Optimization of Development Assurance Level Allocation

Kevin Delmas, Laura Chambert, Christophe Frazza, Christel Seguin

Optimization of Development Assurance Level Allocation

Kevin Delmas
ONERA
Toulouse, France
kevin.delmas@onera.fr

Laura Chambert
SII Sud-Ouest
Toulouse, France
laura.chambert@laposte.net

Christophe Frazza
SATODEV
Toulouse, France
christophe.frazza@satodev.fr

Christel Seguin
ONERA
Toulouse, France
christel.seguin@onera.fr

***Abstract*—In the aeronautics domain, development errors may contribute to catastrophic failures and they shall be prevented by appropriate assurance activities. So the aeronautics standards propose Development Assurance Levels (DAL), which define the levels of development rigor applicable to functions, software or hardware items of an aircraft.**

The allocation of DALs to items follows rules. Basically, the DAL of each item shall be proportionate to the severity of the effects of the item development errors. Moreover, severe failures may results from a combination of independent development errors of several items. In such case, additional rules introduce the possibility to downgrade the DAL levels of the independent items. Thus, many DAL allocations are possible for a given system.

Consequently, we have investigated means to assist the safety specialists when verifying or optimizing a DAL allocation. We propose a formalization of the DAL allocation as an optimization problem integrating user-defined constraints and cost criteria used to focus the exploration on most interesting allocations.

Experimentation is conducted on large-scale aeronautics systems to highlight the scalability and benefits of our approach compared to the heuristic-based approach.

I. Introduction

In the aeronautics domain, systematic development errors may contribute to catastrophic events. To tackle this issue, the aeronautics standards ([1], [2]) propose to allocate a Development Assurance Level (DAL) indicating the level of rigor of the development of function, a software or hardware item of an aircraft. The DAL guides the assurance activities that should be applied at each stage of development to eliminate design errors that would have a safety effect on the aircraft.

The DAL allocation is risk-driven, *i.e.*, based on the severity of the effects of a function specification or implementation error. Moreover, severe failures may results from a combination of independent development errors of several items. In such case, additional rules introduce the possibility to downgrade the DAL levels of the independent items. Thus, many DAL allocations are possible for a given system.

Among the DAL allocations satisfying the safety constraints, designers are trying to find the one minimizing the development costs. A manual and exhaustive exploration is intractable for industrial systems; therefore, designers rely on allocation heuristics that may lead to non-optimal or even incorrect solutions. Consequently, we have investigated means to assist the safety specialists when verifying a manual allocation or optimizing a DAL allocation.

In this paper, we extend a previous formalization of the DAL allocation problem as a pseudo-Boolean constraint-based optimization problem presented in [3]. This formalization links the maximal allowed reduction of DAL and the independence of members appearing in the minimal functional failure sets (FFS) leading to a failure condition. We recall in Section II the minimal concepts on which the presented approach is built upon.

To enable DAL allocation optimization we introduce, in Section III, optimization criteria based on the DAL cost of each function and the number of functions that must be independent. Furthermore, we introduce in Section III user constraints used to focus the exploration on more interesting allocations. For instance, the designer can enforce a given range of possible DAL for a function, or add dependencies between the DAL allocated to distinct functions. This formalization enables to solve the problem using very efficient constraint solvers and to provide optimality guarantees. A fully automated tool implementing the proposed formalization provides DAL allocation and a set of function independence requirements from safety analyses and user constraints.

Section IV details the experimentations conducted on several systems and highlights the benefits of the constraints-based allocation approach compared to the heuristic-based approach. We also demonstrate that the scalability of the approach enables the usage of our allocation tool on industrial systems.

II. Background

A. Development Assurance Level

DALs are agreed qualitative levels of development rigor which are requested to prevent systematic faults. This approach is shared by aeronautics and other industries of safety-critical domains such as space, nuclear, railway or automotive [4].

In aeronautical standards [1], DALs range from E (poor assurance of development rigor) to A (highest assurance of development rigor). They are required for functions (FDAL), software and hardware items (IDAL). Complementary standards clarify how to reach the targeted DAL according to the developed technology (e.g. [5] for software or [6] for complex hardware).

Moreover, the DAL requirements have to be proportioned to the severity of the effects of potential design errors. So DALs are determined by the system safety assessment process. One activity of this process is the identification of all the Functional Failure Sets (FFSs) leading to safety critical failure conditions. A FFS is defined by [1] as a *set of one, or more members that are considered to be independent from one another, whose development errors leads to a top level failure condition*. The section IV will present some methods used to compute the FFSs for concrete cases.

In the following, we assume we have means to compute FFSs and we focus on the use of FFSs to verify or optimize the DAL allocation. We will consider that we either perform an FDAL or an IDAL allocation, thus that FFSs contain only either functional development errors or item development errors. Therefore we will not distinguish these two allocations and simply consider them as DAL allocations. The problem of an allocation mixing functional and item development errors are left aside for future works.

B. Illustrative Use case

Let us first use the simple display system described in [3]. The architecture, provided by the Figure 1, is composed of two sensors denoted SL and SR; two display units denoted DL and DR; and four data streams denoted $DF\_SL\_to\_DL$, $DF\_SL\_to\_DR$, $DF\_SR\_to\_DL$ and $DF\_SR\_to\_DR$. The failure condition considered in this example is *"both display units DL and DR stop working"* and it's severity is HAZ.

