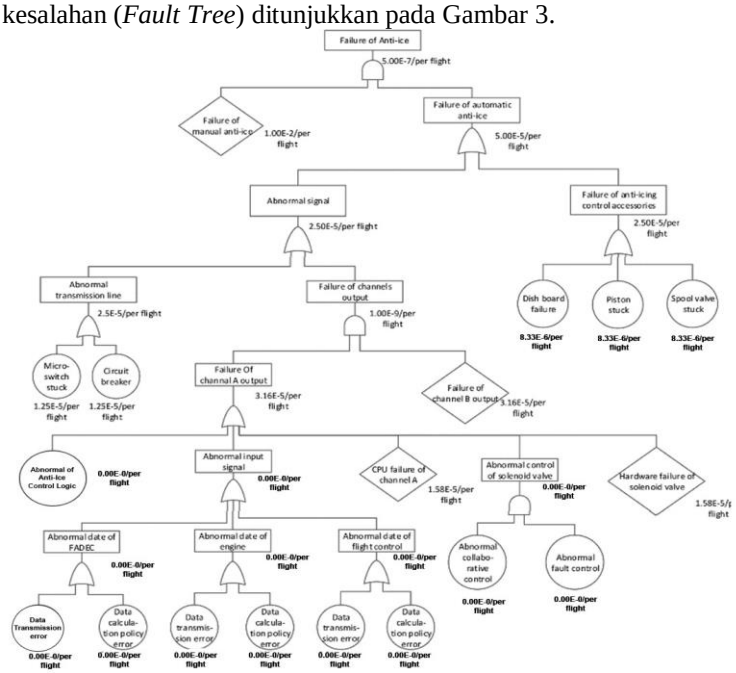
Kedua, tahap kerja yang sesuai dari kondisi gangguan ditentukan, termasuk F0: operasi darat; F1: meluncur dan lepas landas; F2: memanjat; F3: penerbangan leveling; F4: mendarat; F5: mendarat dan meluncur. Akhirnya, bahaya kegagalan perangkat lunak anti-icing di setiap tahap kerja dianalisis dengan mencocokkan status kegagalan fungsi dan tahap kerja, dan hasil analisis FHA diberikan, seperti yang ditunjukkan pada Tabel 1. Menurut kondisi kegagalan dan klasifikasi keamanan yang sesuai dari kontrol anti-icing otomatis, persyaratan keselamatan dapat diturunkan (hanya dengan mempertimbangkan kondisi kegagalan dengan klasifikasi seperti di atas III) sebagai tingkat kegagalan anti-icing (F1, F2, F3 , F5, F6) harus kurang dari 5e-7 per penerbangan; (dalam persyaratan keselamatan yang diberikan di bawah ini, waktu rata-rata setiap misi penerbangan adalah 5 jam).

TABEL I
HASIL FHA PERANGKAT LUNAK ANTI-ICE

Fungsi	Kondisi Kegagalan	Fase	Pengaruh Kondisi Kegagalan pada Pesawat/Awak Pesawat	Klasifikasi	Verifikasi
Perangkat Lunak Anti-ice	Kegagalan fungsi kontrol anti-ice	F1,F2, F3,F4, F5	Cedera personel, gangguan misi penerbangan, kerusakan mesin serius	II	FTA
		F0	Tidak berbahaya bagi personel, tidak berpengaruh pada misi, tidak ada kerusakan pada mesin	IV	
	Tidak bisa turn off fungsi anti-ice	F0,F1, F2, F3,F4, F5	Tidak berbahaya bagi personel, tidak berpengaruh pada misi, tidak ada kerusakan pada mesin	IV	
	Kesalahan opening pada fungsi anti-ice	F0,F1, F2, F3,F4, F5	Tidak berbahaya bagi personel, tidak berpengaruh pada misi, tidak ada kerusakan pada mesin	IV	

B. FTA dari Perangkat Lunak

Analisis FTA perangkat lunak adalah dekomposisi persyaratan keselamatan ke bawah berdasarkan Kondisi Kegagalan yang diperoleh dari FHA. Persyaratan keamanan perangkat lunak diturunkan ke desain. Level kejadian yang disepakati di bagian bawah Fault Tree adalah software item design (SI) dan bagian dari hardware item design (HI). Pohon kesalahan (*Fault Tree*) ditunjukkan pada Gambar 3.



