Acta Astronautica 152 (2018) 360–369

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Resilience evaluation of the environmental control and life support system of a spacecraft for deep space travel

José Alexandre Matelli[a,b,*], Kai Goebel[a,c]

[a] NASA Ames Research Center, Intelligent Systems Division, Discovery and Systems Health, Moffett Field, CA, USA
[b] São Paulo State University (UNESP), School of Engineering, Department of Energy, Guaratinguetá, SP, Brazil
[c] Luleå University of Technology, Division of Operation and Maintenance Engineering, Luleå, Sweden

ARTICLE INFO

Keywords:
Resilience
Resilient design
Life support
Conceptual design
Deep space

ABSTRACT

In deep space manned travels, the crew life will be totally dependent on the environment control and life support system of the spacecraft. A life-support system for manned missions is a set of technologies to regenerate the basic life-support elements, such as oxygen and water, which makes resilience a paramount feature of this system. The resilience of a complex engineered system is the ability of the system to withstand failures, continue operating and recover from those failures with minimum disruption. Resilient design is a new design framework on which the main goal is to quantify system resilience upfront in order to guide the design team during the conceptual design stage. In this article, we present a tool that combines a rule-based approach with a Monte Carlo-based approach to evaluate the resilience of a proposed environment control and life support system designed for deep space travel. Based on the results found, we explore a few design alternatives in order to increase system resilience.

1. Introduction

In deep space manned travels, the crew will be subject to long periods of time onboard a spacecraft. In this situation, the crew will experience strict limitations on communication between the crew in space and control centers on Earth [1] and even monotony [2]. Also, the crew life will be totally dependent on the Environment Control and Life Support System (ECLSS) of the spacecraft. A life-support system for manned missions is a set of technologies to regenerate the basic life-support elements, i.e., oxygen, potable water, and food during long-time spaceflights, such as in low-Earth orbit, on the Moon, on Mars, or beyond [3]. Life support technologies are also being studied for colonization of the Moon [4] and Mars [5,6].

Quite likely, the human exploration of Mars will be the first deep space manned mission. A Mars mission would take up to 1100 days, including the trip itself, descent to the surface, exploration of the surface, ascent form the surface, and return to Earth [7]. According to Stapleton et al. [8], “mitigating safety hazards ensures the ECLSS systems are safe during operation and do not create hazardous conditions for the crew. Mitigating functional hazards prevents a failure or failures from causing a loss of a critical life sustaining function, such as providing oxygen or drinking water to the crew. For deep space exploration, mitigating safety hazards should be similar to International Space Station (ISS). However, functional hazard mitigations may change significantly” [8], especially because re-supply visits from Earth are not possible in deep space travel. Thus, certain systems that are not considered critical life-sustaining on ISS will be considered critical life-sustaining in deep space exploration. For example, the ISS has an oxygen generator assembly (OGA) to provide oxygen to the crew. Since bottles of oxygen can be brought up to ISS in the event of OGA failure, the OGA is not considered to perform a life-sustaining function. Indeed, in 2011 the OGA of the ISS U.S. segment experienced problems for several months because the water was slightly too acidic, so the station crew used oxygen brought aboard by supply ships while awaiting delivery of OGA repair equipment [9]. In deep space travel, on the other hand, the OGA performs a critical life-sustaining function because any failure to generate oxygen onboard would compromise the oxygen supply for the crew and could ultimately threaten crew life.

There are other life-sustaining functions that the ECLSS must perform, such as provide drinkable water and remove carbon dioxide, which makes resilience a paramount feature of this system. Haimes [10] defines resilience as the ability of the system i) to withstand a major disruption within acceptable degradation parameters and ii) to recover within an acceptable time and composite costs and risks. The

[*] Corresponding author. São Paulo State University (UNESP), School of Engineering, Department of Energy, Av. Ariberto Pereira da Cunha, Guaratinguetá, SP, Brazil.

E-mail addresses: jose.a.matelli@unesp.br (J.A. Matelli), kai.goebel@nasa.gov (K. Goebel).

https://doi.org/10.1016/j.actaastro.2018.08.045
Received 21 June 2018; Received in revised form 21 August 2018; Accepted 27 August 2018
Available online 28 August 2018
0094-5765/

ECLSS can be considered a Complex Engineered System (CES), i.e., a system composed of densely interrelated subsystems in order to perform one or more high level functions. A subsystem is itself is a collection of interrelated components that has to perform a specific function. Design teams seek conceptual solutions for low life-cycle cost, which includes, among others, capital, operation and maintenance costs throughout the system life time. Capital cost is related to the costs of the individual system components, so the design team can optimize for low-cost configurations. Operation cost is related to the system performance, so designers can optimize for high efficiency solutions. However, when it comes to maintenance costs, the design team usually considers the costs of scheduled maintenance and ignores the costs related to unpredicted failures. This is no surprise, because during conceptual design detailed knowledge of system components and their performance criteria typically are not yet available [11], so in absence of that information, this aspect is largely ignored.

Design for resilience, or resilient design, is a new design framework where the CES is designed to stay maximally operational while considering the existence of failures. Recent efforts towards resilient design can be found in Refs. [11–15]. To evaluate the CES resilience computational aid is highly desirable to explore a multitude of possible outcomes. In a previous work, we proposed a novel resilient design framework specific for cogeneration plants [13]. Here, we generalize the framework developed in Ref. [13] and apply it to a proposed ECLSS design [8,16] for a deep space spacecraft. The objective is to present the Generalized Resilient Design Framework (GRDF) and evaluate the resilience of the ECLSS during the conceptual design phase. The GRDF is embedded in a computational tool based on declared rules and Monte Carlo simulations. The article is organized as follows: in section 2, we propose metrics to quantify CES resilience; in section 3, we present the computational tool and show how it computes the metrics presented in section 2; in section 4, we apply the tool to evaluate the resilience of the proposed ECLSS, explore some changes in the design (thereby affecting its resilience) and discuss the results; finally, in section 5 we present conclusions and final remarks.

2. Generalized resilient design framework

A resilient design framework for cogeneration plants was previously developed by the authors [13]. In order to evaluate the resilience of the ECLSS, that framework is generalized for any CES here. The Generalized Resilient Design Framework (GRDF) is based on simulation of failure propagation within the CES. The CES is designed for a determined useful life, represented by T_u in the simulation. For every time step of the simulation, a component or subsystem i is randomly picked as candidate to a random failure f_1. Component i has a known probability p_i to work properly (the higher the component quality, the higher p_i). The failure f_1 occurs with a random probability p_f and f_1 is injected in component i if $p_f > p_i$ at time step t_1; otherwise, component i does not fail. As shown in Fig. 1, there are four possibilities: i) f_1 does not affect the CES and it keeps fully working until time T_u; ii) f_1 does affect the CES and it keeps partially working until time T_u; iii) f_1 does affect the CES and it completely fails before T_u; iv) a new random failure f_2 occurs in another randomly picked component after f_1. After f_2, the possibilities are the same as before: CES fully or partially working until time T_u, CES completely fails before T_u or a new failure f_3 occurs and so on.