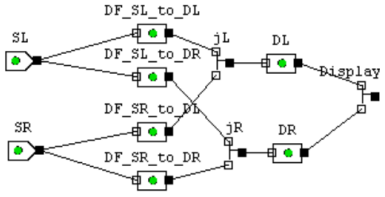

Fig. 1. Display system physical architecture

Let us suppose that an analysis tool identified the FFSs provided by Table I that could lead to the loss of data display on both screens.

As the display system is part of a large civil aircraft, the initial DAL corresponding to HAZARDOUS failure condition is B. The objective is to perform a DAL allocation:

- compliant with the allocation constraints applicable to aeronautics systems [1];
- taking into account common mode development errors and/or the lack of item dissimilarity;
- minimizing a cost criterion based on a coarse estimation of the development cost of an item at a given DAL level.

C. DAL costs

The DAL associated with a software (resp., complex hardware) item guides the assurance activities that have to be performed during its development following DO178 [5] (resp., DO254 [6]). The higher the DAL, the more detailed and rigorous are the assurance activities to be performed. For instance, Table II describes three objectives of the Software Coding and Integration Process.

It indicates which objectives are applicable at a given DAL level. A cell containing R means that this is a Required objective at this level, a blank cell means the objective is not required and a cell containing I means that the objective should be achieved with independence.

High DALs require a great number of assurance activities. The increase in the level of rigor, level of detail and the need to involve independent teams leads to greater development cost of software and hardware items. Consequently, designers aim at allocating a DAL to software and hardware as low as possible, within the bounds imposed by safety regulation, in order to reduce the development cost of their systems.

Studies such as [7] provide an evaluation of the DAL cost evolution w.r.t. a baseline cost (for E DAL). Thus, a user may want to provide for each item or function and for each DAL its cost estimation. In the following, we will reuse the cost evolution of [7].

A coarse estimation of the cost of a DAL allocation can be obtained by summing the individual costs of the system's components. An objective is then to find the optimal DAL allocation(s) w.r.t. this cost criterion and compliant to allocation rules.

D. Allocation Rules

The rules for assurance level allocation follow the same paradigm in various safety critical domains. Usually, severity classes are defined for failure conditions (e.g. CATASTROPHIC, HAZARDOUS, MAJOR, MINOR, No Safety Effect). Domain-specific tables assign a DAL objective to each severity class. It is worth noting that an item error may contribute to failure conditions of different severity. Then the item will inherit from the DAL objective of the most severe failure condition.

For instance, the table applicable to systems of civil aircraft is provided in [1]. It indicates that functions or items whose development errors are contributing to CATASTROPHIC failure conditions must be allocated a DAL A.

Example 1 *(Initial DAL) The failure condition of the Display system is HAZARDOUS so the corresponding DAL is B.*

Another example for Unmanned Aerial Systems (UAS) can be found in [8]. This document identifies different classes of UAS according to their weight and level of automation. For these systems, the table assigns a DAL B to items or functions whose development errors are contributing to CATASTROPHIC failure conditions. Moreover, the takes into account architectural hypothesis to downgrade the DAL. For instance, a light UAS may have independent primary and secondary systems to maintain safe flight and landing. In such a case, the table accepts DAL C for each system (instead of B).

Id	Order	FFS			
1	2	DL.fail_loss	DR.fail_loss		
2	2	SL.fail_loss	SR.fail_loss		
3	3	DF_SL_to_DL.fail_loss	DF_SL_to_DR.fail_loss	SR.fail_loss	
4	3	DF_SL_to_DL.fail_loss	DF_SR_to_DL.fail_loss	DR.fail_loss	
5	3	DF_SL_to_DL.fail_loss	DR.fail_loss	SR.fail_loss	
6	3	DF_SL_to_DR.fail_loss	DF_SR_to_DR.fail_loss	DL.fail_loss	
7	3	DF_SL_to_DR.fail_loss	DL.fail_loss	SR.fail_loss	
8	3	DF_SR_to_DL.fail_loss	DF_SR_to_DR.fail_loss	SL.fail_loss	
9	3	DF_SR_to_DL.fail_loss	DR.fail_loss	SL.fail_loss	
10	3	DF_SR_to_DR.fail_loss	DL.fail_loss	SL.fail_loss	
11	4	DF_SL_to_DL.fail_loss	DF_SL_to_DR.fail_loss	DF_SR_to_DL.fail_loss	DF_SR_to_DR.fail_loss

TABLE I
FFSs OF THE DISPLAY USECASE

Indeed, such a kind of architectural hypothesis specifies that the severe failure condition does not result from a single development error but does result from a combination of independent development errors of several items. In such a case, additional rules introduce the possibility to reduce the DAL levels of the independent items.

Let us here illustrate succinctly the reduction rules of [1]. The interested reader can find a detailed description and formalisation of the reduction rules in [3].

Example 2 *Let us consider the FFS {DF_SL_to_DL.fail_loss, DF_SL_to_DR.fail_loss, SR.fail_loss } of the Display system, if these development errors are independent then:*

- ***Option 1*** *A first reduction option is to keep one item (*e.g.*, $DF\_SL\_to\_DL$) at least at the initial DAL (here B) and ensure that other items are allocated a DAL at least equal to the initial DAL minus two levels (here D).*
- ***Option 2*** *A second reduction option is to keep two items (*e.g.*, $DF\_SL\_to\_DL$ and $DF\_SL\_to\_DR$) at least at the initial DAL minus one (here C) and ensure that other items are allocated a DAL at least equal to the initial DAL minus two levels (here D).*

Obviously there are many possible allocations that can be obtained by applying these rules. Moreover, the development error independence plays a prominent role during the DAL allocation. For instance, three possible DAL allocations are detailed in Table III where the last one considers two common development errors. A common development error affecting {DL, SL, DF_SL_to_DL, DF_SL_to_DR} and another one affecting {DR, SR, DF_SR_to_DR, DF_SR_to_DL}.