Gbr.3. Pohon Kesalahan Perangkat Lunak Anti-Ice

C. CCA dari Perangkat Lunak

Fungsi perangkat lunak kontrol anti-icing harus independen untuk memastikan tidak ada kesalahan mode umum karena kesalahan pengembangan. Dengan menganalisis "dan peristiwa gerbang" dari pohon kesalahan pada Gambar. 3, dapat disimpulkan bahwa peristiwa independen perangkat lunak CCA meliputi:

1) Saluran A dan B perangkat lunak anti-icing harus independen satu sama lain untuk memastikan bahwa kegagalan fungsi anti-icing dalam satu saluran tidak akan menyebabkan kegagalan anti-Ice;
2) Kesalahan dan kontrol kolaboratif katup solenoid di setiap saluran perangkat lunak anti-icing harus independen untuk memastikan kegagalan tunggal tidak akan menyebabkan kegagalan fungsi anti-icing mesin. Hasil CCA ditunjukkan pada Tabel II.

TABEL III
HASIL CCA PERANGKAT LUNAK ANTI-ICE

Failure condition	Examples of common mode source	Examples of common mode failures
Both channel A and B failures software fails at the same time	Share the same CPU or memory address	CPU failures or memory leak
Fault control and collaborative control fail at the same time	Same input variables	an error in the input variables

2. Implementasi Perangkat Lunak Anti-Ice

Implementasi perangkat lunak merupakan bagian dari pengembangan perangkat lunak tetapi bukan bagian dari metode analisis keamanan. Namun, hasil analisis perangkat lunak FHA, FTA, dan CCA harus disertakan dalam kode untuk memastikan persyaratan keamanan perangkat lunak anti-Ice.

3. Verifikasi Perangkat Lunak Anti-Ice

A. FMEA dan FMES dari perangkat lunak

Menurut hasil pengujian perangkat lunak, penulis melakukan perangkat lunak FMEA. Mode kegagalan menggunakan pustaka mode kegagalan perangkat lunak yang dirangkum dari berbagai standar dan pengalaman, kemudian penulisi merangkum hasil FEMA melalui perangkat lunak FMES. Untuk beradaptasi dengan ARP4761, tingkat kegagalan perangkat lunak adalah 0 atau 1. Jika item perangkat lunak lulus uji, tingkat kegagalannya adalah 0, jika tidak maka 1. Karena ada banyak konten analisis FMEA, penulis tidak bermaksud hasilnya pada makalah ini. Hanya hasil di mana tingkat kegagalan adalah 1 dari perangkat lunak FMES diberikan, seperti yang ditunjukkan pada Tabel III.

TABEL IIIII
FEMS DARI PERANGKAT LUNAK ANTI-ICE

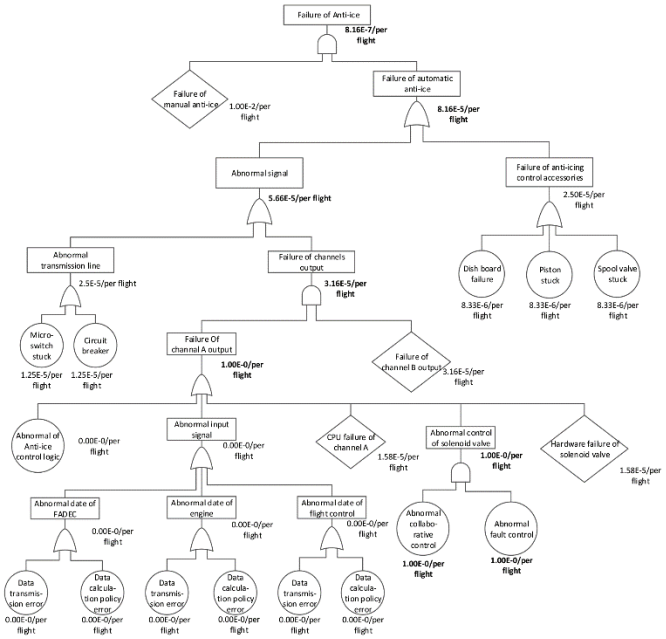
Kode Perangkat Lunak	Kode Kegagalan	Tingkat Kegagalan	Efek Kegagalan
Logika anti-ice	Anti-Ice otomatis tidak memiliki keluaran	1	Kegagalan anti-ice otomatis (karena desainnya akan tetap terbuka)
	Tidak dapat keluar dari logika control anti-ice	0	Tidak ada efek keamanan
Kontrol Kolaboratif	Masih dalam kondisi gagal saluran	1	Kegagalan anti-ice otomatis
Kontrol Kesalahan	Kontrol tidak normal	1	Kesalahan diagnosis

B. Pembaruan CCA perangkat lunak

mendapatkan kemungkinan hasil kegagalan mode umum dari perangkat lunak di set 3.2, Melalui hasil analisis FMEA, dapat diketahui bahwa masalah tersebut tidak ada pada perangkat lunak yang dirancang, sehingga analisis CCA perangkat lunak tidak perlu dilakukan.