The CES is simulated N times, each simulation set for T_u operating hours. In order to assure that only the CES configuration affects its resilience, the GRDF considers an ideal failure propagation mechanism. The following assumptions are made:

a) All components or subsystems have the same probability p_i to work properly;
b) p_i does not change and is time-independent;
c) A failure in a component is instantaneously propagated to any other component connected to the failed component, regardless the nature of the connection;
d) A failure in a component propagates to any other component connected to the failed component with a constant, time-independent probability equal to 1;
e) No partial failure of any component or subsystem is admitted;

Regarding the first assumption, we acknowledge that failure probability is not the same for all components. However, access to accurate failure rates or failure probabilities depends on large amount of historic operational data for all components. In the case of brand new systems, such as the ECLSS, these data do not even exist. Since such information is often not available during early design phases, a design framework to compare the resilience of different designs should depend solely on the CES configuration. In this case, specific engineering information of the components, such as failure probability, would not affect the resilience comparison. As a matter of fact, it is shown in Ref. [13], without formal proof, that the resilience comparison of different systems is not affected whether considering equal p_i or not, provided that p_i is large and, in the case of different p_i, the difference between them is within an order of magnitude. The set of failure probabilities, being p_i all equal or not, can be interpreted as establishing a reference frame from which the resilience of different designs can be evaluated. It follows that a fair resilience comparison is possible when the different designs presents the same set of probabilities p_i. Although the GRDF can be used irrespective of whether the set of failure probabilities has all p_i equal or not, we consider a set with equal probabilities for the purposes of the present work, as initially hypothesized. From this framework, six metrics for resilience are proposed:

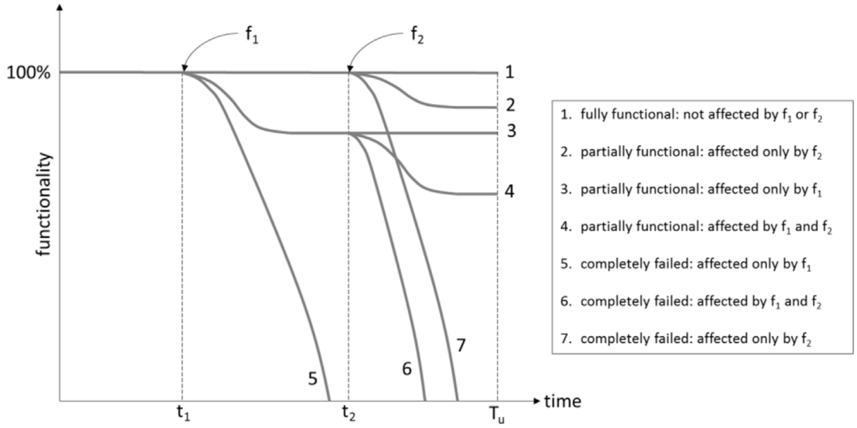


Fig. 1. Failures affecting functionality of complex engineered systems.

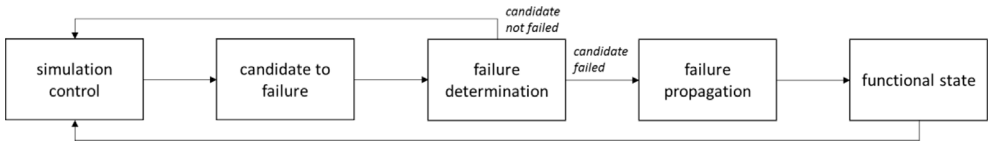


Fig. 2. Algorithm organization in block of rules.

i. Probability of resilient operation: fraction of simulations that result in resilient operation during a period of time T_u for a given p_i and infinite number of simulations (Eq. (1)). The higher $p_r(T_u, p_i)$, the higher the resilience.

$$p_r(T_u, p_i) = \lim_{N\to\infty}(N_r/N) \tag{1}$$

ii. Probability of resilient operation: fraction of simulations that result in failed operation in a period of time close to $\bar{f} < T_u$ for a given p_i and infinite number of simulations (Eq. (2)). The higher $p_f(T_u, p_i)$, the lower the resilience.

$$p_f(T_u, p_i) = \lim_{N\to\infty}(N_f/N) \tag{2}$$

ii. Resilient operating time: average of the resilient time r, weighted by $p_r(T_u, p_i)$, for all simulations n on which $t_n = T_u$ and $0 < r_n \le t_n$ for a given p_i (Eq. (3)). A CES presenting a high value $\bar{r}(T_u, p_i)$ is a CES with high resilience.

$$\bar{r}(T_u, p_i) = \frac{p_r(T_u, p_i)}{N_r}\sum_{n=1}^{N_r} r_n\{n \mid t_n = T_u,\ 0 < r_n \le t_n\} \tag{3}$$

iv. Time until failure: average of the total operating time t_n for all simulations n on which $t_n < T_u$ and $r_n \le t_n$ for a given p_i (Eq. (4)). The higher $\bar{f}(T_u, p_i)$, the higher the resilience.

$$\bar{f}(T_u, p_i) = \frac{1}{N_f}\sum_{n=1}^{N_f} t_n\{n \mid t_n < T_u,\ r_n \le t_n\} \tag{4}$$

v. Average operating time: the weighted average between the average operating time of all simulations n on which $t_n < T_u$ and the average operating time of all simulations n on which $t_n = T_u$. The weights are $p_f(T_u, p_i)$ and its complement, respectively (Eq. (5)). The higher $p_f(T_u, p_i)$, the higher the resilience.

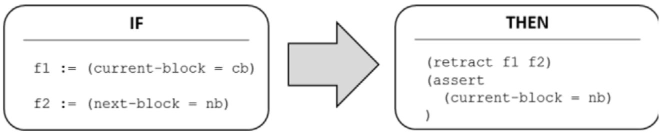
$$\bar{t}(T_u, p_i) = p_f(T_u, p_i)\bar{f} + [1 - p_f(T_u, p_i)]T_u \tag{5}$$

vi. Normalized resilience index: the ratio between the average operating time and the determined useful life T_u (Eq. (6)). The higher $\rho(T_u, p_i)$, the higher the resilience. The limit case $p_f = 0$ in Eq. (6) results in a upper limit $\rho(T_u, p_i) = 1$, regardless of $\bar{f}$. It is not possible to anticipate the value of $\bar{f}$ when $p_f = 1$ because the simulations are stochastic. Assuming that $\bar{f}$ remains the same, $\rho(T_u, p_i) \cong \bar{f}/T_u$ is a reference lower limit obtained from Eq. (6) for $p_f = 1$. For all concepts with the same T_u and p_i, the most resilient design possible would present $\rho(T_u, p_i) = 1$ and the least resilient one would have $\rho(T_u, p_i) \cong \bar{f}/T$.

$$\rho(\mathrm{T_u}, \mathrm{p_i}) = \bar{t}(T_u, p_i)/T_u \tag{6}$$

3. Computational implementation of the generalized resilient design framework

A computational tool encompasses the generalized resilient design framework described in section 2. As common in rule-based systems (e.g. Refs. [17,18]), the tool is programmed in a declarative fashion. However, the simulation requires some control flow. The algorithm is organized in blocks of rules, as shown in Fig. 2 with typical control rules that manipulate control facts (see example in Fig. 3).



```
f1 := (current-block = cb)

f2 := (next-block = nb)
```

```
(retract f1 f2)
(assert
  (current-block = nb)
)
```

Fig. 3. Control rule.