The allocations provided by Table III illustrates the impact of the DAL reduction rule choice on the DAL allocation of DL, DR, SL and SR. For instance, theirs development errors contribute to the #1 FFS of Table I so by selecting:

- Option 1 DR remains at B and DL is degraded at D.
- Option 2 DR and DL are degraded at C.

Note that common development errors do not affect the DAL allocation obtained using Option 2. This result demonstrates that the proposed allocation using Option 2 does not rely on independence assumption among the development errors of {DL, SL, DF_SL_to_DL, DF_SL_to_DR} and of {DR, SR, DF_SR_to_DR, DF_SR_to_DL}.

Eventually, a user can consider that some DAL allocations are not valuable (even if compliant to the regulation). One may want to allocate the same DAL to similar items or functions, or enforce that specific items or functions cannot be implemented for some DAL level.

Objective		Applicability			
Description	Ref	A	B	C	D
Software high-level requirements comply with system requirements.	6.3.1a	I	I	R	R
High-level requirements are accurate and consistent.	6.3.1b	I	I	R	R
High-level requirements are compatible with target computer.	6.3.1c	R	R		

TABLE II
EXTRACT OF THE REQUIREMENT TABLE OF [6]

III. DAL ALLOCATION AS AN OPTIMIZATION PROBLEM

The following work is based on the formalisation of the DAL allocation problem based on pseudo Boolean logic presented in [3]. So let us remind first in Section III-A the minimal concepts used by [3] to formalise the DAL allocation problem. We then extend the approach to handle common development errors (Section III-B), dissimilarity assumptions (Section III-C) and cost-based optimization (Section III-D). Note that we only present the formal ground of these extensions but a user language has been defined to specify these constraints to the DALCULATOR .

A. Reminder on DAL allocation problem formalisation

ONERA proposed a method and a tool to solve the DAL allocation problem [3]. The inputs of this method are:

- the failure conditions of the system and their severity;

Item	DAL allocation with		
	Option 1	Option 2	Common causes
DF_SL_to_DL	D	D	D
DF_SL_to_DR	B	C	C
DF_SR_to_DL	D	C	C
DF_SR_to_DR	D	D	D
DL	D	C	C
DR	B	C	C
SL	B	C	C
SR	D	C	C

TABLE III
POSSIBLE DAL ALLOCATIONS FOR THE DISPLAY SYSTEM

- the FFSs leading to the failure conditions;
- DAL allocation rules applicable to the system;
- Designer allocation constraints and optimisation directives.

Then the tool searchs an allocation of DAL for all the items which occur in FFSs such that the allocation is compliant both with the standard allocation rules and the with the designer constraints. This a decision problem and the tool has to decide whether it is true or false that an item f has a DAL d greater or equal to a given level ($A, B, CorD$)

The formalisation introduced in [3] is based on an encoding of the above information and the allocation rule of [1] using pseudo-Boolean logic. A pseudo-Boolean variable can be valuated either at 1 or 0. It can then be used in classical Boolean logic formula using Boolean connectives (*e.g.,* $\vee$) or in pseudo-Boolean constraints (*e.g.,* $2v_1 + 3v_2 \geq 5$).

The decision variable used in the formalisation of [3] is provided in Definition 1.

Definition 1 *(Decision variable) Let $\mathcal{F}$ be the set of item or function identifiers and let $\mathcal{D} = \{A, B, C, D\}$ be the ordered set of DAL level such that $D < C < B < A$.*

Then the set of decision variables $\mathcal{V}$ over $\mathcal{F}$ and $\mathcal{D}$ is :

$$\mathcal{V} = \{v_{f,d} | f \in \mathcal{F}, d \in \mathcal{D}\}$$

$v_{f,d} = 1$ if and only if the DAL of f is greater or equal to d else $v_{f,d} = 0$

Example 3 *(Decision variable) The DAL allocation for S_L with Option 1 provided in Table III is encoded as follows:*

$$v_{S\_L,A} = 0,\ v_{S\_L,B} = 1,\ v_{S\_L,C} = 1,\ v_{S\_L,D} = 1$$

For the sake of simplicity, Definition 2 introduces an auxiliary formula encoding that the DAL of a given item or function is equal to a given level.

Definition 2 *(DAL allocation formula) Let $f \in \mathcal{F}$ and $d \in \mathcal{D}$, then $dal(f, d)$ is true iff the DAL of f is equal to d that is:*

$$dal(f, d) \Leftrightarrow \bigwedge_{d' > d} \neg v_{f,d'} \wedge \bigwedge_{d' \leq d} v_{f,d'}$$

Example 4 *(DAL allocation formula) For the DAL allocation for S_L with Option 1 provided in Table III we have:*

$$dal(S\_L, A) = 0,\ dal(S\_L, B) = 1,$$
$$dal(S\_L, C) = 0,\ dal(S\_L, D) = 1$$

The authors of [3] define pseudo-Boolean constraints (denoted $\mathcal{C}_u$) enforcing that DAL allocations are compliant to the allocation rules of [1]. We will not present in detail the actual constraints within C_u that are already provided in [3].

Finding an optimal DAL allocation is therefore formalised as finding a solution of the problem defined in Definition 3.