C. Pembaruan FTA perangkat lunak

Menurut hasil analisis FMES, kami menemukan bahwa beberapa kejadian terbawah di pohon kesalahan perangkat lunak muncul di perangkat lunak. Oleh karena itu, kami perlu memperbarui pohon kesalahan perangkat lunak dan menghitung ulang laju kejadian teratas. Hasilnya ditunjukkan pada Gambar 4. Dengan membandingkan pohon kesalahan sebelum dan sesudah pembaruan, dapat dilihat bahwa tingkat sebenarnya dari "kegagalan fungsi kontrol anti-icing" adalah 8.16e-7 per penerbangan, yang lebih tinggi dari persyaratan keselamatan 5.0e-7 yang ditentukan oleh FHA. Tingkat kegagalan tidak memenuhi persyaratan keselamatan, oleh karena itu, perangkat lunak kontrol kolaboratif dan perangkat lunak kontrol kesalahan perlu dimodifikasi untuk memenuhi persyaratan tingkat kegagalan.



Gbr 4. Pohon Kesalahan Anti-Ice yang Diperbarui

5. KESIMPULAN DAN SARAN

Penelitian ini menyajikan metode analisis keamanan perangkat lunak berdasarkan ARP4761. Metode ini mempertahankan karakteristik dasar ARP4761 yang dapat dengan mudah mengintegrasikan hasil analisis ARP4761, menjaga konsistensi proses analisis keamanan, mengintegrasikan proses analisis keamanan perangkat lunak ke dalam proses analisis keseluruhan sistem, dan memungkinkan analis untuk melakukan analisis keamanan perangkat lunak dengan cepat.

1. Kesimpulan

hasilnya menunjukkan bahwa ada cacat pada perangkat lunak anti-icing, dan cacat tersebut akan menyebabkan

tingkat kegagalan seluruh sistem kontrol anti-icing lebih tinggi dari persyaratan, jadi harus dimodifikasi.

2. Saran

   Karena penulis menerapkan metode ini ke perangkat lunak anti-icing dan masih memiliki beberapa keterbatasan. Maka penulis memberikan saran berdasarkan 2 aspek, yaitu:

   a) Analisis perangkat lunak masih harus dilakukan menurut proses sistem, bukan berdasarkan profil pengoperasian perangkat lunak;
   b) Mode kegagalan perangkat lunak FMEA terkait erat dengan kualitas analisis. Berikut ini, rencana penelitian lebih lanjut kami mencakup dua poin. Salah satunya adalah
   c) Perlu mempertimbangkan skenario kegagalan perangkat lunak melalui profil operasi.

REFERENSI

[1] SAE International- U.S., ARP 4761. *Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment.*

[2] RTCA/DO-178C, *Software Considerations in Airborne Systems and Equipment Certification[S].RTCA*, 2019.

[3] Aircraft Accident Investigation Report of PT. Lion Mentari Airlines Boeing737-8(MAX) http://knkt.dephub.go.id/knkt/ntsc_home/ntsc.htm, Accessed08/21/2020

[4] Yuxin Ding, Deming, Zhong, at el. An Airborne Software FMEA Application Method Based on ARP4761. 2019 2nd *International Conference on Management Engineering, Software Engineering and Service Sciences*

[5] M. Takahashi, R. Nanba, and Y. Fukue, *A Proposal of Operational Risk Management Method Using FMEA for Drug Manufacturing Computerized System, Transaction on The Society of Instrument and Control Engineers*, Vol.45, No. 5. pp285-294, 2012 (in Japanese)..

[6] Asim Abdulkhaleq, Stefan Wagner,*A controlled experiment for the empirical evaluation of safety analysis techniques for safety-critical software, Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*, 2016.

[7] Andrea Bondavalli, Francesco Brancati, *Certifications of Critical Systems - The CECRIS Experience, Chapter 9, Composable Framework Support for Software-FMEA through Model Execution*., 2017.

[8] Sulaman, S., Beer, A., Felderer, M. et al. *Comparison of the FMEA and STPA safety analysis methods-a case study. Software Qua*l J 27, 349-387 (2019).

[9] A. Abdulkhaleq, S. Wagner, *Integrated safety analysis using systems-theoretic process analysis and software model checking, Computer Safety, Reliability, and Security Volume 9337 of the series Lecture Notes in Computer Science*, 2017, pp. 121-134.

[10] N. Snooke, Model-based Failure Modes and Eff ects *analysis of Software, Proc. of 15th International Workshop on the Principles of Diagnosis*,pp.221-226,2020