The ontology is quite simple, composed of two classes. One, named [system], represents the CES; the other, named [component], represents the components that together compose the CES. The relations between those classes are represented with "[component] is part of [system]". An object of the class [system] is composed of several interrelated objects of the class [component]. The interrelation between the objects of [component] is represented by the attributes [affectedBy] and [redundancies]: the attribute [affectedBy] of a given object lists all other objects of [component] whose failure causes the given object to fail as well; the attribute [redundancies] lists all other objects of [component] that performs the same function of the given object. The attributes of the classes [system] and [component] are shown in Table 1. Class [component] also presents a failure propagation procedure, which is detailed further down.

The CES configuration is represented as an object of class [system] and respective objects of class [component]. The *simulation control* block starts the simulations prompting the user for information on the CES configuration, the operating time T_u and the number of simulations N. As shown in Fig. 4, this block also computes the average resilience metrics as the simulations run and it controls the information flow among the blocks through the control rule depicted in Fig. 3. When simulation n starts, a non-failed component i is randomly chosen as a failure candidate at time t (component i has a known probability p_i to work correctly) in the *candidate to failure* block. In the *failure determination* block, a failure probability $p_b(t)$ for time t is randomly assigned. If $p_b(t) \le p_i$, component i does not fail and the control rule takes the execution back to the *simulation control* block, time step t (and eventually r) is updated and a new candidate to failure is randomly chosen. If $p_b(t) > p_i$, the control rule takes the execution to the *failure propagation* block.

The *failure propagation* block determines whether the failure in component i propagates to component j. If it does, there is a check whether failure in j propagates to component k and so on, until failure propagation eventually stops. Failure propagation is evaluated based on a rule whose premises are that one of two objects [component] has failed. In order to represent the knowledge that a failure does not propagate to a physical redundancy, the object's attribute [componentType] must differ. The rule for failure propagation is depicted in Fig. 5. Fig. 6 shows the associated procedure [propagates] for the non-failed component that evaluates the failure propagation.

Successive execution of the failure propagation rule results in a set $c(t)$ composed of all objects [component] whose [failed] attribute is equal to *yes* at time t. Since no repair action is considered, $c(t)$ can only increase in time. The *functional state* block verifies whether $c(t)$ causes CES to fail. This is done by a rule (Fig. 7) that takes the number of objects [component] that perform system functions (described in the

[1] For the sake of simplicity, only the resilient operating time $\bar{r}(T_u, p_i)$(computed as rot in Fig. 4) is depicted. All other metrics presented in section 2 are computed in a similar fashion.

Table 1
Attributes of classes [system] and [component].

Class [system]

Attributes	Type	Allowed values	Note
systemID	Integer	any positive	System identifier
whatIsThisSystem	Symbol	any	System description
numberOfComponents	Integer	any positive	Number of components of the system
howManyMeetFunction1	Integer	any positive	Number of components meeting function1
howManyMeetFunction2	Integer	any positive	Number of components meeting function2 (as many as required)
OverallOperationalState	Symbol	Normal, resilient, failed	System operational state
function1FunctionalState	Symbol	normal, resilient, failed	Function1 operational state
function2FunctionalState	Symbol	normal, resilient, failed	Function2 operational state (as many as required)

Class [component]

Attributes	Type	Allowed values	Note
componentID	Integer	any positive	Component identifier
componentType	Symbol	any	Component type
systemFunction	Symbol	none, function1, function2	Component perform one of the system function
failed	Symbol	yes, no	Component is failed
failureOrigin	Symbol	none, original, propagated	Origin of component failure
failureProbability	Float	any between 0 and 1	Probability of component failure
redundancies	Integer	any	List of components that perform the same function
affectedBy	Integer	any	List of components that affect the functionality

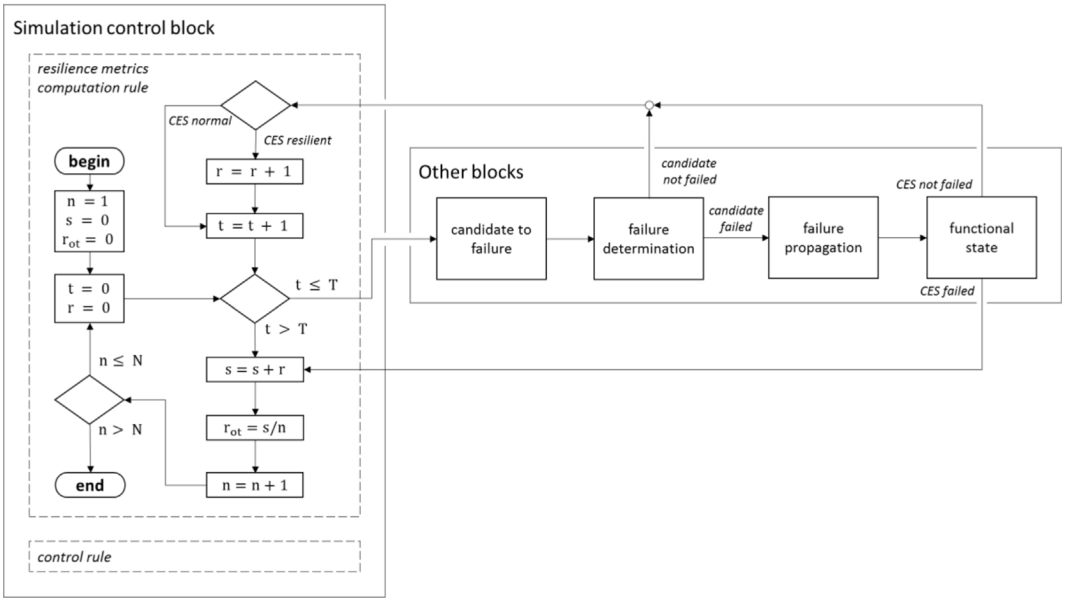


Fig. 4. Average resilience metrics[1] computation and control of the information flow.

attribute [system Funcion]) and compares them to the respective number of components of the object [system] that perform the same function (described in one of the attributes [howManyMeetFuncion1], [howManyMeetFuncion2], …).

It should be noted that at time t: i) the CES is completely failed only if all objects [component] that perform system functions are failed; ii) the CES is in normal operation only if all objects [components] are not failed; iii) the CES is in resilient operation otherwise. The information flow goes back to the *simulation control* block to update t and r (and all other resilient metrics). The simulation n ends with one out of three possible CES operating states:

Normal: no component is failed at time $t = T_u$ and resilient time results $r = 0$;

Failed: there are failed components and the CES is not capable to perform its functions at time $t < T_u$. Resilient time results $r \leq T_u$;

Resilient: there are failed components, but the CES is capable to perform within acceptable degradation parameters, as stated in the first part of Haimes' definition [10], at time $t = T_u$. Resilient time results $r \leq T_u$.