Definition 3 *(DAL allocation problem) The DAL allocation problem is a lexicographic min optimisation problem defined as:*

$$\begin{cases} lex \min \sum_{f \in \mathcal{F}} dal(f, A), \ldots, \sum_{f \in \mathcal{F}} dal(f, D) \\ subject\ to\ C_u \end{cases}$$

The current modelling of the DAL allocation raises however some limitations. For instance, it is not possible to provide a custom definition of the cost criterion used to drive the exploration. Moreover, some user constraints such as the consideration of common development errors are not considered. The remaining of this section therefore provides an extension of the formalisation overcoming these limitations.

B. Dealing with common development error

To formalise the impact of a common development error on DAL allocation, we consider that the DAL allocated to the components affected by the common development errors should be greater or equal to the one allocated to the common development error event. So here we need to introduce DAL constraints between the allocation on two elements of the problem. This extension is formalised by Definition 4.

Definition 4 *(DAL allocation constraint) Let l and r be two items or functions of $\mathcal{F}$. The user can enforce the following constraints:*

$$dal(l) \odot dal(r)$$

where $\odot \in \{<, \leq, =, >, \geq\}$

We only provide in Definition 5 the encoding when $\odot$ is $\geq$, the other cases are easily derived from this encoding.

Definition 5 *(DAL allocation constraint encoding) Let l and r be two items or functions of $\mathcal{F}$. Then:*

$$dal(l) \geq dal(r) \Leftrightarrow \bigwedge_{d \in \mathcal{D}} v_{r,d} \Rightarrow v_{l,d}$$

Thanks to this extension, Definition 6 encodes the effect of a common development error on the DAL allocation.

Definition 6 *(Common cause modelling) Let e be the common development error* CCF_SW_COM_Incorrect *affecting a subset C of $\mathcal{F}$. Then DALs allocated to the elements of C should be greater or equal to the one associated to e:*

$$\forall c \in C, dal(c) \geq dal(e)$$

Example 5 *(Common cause modelling) Let CC_S be a common development error leading to the loss of both S_L and S_R. Then {CC_S} will be a new FFS w.r.t. Table I. The new constraint will be:*

$$dal(S\_L) \geq dal(CC\_S) \wedge dal(S\_R) \geq dal(CC\_S)$$

C. Exploring dissimilarity assumptions

By default the DALCULATOR assumes that the development errors of all items or functions are independent. Nevertheless, the user can specify that some development errors may be dependent due to item/function similarity. The authors of [3] provided an encoding of this independence constraint through a set of decision variables introduced in Definition 7.

Definition 7 *(Independence decision variables) Let $\mathcal{F}$ the set of items or functions, then the set of independence decision variables denoted $\mathcal{I}$ is defined as follows:*

$$\mathcal{I} = \{indep(l, r) | l \neq r \in \mathcal{F}\}$$

where $indep(l, r)$ stands for the development errors of l are independent of the ones of r.

Thanks to this encoding, the user can enforce that a set of items or functions are similar, hence their development errors should not be considered as independent. Such a constraint will prevent any DAL reduction on FFSs that do not contain independent development errors.

Definition 8 *(No dissimilarity constraint) Let $S \subset \mathcal{F}$ a set of items or functions which are considered as similar, this constraint is encoded as:*

$$\bigwedge_{l \neq r \in S} \neg indep(l, r)$$

Example 6 *(No dissimilarity constraint) Let assume that all links are not dissimilar in the Display example, this can be specified as:*

$$\begin{aligned}
&\neg indep(DF_SL_to_DL, DF_SL_to_DR)\\
\wedge &\neg indep(DF_SL_to_DL, DF_SR_to_DL)\\
\wedge &\neg indep(DF_SL_to_DL, DF_SR_to_DR)\\
\wedge &\neg indep(DF_SL_to_DR, DF_SR_to_DL)\\
\wedge &\neg indep(DF_SL_to_DR, DF_SR_to_DR)\\
\wedge &\neg indep(DF_SR_to_DL, DF_SR_to_DR)
\end{aligned}$$

D. Performing a cost-based DAL allocation optimisation

The last extension enables a user to perform cost-based DAL allocation optimisation. Referring to the work of [7], we enable the user to provide a cost (using arbitrary units) to develop a given item or function f at a given DAL d.

Definition 9 *(Cost constants) Let $f \in \mathcal{F}$, $d \in \mathcal{D}$ and $n \in \mathbb{N}$. Then a user specifies that developing f at level d costs n arbitrary units as follows:*

$$cost(f, d) = n$$

Example 7 *(Cost constants) The costs associated to S_L can be defined as follows:*

$$\begin{aligned}
cost(S_L, D) = 105 \;\; cost(S_L, C) = 137\\
cost(S_L, B) = 157 \;\; cost(S_L, A) = 165
\end{aligned}$$

The new DAL allocation problem is thus an optimisation of a single pseudo-Boolean criterion as formalised in Definition 10.

Definition 10 *(DAL allocation problem) The DAL allocation problem is a lexicographic min optimisation problem defined as:*

$$\begin{cases} \sum_{f \in \mathcal{F}, d \in \mathcal{D}} cost(f, d)\, dal(f, d) \\ \textit{subject to } C_u \end{cases}$$

IV. TOOL AND EXPERIMENTATION

A. Experimental method

To illustrate the adaptability of the DALCULATOR , we considered systems that are not submitted to the same certification framework. Therefore we explain for each of them what are the DAL allocation rules that have been considered.

In the following sections we detail for each experiment the failure conditions and their severity. Concerning the FFSs, we choose to use Model-Based Safety Assessment (MBSA) and more specifically AltaRica [9] to formalize the safety outcomes of development errors. For the sake of conciseness, we do not detail the models and the generated FFSs but provide instead extracts. The interested reader can find the FFS files used to perform the DAL allocation at https://w3.onera.fr/PHYDIAS/tools.

Eventually, one purpose of the DALCULATOR is to explore the possible DAL allocation. To do so, the user can specify allocation and independence constraints to assess their impact on the DAL allocation. Moreover, the user can provide the development cost for each DAL level for each item to obtain the optimal DAL allocation. The use of these constraints are illustrated by the following experiments.

The objective of the experiments is:

1) to verify that the DALCULATOR is able to generate DAL allocations that are consistent with the one provided on well studied systems *i.e.*, the Wheel Braking System of [2];
2) to illustrate how the DALCULATOR can be used to explore possible DAL allocations on a complex system, in our case a fixed wing Unmanned Aerial Vehicle (UAV).
3) to demonstrate that the DALCULATOR is able to provide DAL allocations on large systems, in our case an aircraft communication system.

B. DAL allocation verification

A first experiment has been conducted on the Wheel Braking System (WBS) that is a well studied system used as a running example of the [2] to illustrate the safety assessment and DAL allocation principles. The purpose of the WBS is to decelerate the aircraft on the ground. The WBS performs this function automatically upon landing or manually upon pilot activation. The WBS is also used for directional control on the ground through differential braking, stopping the main landing gear wheel rotation upon gear retraction, and preventing aircraft motion when parked.

Failure condition	FFS	
	order	number
Loss of Braking	1	2
	2	26
	3	92
	4	490
Untimely Braking	1	2
	2	15
	3	47
	4	85

TABLE IV
FFSs REPARTITION FOR THE BRAKING SYSTEM

A MBSA model[1] of the WBS was developed by Pierre Darfeuil, Christophe Frazza and Jean Gauthier in parallel with activities carried out by the EUROCAE WG63 and SAE S18 working groups for drafting the ED-135A / ARP 4761A guides. The comprehensive model formally specifies how the development errors of the items of the WBS may lead to the following failure conditions identified in [2] :

- **Loss of Braking**: Total loss of wheel deceleration (80% coverage or more) considered as HAZARDOUS.
- **Untimely Braking**: Untimely full symmetric wheel deceleration considered as CATASTROPHIC.

Automatic analyzers are then used to derive the FFS for the two failure conditions. The results are summarized in Table IV.

The model contains approximately 80 items, among which 27 contribute to the failure conditions. The Table V provides a selection of items and the numbers of FFSs they contribute to (limited to FFSs of atmost order 3).

As the WBS is part of a large civil aircraft, the initial DAL corresponding to CATASTROPHIC (resp., HAZARDOUS) failure conditions is A (resp., B).

In the model some common development errors are considered and modelled as specific development errors.

Example 8 *(Common cause development errors) Let* CCF_SW_COM_Incorrect *be the common development error of the two monitor's software* Monitor1 *and* Monitor2*. Then the DAL allocated to* Monitor1 *and* Monitor2 *should be greater or equal to the one allocated to* CCF_SW_COM_Incorrect*:*

$$dal(Monitor1) \geq dal(CCF\_SW\_COM\_Incorrect)$$
$$dal(Monitor2) \geq dal(CCF\_SW\_COM\_Incorrect)$$

[1]available at https://satodev.com/nos-produits/cecilia-workshop/

Item	Failure condition					
	Loss of Braking			Untimely Braking		
	1	2	3	1	2	3
Accumulator	0	1	31	0	0	0
Alt_EmerMeterValve_L	0	0	10	0	0	5
Alt_EmerMeterValve_R	0	0	10	0	0	5
CCF_HW_COM_MON	1	2	8	1	0	3
CCF_SW_COM	0	7	16	0	5	6
CCF_SW_MON	0	2	17	0	2	12
Command1	0	4	7	0	3	7
Command2	0	2	22	0	1	22
ElectricalBrakeUnit	1	0	0	1	0	0
HYD1	0	0	5	0	0	0
HYD2	0	0	18	0	0	0
IsolationValve2	0	0	18	0	0	0
Monitor1	0	2	4	0	2	4
Monitor2	0	1	12	0	1	12

TABLE V
EXTRACT OF CONTRIBUTION OF ITEM TO WBS FFSs (LIMITED TO ORDER 3)

Furthermore, some components of the WBS are considered as similar; so they should have the same DAL. For instance the two valves used to transmit breaking order from the pilot pedal should have the same DAL.

Eventually, costs were derived from a study of [7] providing an evaluation of the evolution of DAL cost w.r.t. to baseline cost (for E DAL). For instance the baseline cost of the Accumulator is 100 arbitrary units, 105 for DAL D, 137 for DAL C, 157 for DAL B and 165 for DAL A.

Table VI provides the DAL allocation obtained with the DALCULATOR and the one obtained by manual analysis of the FFSs in the appendix Q of [2]. Two solutions from the appendix Q and three optimal solutions from the DALCULATOR are provided: one using only Option 1 rule, one using only Option 2 rule and the last one using both Option 1 and Option 2 rules. Note that the manual allocation has been limited to FFSs of order two and to a subset of items. The symbol _ in Table VI identifies the items for which the manual DAL allocation is not available.