As simulation n ends, simulation number n, resilient operating time r and other resilient metrics are updated. Then, t and all resilient metrics are set to zero, so that a new simulation $n + 1$ starts. This loop ends when N simulations are performed and the resilient metrics are evaluated.

The GRDF tool can also be used to study failure propagation mechanisms. By choosing this option, no resilience metric is calculated; instead, the designer chooses a component to fail. The tool settings become $p_b(t) = 1$, $T = 1$ and $N = 1$, resulting in only one $c(t)$ corresponding to the propagation of the failure originated in the component chosen. An explanation of how this failure propagates throughout the CES is then presented, which is done by generating dynamic strings in the procedure [propagates], as shown in Fig. 6. The value of each string is shown in Table 2. Value of attribute [componentType] of the object's components involved are respectively assigned to variables i, j and k.

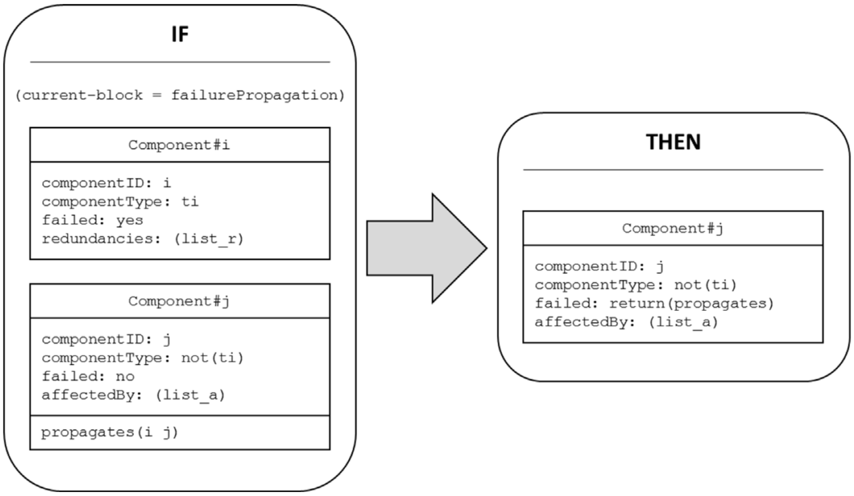


Fig. 5. Failure propagation rule.

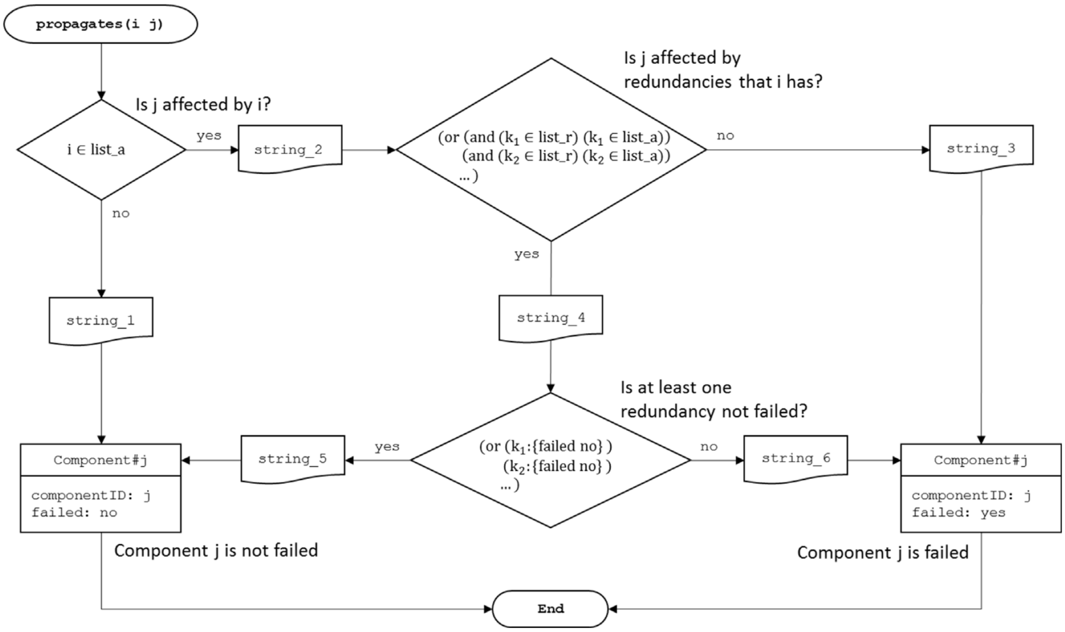


Fig. 6. Procedure [propagates] of class [component].

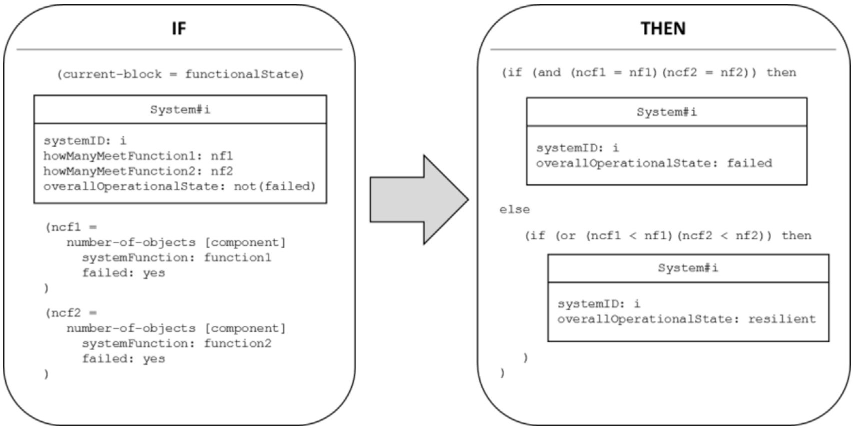

IF
(current-block = functionalState)
System#i
systemID: i
howManyMeetFunction1: nf1
howManyMeetFunction2: nf2
overallOperationalState: not(failed)
(ncf1 =
number-of-objects [component]
systemFunction: function1
failed: yes
)
(ncf2 =
number-of-objects [component]
systemFunction: function2
failed: yes
)
THEN
(if (and (ncf1 = nf1)(ncf2 = nf2)) then
System#i
systemID: i
overallOperationalState: failed
else
(if (or (ncf1 < nf1)(ncf2 < nf2)) then
System#i
systemID: i
overallOperationalState: resilient
)
)

Fig. 7. Rule to determine the functional state of the complex engineering system.

Table 2
Dynamic strings to explain failure propagation.

String_1	i " does not affect functionality of " j ". Thus, a failure in " i " cannot directly propagate to " j "."
String_2	i " affects functionality " j ". A failure in " i " propagates to " j "."
String_3	i " has no redundancies. Thus, a failure in " i " propagates to " j "."
String_4	k " is a redundancy of " i " that affects component " j "."
String_5	"Redundancy " k " is not failed. Thus, failure in " i " does not propagate to component " j "."
String_6	"Since redundancy " k " is failed, failure in " i " can propagate to " j "."