A first observation concerning Table VI is that the DALCULATOR finds allocations that are consistent with the manual analysis. The discrepancy observed on CCF_SW_COM and CCF_SW_MON is explicitly addressed by an editor note in the appendix Q indicating that such an alternative is acceptable.

Since the manual analysis has been restricted to FFS of order two, the DAL constraints are not considered for items whose development errors are involved only in order three FFS. These components (*e.g.,* Alt_EmerMeterValve_L, HYD1, HYD2, IsolationValve2) can be found in the Table V. One may argue that higher order FFSs containing the development errors of these items also contain items for which more stringent DAL allocation constraints have been considered for lower order FFS. But this is not always the case, for instance the FFS {HYD1, HYD2, IsolationValve2} only contains items involved in FFSs of order three. Let us remind that highly safety critical systems are acceptable for very remote failure conditions, thus the order of the FFSs to analyze should be quite large. Therefore being able to take into account high order FFSs during the DAL allocation is paramount to ensure the allocation soundness. Due to combinational explosion, the number of high order FFSs may be quite large. Thus an automatic analysis may be really helpful to avoid time consuming and error-prone manual DAL allocation.

C. DAL allocation exploration

Our second use case is a medium size fixed wing UAV operated for visual inspection of long range infrastructures such

Item	Q appendix of [2]		DALCULATOR		
	Allocation 1	Allocation 2	Option 1	Option 2	Combined
Accumulator	_	_	D	C	C
Alt_EmerMeterValve_L	_	_	A	B	B
Alt_EmerMeterValve_R	_	_	A	B	B
CCF_HW_COM_MON	A	A	A	A	A
CCF_SW_COM	B	B	**C**	B	C
CCF_SW_MON	B	B	**A**	B	A
Command1	_	_	C	B	C
Command2	_	_	C	B	C
ElectricalBrakeUnit	_	_	A	A	A
HYD1	_	_	B	B	B
HYD2	_	_	B	B	B
InternalPower1	C	B	C	B	C
InternalPower2	C	B	C	B	C
IsolationValve1	_	_	C	C	C
IsolationValve2	_	_	C	C	C
Monitor1	_	_	A	B	A
Monitor2	_	_	A	B	A
PWR1	C	B	C	B	C
PWR2	C	B	C	B	C
PowerMonitor1	A	B	A	B	A
PowerMonitor2	_	_	A	B	A
Selection_Mgt	A	B	A	B	A
SelectorValve	_	_	B	C	C
ShutOff_Valve	_	_	B	D	D
Total cost	_	_	12356	13055	12260

TABLE VI
COMPARISON OF MINIMAL DAL ALLOCATION OF THE Q APPENDIX OF [2] AND THE DALCULATOR

as railways or pipelines. These infrastructures are assumed to be located in sparsely populated areas. Moreover, to handle in flight failures, the UAV is able to trigger a flight termination system (FTS) ensuring its containment within a predefined volume. So the main risks result from the following high-level scenarios:

- CAT_Ground: inability to ensure flight continuation and to perform a FTS
- HAZ_Ground: crash performed by the FTS ensuring both a kinetic energy reduction and a containment within the predefined volume
- MAJ_Ground: a landing on an appropriate site (either next to the take-off point or a within pre-defined area)

To obtain the flight authorization for such UAV operation, an applicant shall first perform a Specific Operational Risk Assessment (SORA, cf [10]). On one hand, the SORA defines which specific assurance and integrity level (SAIL) is proportionate to the operational risks. On the other hand, it defines operational safety objectives (OSO) with levels proportionate to the needed SAIL. Then, the applicant shall demonstrate that the operation is compliant with the OSOs .

Thus, an objective (called OSO#5) requests a demonstration of software integrity and assurance based on standard for high risk operations and enhanced containment system. Such a demonstration relies on the allocation of DAL. Guidance material suggests that DALs for SW/AEH may be derived from [8] The authors of [8] provide an adaptation of civil aviation standards to drone by defining how a DAL allocation commensurate with the operation level of risk. A proposition is to consider that the initial DAL corresponding to CATASTROPHIC (resp., HAZARDOUS or MAJOR) failure conditions is B (resp., C).

MBSA was used to support the safety assessment of this case study. The model contains both the functional and the physical architectures of the UAV. It includes also the behavior

Failure condition	FFS	
	order	number
CAT_Ground	1	0
	2	3685
HAZ_Ground	1	26
	2	142
MAJ_Ground	1	230
	2	7

TABLE VII
FFS REPARTITION FOR THE UAV SYSTEM

of safety functions (monitoring, reconfiguration, ...). Finally, it captures the hypotheses concerning the development errors of the system components and their local effects. The automatic analysis provided by the Altarica ecosystem derives the global effects of considered development errors. Note that we do not provide here the hypotheses and models of the detailed architecture of the UAV, but rather an abstraction to illustrate the process.

The analysis tools automatically identify the impact of single and combined development errors of functions or items. We used these tools to compute the FFSs containing item contributors (representing item development errors). Table VII provides the repartition of the computed FFS for the three failure conditions: CAT_Ground, HAZ_Ground, MAJ_Ground.

The model contains approximately more than 100 items, among which 56 contribute to the failure conditions. The Table VIII provides an extract of the number of FFSs an item's development errors contribute to.

The DALCULATOR enables the exploration of the possible DAL allocations and DAL reductions. As reminded in Section III-A, the DAL reduction rules require some independence between items or functions. Demonstrating the independence of two elements must be substantiated by the analysis of various sources of common development errors, among which the absence of dissimilarity. The following sections proposes several dissimilarity hypotheses for the UAV.

Hypothesis 1 *(Common cause) We consider here that the absence of dissimilarity is the only remaining common development errors to consider. The other common development errors (such as function allocation, external events) have been*

Item	CAT_Ground	HAZ_Ground		MAJ_Ground
	2	1	2	1
AilL airspeed computation SW	0	0	0	2
AilL control SW	192	0	0	6
AilL flight control selection SW	96	0	0	2
AilL servo status SW	192	0	0	4
Backup flight control HW	80	0	4	2
Backup flight control SW	1204	0	48	28
Backup internal geocaging SW	160	0	8	4
Backup monitoring SW	202	0	4	4
Battery monitoring	40	2	0	0
External geocaging HW	0	0	3	2
Primary internal geocaging SW	160	0	8	4

TABLE VIII
EXTRACT OF CONTRIBUTION OF ITEM TO UAV FFSS

Item	Dissimilarity hypothesis		
	Optimistic	Pessimistic	Custom
AilL HW	C	B	B
AilL airspeed computation SW	C	C	C
AilL control SW	C	B	B
AilL flight control selection SW	C	B	B
AilL servo status SW	C	B	B
AilR HW	C	B	B
AilR airspeed computation SW	C	C	C
AilR control SW	C	B	B
AilR flight control selection SW	C	B	B
AilR servo status SW	C	B	B
Altitude Acquisition HW	C	C	C
Attitude acquisition HW	C	B	C
Backup flight control HW	C	B	C
Backup flight control SW	C	B	C
Backup internal geocaging SW	C	B	C
External geocaging HW	C	C	C
Primary internal geocaging SW	C	B	C

TABLE IX
EXTRACT OF POSSIBLE DAL ALLOCATIONS ACCORDING TO DISSIMILARITY HYPOTHESES

taken into account in the safety assessment and thus during the computation of the FFSs.

Hypothesis 2 *(Optimistic physical dissimilarity) All software and hardware items are dissimilar. This hypothesis is optimistic since some items may rely on similar technologies.*

Example 9 *(Optimistic physical dissimilarity) The primary and backup flight control hardware are dissimilar (e.g, implementation using dissimilar processors).*

Hypothesis 3 *(Pessimistic physical dissimilarity) Two pieces of hardware or software are dissimilar if they are not of the same type. This hypothesis is pessimistic since some redundant items may be dissimilar.*

Example 10 *(Pessimistic physical dissimilarity) A pitot sensor is dissimilar from an altimeter but two pitot sensors are similar. Another example, the flight mode selector is dissimilar from the flight control system, but the replications of the flight mode selector on several processors are similar.*

Hypothesis 4 *(Custom physical dissimilarity) We consider that the hardware and software of the nominal/degraded flight control/monitoring, the positioning and the acquisition are dissimilar.*

Example 11 *(Custom physical dissimilarity) The processors used by the two nominal/degraded flight control/monitoring are dissimilar. The flight mode selectors allocated on the processors are dissimilar.*

Table IX provides some possible DAL allocations considering the above dissimilarity hypotheses.

As shown by Table IX, Hypothesis 2 enables to reduce all items to DAL C; no further reduction is possible since each of them belongs to a singleton FFS for HAZ or MAJ outcome. This hypothesis enables the designer to identify the maximal achievable DAL downgrade with full dissimilarity.

Conversely, the allocation obtained with Hypothesis 3 gives an idea of the DAL levels when minimum dissimilarity is considered. Interestingly we can observe that some specific items are not impacted by the choice of dissimilarity hypothesis. For instance the external geocaging DAL reduction is always possible since it does not belong to CAT FFS of order one and its HW is always considered as independent from UAV HW in the hypotheses (not the same type than Primary internal geocaging HW).

Some items are impacted by the choice of the dissimilarity hypothesis, for instance the DAL of the SW used to monitor the servomotors of aileron cannot be downgraded if these SW are not dissimilar. The situation occurs when two similar items belongs to a same FFS of order 2.

In some specific cases, the dissimilarity may lead to allocate heterogeneous DAL on SW hosted on the same execution unit. For instance the airspeed computation SW hosted on aileron are not involved in a CAT FFS of order two, so even if all these SW are similar, the reduction is still possible since a design fault affecting all these SW does not directly lead to a CAT failure condition. The exploration of specific dissimilarities can help the designer to identify what are the relevant dissimilarities in order to lower the DAL of complex items. For instance the consideration of dissimilar flight control/monitoring has been exploited by the DALCULATOR .

A last remark in this specific use case is that the MAJ_Ground failure condition is driving a lot the DAL allocation. This observation illustrates the need to consider event moderate severity failure conditions for such systems.

D. Automatic DAL Allocation scalability

Our last use case is a fine grain analysis of the communication system of a military transport aircraft. The communication system is composed of three subsystems working on non interfering frequency ranges (HF, VHF, UHF).

An MBSA model of this system has been developed by Christophe Frazza and formalizes the dependency of the communication system's components w.r.t. the power supply system. The studied failure condition is the inability to provide a reliable communication *i.e.*, the total loss of the communication or misleading information. The severity of such failure condition is considered as MAJOR. Note that the combination of the navigation and communication system loss is CATASTROPHIC. As the communication system is part of a large aircraft, the initial DAL corresponding to MAJOR failure conditions is C.

As shown by Table X, a large amount of FFSs are contributing to the failure condition. A manual analysis may be achievable for FFSs of order two but seems hardly possible when considering FFSs of order three and four.