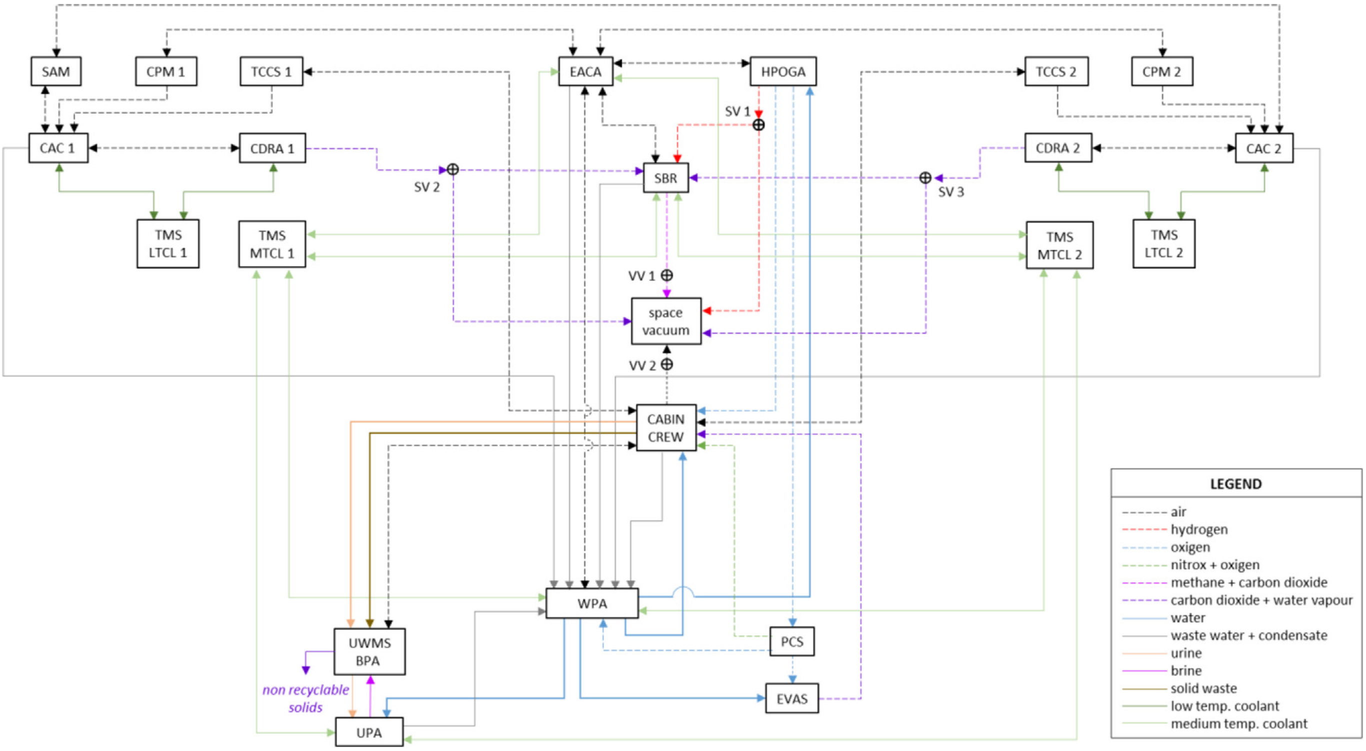


Fig. 8. Proposed design of the environment control and life support system (based on [19]).

The GRDF tool is fully developed in CLIPS[2] a well-known open source shell in the expert system developer community. CLIPS is specifically designed to build rule-based systems and it also supports object-oriented programming.

4. Resilience evaluation of the environment control and life support system

We apply the tool here to evaluate the resilience of a proposed ECLSS for deep space travel. The CES depicted in Fig. 8 is an ECLSS conceptualized based on the work of Stapleton et al. [8]. Table 3 lists the abbreviations used to identify the subsystems presented in Fig. 8. This ECLSS is currently in the conceptual design phase [6,11], so it is an excellent candidate to apply the GRDF tool to evaluate system resilience.

The system depicted in Fig. 8 and the components shown in Table 3 are represented according to the ontology presented in section 3, so it can be simulated by the GRDF tool. The ECLSS main function is sustain life of the crew and is further divided into five sub-functions: remove humidity, remove carbon dioxide, provide oxygen, remove air contaminants and provide drinkable water. In this work, the “acceptable degradation” [10] is considering the ECLSS failed if components that are essential to perform at least one of the life sustaining functions fail in a given simulation. This is a more rigorous failure criterion than that presented in section 3 and it is adopted as a conservative strategy regarding safety sensitive functions.

4.1. Study of failure propagation

Failure propagation is studied by choosing a component from a list of all ECLSS components. The tool then generates the failure propagation trail originated in the chosen component and the conclusions regarding the operational status of each life sustaining function, as well as the overall ECLSS operational status. For example, the failure propagation trail originating in the equipment air cooling assembly (EACA) is presented in Table 4 (the trails of the failures that did not propagate have been omitted for the sake of brevity):

This is an interesting case because even though EACA itself does not perform life-sustaining function, it may eventually cause other components that are life-sustaining to fail. In this particular case, the ECLSS is considered failed because the failure originating in the EACA causes the ECLSS to not perform two life-sustaining functions, namely oxygen generation and drinkable water production. Other life-sustaining functions are not affected by the EACA failure.

It can also be noted that the EACA does not have redundancy. Since it affects many other components, a failure originating in the EACA would easily propagate throughout the system. On the other hand, a failure originating in a component that affects just a few other components does not spread that easily, even if it has no redundancy. This is the case of a failure originating in the universal waste management system-brine processor assembly (UWMS-BPA), as shown in Table 5.

It can be seen that a failure in UWMS-BPA spreads only to the urine processor assembly (UPA). Since none of these components perform life-sustaining functions, these functions are in normal operational state (as far as life support is concerned) and the tool infers that the ECLSS operates in resilient mode.

A component highly connected to others would be expected to

[2] Available at http://clipsrules.sourceforge.net/. Access June 15th, 2018.

Table 3
Abbreviations and subsystems identification in Fig. 8.

Abbreviation	Subsystem	Function	Life-sustaining?
CAC	Condensing Air Cooling	Controls relative humidity of the cabin air (max. 95% [8]).	Yes
CDRA	Carbon Dioxide Removal Assembly	Removes carbon dioxide from cabin air (max. partial pressure 6.8 mmHg [8]).	Yes
CPM	Combustion Products Monitor	Monitors combustion products related to occurrence of fire in the spaceship.	No
EACA	Equipment Air Cooling Assembly	Generates low temperature air to cool down other equipment.	No
EVAS	Extra Vehicular Activity System	Supports astronauts during activities outside the spaceship.	No
HPOGA	High Pressure Oxygen Generator Assembly	Generates oxygen for the cabin air (min. partial pressure 2.7 psia [8]).	Yes
PCS	Pressure Control System	Regulates ambient pressure inside the spaceship and makes up nitrogen and oxygen in case of cabin air venting.	No
SAM	Spacecraft Atmospheric Monitor	Monitors the atmospheric conditions of the spacecraft.	No
SBR	Sebartier Reactor	Produces water and vents methane and exceeding carbon dioxide to the space vacuum.	No
SV	Selector Valve	Directs the mixture of carbon dioxide and water from CDRA to either SBR or space vacuum.	No
TCCS	Trace Contaminant Control System	Removes air contaminants typically found in low concentration, such as ammonia.	Yes
TMS-LTCL	Thermal Management System-Low Temp. Coolant Loop	Provides low temperature coolant for dehumidification purposes.	No
TMS-MTCL	Thermal Management System-Medium Temp. Coolant Loop	Provides medium temperature coolant for refrigeration purposes.	No
UPA	Urine Processor Assembly	Recovers water from urine	No
UWMS-BPA	Universal Waste Managt. System-Brine Processor Assembly	Treats the crew solid wastes and the brine resulting from the UPA	No
VV	Vacuum Valve	Controls cabin vent for space vacuum.	No
WPA	Water Processor Assembly	Provides drinkable water for the crew.	Yes

Table 4
Failure trails originated in Equipment Air Cooling Assembly.

Failure event	Original fail » Propagated fail (first » second)	First component has non failed redundancies	First component's redundancies affect the second one	Second component has non failed redundancies	Life-sustaining function failed
1	EACA » WPA	No	–	No	H_2O production (total)
2	WPA » EVAS	No	–	No	None
3	WPA » UPA	No	–	No	None
4	WPA » HPOGA	No	–	No	O_2 generation (total)
5	HPOGA » PCS	No	–	No	None
6	HPOGA » SBR	No	–	No	None
7	EACA » CPM2	No	–	Yes	None
8	EACA » CPM1	No	–	Yes	None
				ECLSS#1 status: failed mode	

Table 5
Failure trails originated in Universal Waste Management System-Brine Processor Assembly.

Failure event	Original fail » Propagated fail (first » second)	First component has non failed redundancies	First component's redundancies affect the second one	Second component has non failed redundancies	Life-sustaining function failed
1	UWMS-BPA » UPA	No	–	No	None
				ECLSS#1 status: resilient mode	

Table 6
Failure trails originated in the first Thermal Management System-Low Temperature Coolant Loop.

Failure event	Original fail » Propagated fail (first » second)	First component has non failed redundancies	First component's redundancies affect the second one	Second component has non failed redundancies	Life-sustaining function failed
1	TMS-LTCL1 » CAC1	Yes	No	Yes	Hum. removal (partial)
2	CAC1 » TCCS1	Yes	No	Yes	Trace cont. (partial)
3	CAC1 » CDRA1	Yes	No	Yes	CO_2 removal (partial)
4	CDRA1 » SBR	Yes	Yes	No	None
				ECLSS#1 status: resilient mode	

easily propagate failure unless it has redundancy, such as the thermal management system-low temperature coolant loop (TMS-LTCL1), presented in Table 6.

A failure in the TMS-LTCL1 affects two other components that perform life-sustaining functions: condensing air cooling (CAC1) and carbon dioxide removal assembly (CDRA1). However, it has the TMS-LTCL2 as redundancy, so that CAC2 and CDRA2 keep functioning and assuring the ECLSS capability to (at least partially) perform all life-sustaining functions. Also, the CDRA2 function prevents failure propagation to the Sebartier reactor (SBR), since it is redundant to CDRA1.

4.2. Resilience calculation

Because the GRDF tool is based on a Monte Carlo approach, results

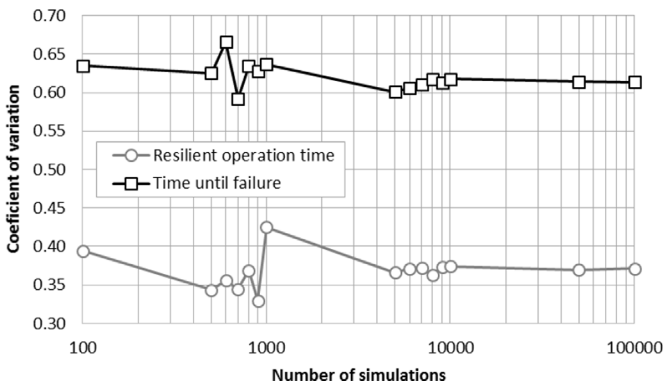


Fig. 9. Effect of the number of simulations on the results.

Table 7
Resilience metrics (N = 8000; T_u = 8760 h; p_i = 0.9995).

Number of simulations	Normal	160
	Failed	5796
	Resilient	2044
Resilience metrics	$p_f(T_u, p_i)$	0.7245
	$p_r(T_u, p_i)$	0.2555
	$\bar{f}\ (T_u, p_i)$	3929 h
	$\bar{r}\,(T_u, p_i)$	1572 h
	$\bar{t}\,(T_u, p_i)$	5051 h
	$\rho(T_u, p_i)$	0.6313

need to ensure statistical significance. The question that needs to be answered is “how many simulations are needed to obtain a specific accuracy of the results?” In that context, the coefficient of variation (Eq. (7)) is a measure of dispersion and, as such, it is expected to converge to a certain value for an infinite number of simulations. We adopted a convergence criteria as $\Delta c_{v,r}/\Delta\bar{r} \leq 10^{-7}$, which is met for 8000 simulations. The coefficient of variation of the resilient operating time and time until failure are plotted against the number of simulations in Fig. 9. Consolidated results obtained from 8000 simulations for ECLSS are shown in Table 7. It is important to keep in mind that the results presented in Table 4 are not related to actual failure prediction; instead, the results strictly follow the generalized resilient design framework based on the ideal failure propagation mechanism described in section 2.

$$c_{v,r} = \bar{r}/\sigma_r \tag{7}$$

We now investigate a modification in the ECLSS configuration in order to assess its impact on resilience. The modification includes the connection of TMS-LTCL1 with CAC2 and CDRA2, maintaining its connection with CAC1 and CDRA1. In the same way, TMS-LTCL2 is connected with CAC1 and CDRA1, maintaining its connection with CAC2 and CDRA2. Additional selector valves (SV) are provided accordingly. The modification is shown in Fig. 10.

The impact of the failure propagation originating from the TMS-LTCL1 in the new configuration (ECLSS#2) is shown in Table 8.

It is interesting to compare the failure propagation originating from the TMS-LTCL1 in the new configuration (ECLSS#2) with the failure propagation from the same component in the previous configuration (ECLSS#1). Because of the new connections, TMS-LTCL1 also affects CAC2 and CDRA2. On the other hand, the failure in TMS-LTCL1 does not propagate to CAC1 and CDRA1 because TMS-LTCL2 is also connected to them. As in the ECLSS#1 configuration, a resilient operating mode was inferred, but all life-sustaining functions are in normal conditions in the ECLSS#2 configuration.

The fault propagation trail in EACA in ECLSS#1 suggests that a failure originating in a component with many connections and no redundancy propagates easily. Thus, a design modification including a redundancy for the EACA is investigated in configuration ECLSS#3, as proposed in Fig. 11. As in the ECLSS#2, a resilient operating mode is inferred with all life-sustaining functions in normal conditions for ECLSS#3. Simulation results for ECLSS#2 and ECLSS#3 are compared to those from ECLSS#1 in Table 9.

According to the metrics presented in Table 9, configurations ECLSS#2 and ECLSS#3 are more resilient design alternatives than ECLSS#1. Significant reduction in the number of failed simulations is observed in both alternatives, with respective increase in the number of resilient operations. The metrics indicate that alternatives ECLSS#2 and ECLSS#3 are less prone to completely fail (low p_f) and have higher probability of resulting in resilient operation (high p_r), staying longer in resilient operation (high $\bar{r}$), having higher time until failure (high $\bar{f}$), higher average operating time (high $\bar{t}$) and higher resilience index (high

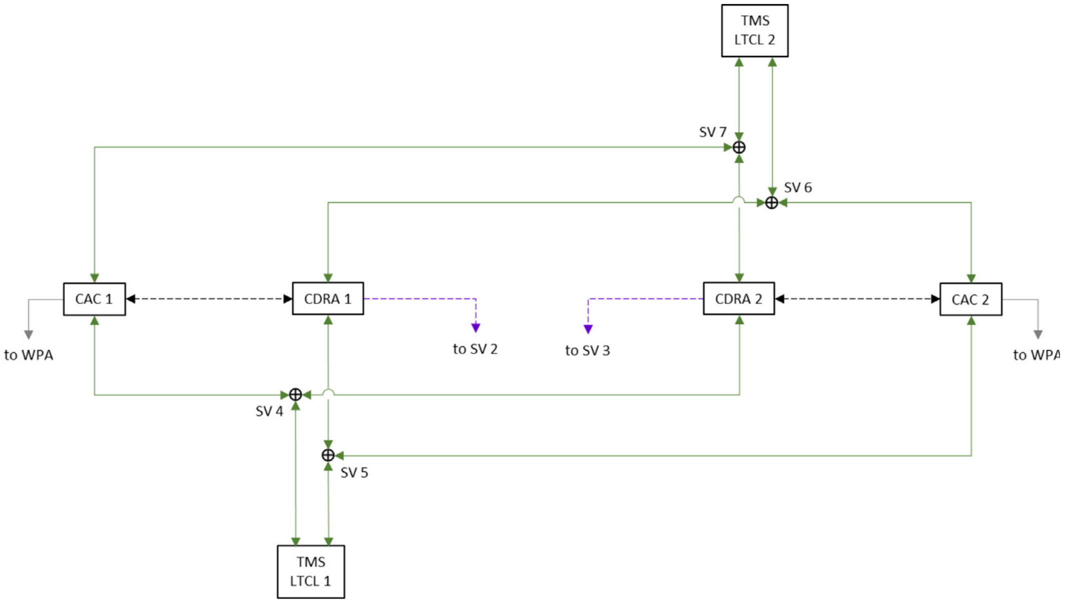


Fig. 10. Alternative configuration #2 of the environmental control and life support system.

Table 8
Failure trails originated in the first Thermal Management System-Low Temperature Coolant Loop (new configuration).

Failure event	Original fail » Propagated fail (first » second)	First component has non failed redundancies	First component's redundancies affect the second one	Second component has non failed redundancies	Life-sustaining function failed
1	TMS-LTCL1 » CAC2	Yes	Yes	Yes	None
2	TMS-LTCL1 » CAC1	Yes	Yes	Yes	None
3	TMS-LTCL1 » CDRA2	Yes	Yes	Yes	None
4	TMS-LTCL1 » CDRA1	Yes	Yes	Yes	None
ECLSS#2 status: resilient mode					

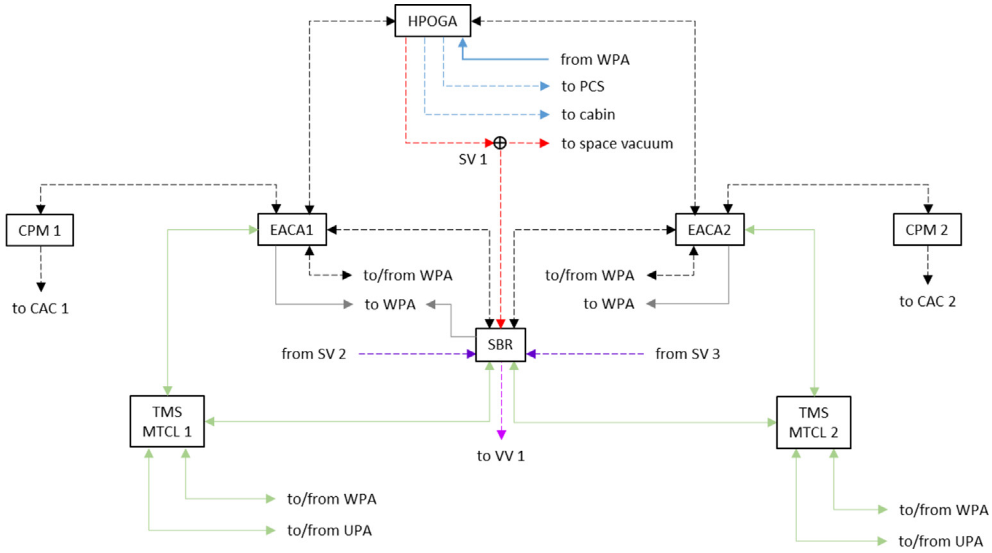


Fig. 11. Alternative configuration #3 of the environmental control and life support system.

Table 9
Comparison between different designs ($N = 8000$; $T = 8760\ h$; $p_i = 0.9995$).

ECLSS#	N_n	N_f	N_r	p_f	p_r	$\bar{f}$ (h)	$\bar{r}$ (h)	$\bar{t}$ (h)	ρ
1	160	5796	2044	0.7245	0.2555	3929	1542	5051	0.6313
2	144	5165	2691	0.6456	0.3364	4086	2061	5473	0.6841
3	164	5300	2536	0.6625	0.3170	4239	1883	5508	0.6885

Table 10
Resilience index comparison ($N = 8000$; $T = 8760\ h$; $p_i = 0.9995$).

ECLSS#	Lower limit	Resilient index ρ	Upper limit
3	0.5299	0.6885	1
2	0.5108	0.6841	1
1	0.4911	0.6313	1

ρ). More specifically, ECLSS#2 presented the lowest probability of failure, the highest probability of resilient operation and the highest resilient operating time, as long as ECLSS#3 presented the highest time until failure, the highest average operating time and the highest resilience index. Since the resilience index is a metric that is related to all others metrics, it is probably fair to state that alternatives ECLSS#2 and ECLSS#3 presents practically the same resilience. Resilience index comparison is shown in Table 10.

Redundancy is a natural way to increase the resilience and the fault propagation trails originating in EACA1 of ECLSS#3 corroborate that, as shown in Table 11.

Redundancy EACA2 is not failed. Thus, failure in EACA1 does not propagate to HPOGA.

Moreover, the results presented in Table 9 show that ECLSS#3 is more resilient than ECLSS#1 as all metrics improve. It is interesting to note that the resilience analysis during the conceptual design phase shows that redundancy is not the only way to increase resilience and in fact not always the best way, as the comparison of ECLSS#3 to ECLSS#2 indicate (ρ_3 is only 0.6% greater than ρ_2). One has to keep in mind that in deep space missions, weight, volume and costs are very restrictive constraints, so the alternative based on EACA redundancy is heavier, bulkier and more expensive, but it is almost as resilient as ECLSS#2, which differs from ECLSS#1 basically in the number of

Table 11
Failure trails originated in the first Thermal Management System-Low Temperature Coolant Loop (new configuration).

Failure event	Original fail » Propagated fail (first » second)	First component has non failed redundancies	First component's redundancies affect the second one	Second component has non failed redundancies	Life-sustaining function failed
1	EACA1 » WPA	Yes	Yes	No	None
2	EACA1 » CPM1	Yes	No	Yes	None
3	EACA1 » SBR	Yes	Yes	No	None
4	EACA1 » HPOGA	Yes	Yes	No	None
ECLSS#3 status: resilient mode					

connections, with a smaller impact on the overall weight, volume and costs. This illustrates the importance to carry out resilience analysis to support the design team during the conceptual design phase by providing information on fault propagation trails and resilience metrics.

In this study, we focused on the conceptual design stage where notional design solutions are being considered. In resilience analysis of systems as complex as a life support for deep space travel, an experimental validation would be rather complex as it involves the observation of a statistically significant set of design realizations over their entire lifetime of the system. This is one of the reasons why we propose a methodology based on “computers experiments”, such as Monte Carlo simulations. Costs and time would be also major restrictions in an experimental validation, especially for complex engineered systems. Other researches in the field also acknowledge the need and the difficulties to validate resilient design methodologies [11,19]. Indeed, Fang et al. [19] identify very few works tackling the problem of validation of resilience analysis based on network-centric approaches, which indicates that this is still an open issue in the field.

5. Conclusion

In this article, we presented a Generalized Resilient Design Framework (GRDF) to evaluate the resilience of a proposed Environmental Control and Life Support System (ECLSS) for deep space travel. Without loss of generality, the GRDF is based on an ideal failure propagation mechanism from which we obtained new metrics for resilience of complex engineered systems. The GRDF is embedded in a computational tool that combines rules and a Monte Carlo approach in order to: i) establish the failure propagation mechanism, resulting in a collection of failed components; ii) verify if the collection of failed components cause the system to fail; iii) compute the resilience metrics. The impact of a given failure propagation can be provided by the GRDF tool by inquiring what would happen if a particular component failed. In the ECLSS case, the tool gives insights regarding redundancy and connectivity between components, so two design alternatives were proposed: one based on new connections and another one with extra redundancy. Both modifications resulted in more resilient designs, but the design alternative with redundancy resulted in a resilience index practically equal to the resilience index of the design alternative with new connections. This is a noteworthy finding, because redundancy is intuitively considered as a natural way to increase resilience. However, in a deep space mission weight, volume and costs pose restrictive constraints. The alternative based on redundancy – while as resilient as the alternative based on new connections – is heavier, bulkier and more expensive. The GRDF tool adds utility to the design team by providing rationales and metrics for the most resilient design, although validation of resilient design methodologies is still an open issue.

Acknowledgements

To Jonathan D. O'Neill, from UTC Aerospace Systems, for providing advice on the design of environmental control and life support systems.

To FAPESP – São Paulo Research Foundation, for sponsoring the postdoc internship of the first author at NASA Ames Research Center (Grant number: 2016/19255-7).

Mr. O'Neill just helped us to better understant the system conceived by UTC that we studied in our paper. NASA had me as a visitng scientist during the year 2017.

References

[1] S.G. Love, R.P. Harvey, Crew autonomy for deep space exploration: lessons from the antarctic search for meteorites, Acta Astronaut. 94 (1) (2014) 83–92.
[2] R. Peldszus, H. Dalke, S. Pretlove, C. Welch, The perfect boring situation - addressing the experience of monotony during crewed deep space missions through habitability design, Acta Astronaut. 94 (1) (2014) 262–276.
[3] S. Aydogan-Cremaschi, S. Orcun, G. Blau, J.F. Pekny, G.V. Reklaitis, A novel approach for life-support-system design for manned space missions, Acta Astronaut. 65 (3–4) (2009) 330–346.
[4] B. Xie, et al., The water treatment and recycling in 105-day bioregenerative life support experiment in the Lunar Palace 1, Acta Astronaut. 140 (2017) 420–426 January.
[5] J. Gruenwald, A hybrid plasma technology life support system for the generation of oxygen on Mars: considerations on materials and geometry, Acta Astronaut. 123 (2016) 188–191.
[6] S. Do, A. Owens, K. Ho, S. Schreiner, O. De Weck, An independent assessment of the technical feasibility of the Mars One mission plan - updated analysis, Acta Astronaut. 120 (2016) 192–228.
[7] NASA, NASA's Journey to Mars, Pioneering Next Steps in Space Exploration, (2015).
[8] T.J. Stapleton, et al., Environmental control and life support for deep space travel, ICES-2016: 46th International Conference on Environmental Systems, 2016, pp. 1–9 10–14 July 2016, Vienna, Austria.
[9] W. Harwood, Astronauts Service Space Station's Air Purifier, Oxygen Generator (Updated), CBS News, May 2011 2011.
[10] Y.Y. Haimes, On the definition of resilience in systems, Risk Anal. 29 (4) (2009) 498–501.
[11] H. Mehrpouyan, B. Haley, A. Dong, I.Y. Tumer, C. Hoyle, Resiliency analysis for complex engineered system design, Artif. Intell. Eng. Des. Anal. Manuf. AIEDAM 29 (1) (2014) 93–108.
[12] X. Zhang, S. Mahadevan, S. Sankararaman, K. Goebel, Resilience-based network design under uncertainty, Reliab. Eng. Syst. Saf. 169 (2018) 364–379 March 2017.
[13] J.A. Matelli, K. Goebel, Conceptual design of cogeneration plants under a resilient design perspective: resilience metrics and case study, Appl. Energy 215 (2018).
[14] R. Patriarca, J. Bergström, G. Di Gravio, F. Costantino, Resilience engineering: current status of the research and future challenges, Saf. Sci. 102 (December 2016) 79–100 2018.
[15] D.D. Woods, Four concepts for resilience and the implications for the future of resilience engineering, Reliab. Eng. Syst. Saf. 141 (5–9) (2015).
[16] T. Stapleton, et al., Environmental control and life support system developed for deep space travel, ICES-2017: 47th International Conference on Environmental Systems, Charleston, South Carolina, 2017, pp. 1–10 16–20 July 2017.
[17] J.C. Da Silva, J.A. Matelli, E. Bazzo, Development of a knowledge-based system for cogeneration plant design: verification, validation and lessons learned, Knowl. Base Syst. 67 (2014) 230–243.
[18] J.A. Matelli, Conceptual design of biomass-fired cogeneration plant through a knowledge-based system, J. Brazilian Soc. Mech. Sci. Eng. 38 (2) (2016) 535–549.
[19] Y. Fang, N. Pedroni, E. Zio, Optimization of cascade-resilient electrical infrastructures and its validation by power flow modeling, Risk Anal. 35 (4) (2015) 594–607.