The DALCULATOR has been used to generate a DAL allocation considering FFSs of order three and four. Since more than 60 items are contributing to the failure condition, Table XI only provides an extract of the generated DAL

Failure condition	FFS	
	order	number
Loss of communication	1	6
	2	45
	3	569
	4	16730

TABLE X
FFS REPARTITION FOR THE BREAKING SYSTEM

Item	DAL Allocation considering order	
	3	4
HF_Antenne.HW_data	E	C
HF_Coupler_1.HW_data	_	E
HF_Coupler_1.Stub_inter	_	E
HF_Coupler_2.HW_data	E	E
HF_Coupler_2.Stub_inter	E	E
Execution time	2032ms	5060ms

TABLE XI
EXTRACT OF DAL ALLOCATION PERFORMED BY THE DALCULATOR CONSIDERING FFSS OF ORDER THREE OR FOUR OF THE COMMUNICATION SYSTEM

allocations. For each allocation the execution time has been measured on a laptop with a Intel(R) Core(TM) i7 CPU 2.90GHz. The obtained execution times enable to perform a full DAL allocation within seconds on a large number of FFSs.

The proposed extract illustrates that considering high order FFS may impact the DAL allocation. Indeed, as illustrated by Table XII, the contribution of HF_Coupler_1.HW_data appears only in FFS of order four. These new contributions may also impact the DAL of components contributing to lower order FFSs. For instance considering the FFS of order four increases the DAL allocated to the HF_Antenne.HW_data item from E to C.

V. RELATED WORKS

The assurance level allocation problem has been extensively addressed by the literature in the automotive field. The survey [11] provides a thorough categorisation of the methods using to tackle the allocation of Automotive Safety Integrity Level (ASIL) introduced in the ISO26262 automotive standard. As identified by [11], the considered approaches are either relying on constraint programming or meta-heuristics.

Among the meta-heuristics approaches, many works such as [12], [13] relies on genetic algorithms or Tabu search [14]. Nevertheless, these works are not considering the possible sources of non-independence that may prevent some DAL allocations. The capability of encoding formally these non-independence sources is a strength of the method proposed in this paper. Moreover, unlike pseudo-Boolean based optimisation problems, the meta-heuristics based methods do not provide any optimality guarantees.

Item	FC_Comm			
	1	2	3	4
HF_Antenne.HW_data	0	2	66	628
HF_Coupler_1.HW_data	0	0	0	1728
HF_Coupler_1.Stub_inter	0	0	0	864
HF_Coupler_2.HW_data	0	4	24	1668
HF_Coupler_2.Stub_inter	0	2	12	834

TABLE XII
EXTRACT OF CONTRIBUTION OF ITEM TO COMMUNCITATION SYSTEM FFSS

Among the constraint programming based approaches, the authors of [15] propose an allocation method based on a formalisation of the allocation rules using Satisfiability Modulo Theory (SMT). The authors of [16] propose a more comprehensive method to tackle both the ASIL and reliability allocation problems for time sensitive networks using again SMT solvers. Eventually, the works of [3] provide a DAL allocation based on pseudo-Boolean solver, approach on which the works presented in the paper is based on. The modelling and solving method of the DAL allocation problem belongs to this family of constraint based approaches. Nevertheless, it differs from [15], [16] since: first we address the DAL allocation problem that does not follow the same rules as the ASIL allocation problem. Second [16] restricts the scope of the problem to specific systems (here time-sensitive networks) and [15] encodes the fault tree as a graph-like data-structure. Our method relies on the FFSs provided by the safety analysis therefore the method can theoretically be applied to any kind of systems. Eventually, these methods mainly involve integer linear arithmetic (ILA) while our method is based on pseudo-Boolean logic. As identified in [17], pseudo-Boolean solvers sometimes outperform even commercial ILA solvers by orders of magnitude for complex problems such as arithmetic circuit verification.

VI. CONCLUSION

a) Summary: In this paper, we propose a formalization of the DAL allocation problem compliant with the aeronautic practices. We take into account optimization criteria based on the DAL cost of each function and the number of functions required being independent. Furthermore, we show how to introduce user constraints to focus the exploration on more interesting allocations.

This formalization enables to solve the problem using very efficient constraint solvers and to provide optimality guarantees. A fully automated tool implementing the proposed formalization provides DAL allocation and a set of function independence requirements from the safety analyses and user constraints.

Experimentation has been conducted on several systems and highlights the benefits of the constraints-based allocation approach compared to the heuristic-based approach. This experimentation also demonstrates that the scalability of the approach enables the usage of our allocation tool on industrial systems.

b) Limitations: The proposed method aims at providing a flexible formal framework to tackle assurance level allocation in various applicative domains. Nevertheless, even if the allocation rules in various domains (*e.g.,* DAL for aeronautics and ASIL for automotive) are quite similar and mainly inspired by the SIL allocation provided by [18], it is not yet possible to configure the DALCULATOR to perform a specific assurance level allocation. This limitation is due to the lack of flexibility

of the formalization itself that assumes we rely on the allocation rules of [1]. Concerning the cost-based optimization, the user can only consider scalar costs to encode the development costs. This limitation may be quite restrictive since the notion of development costs can encompass several kinds of costs *e.g,* development time, financial costs or complexity. A last important limitation is the inability of the DALCULATOR to perform simultaneously a IDAL and FDAL allocation.

c) Future works: To enhance the flexibility of the formal framework, a future work could be to enable the user to specify the allocation rules of the applicable standards. The DALCULATOR could then be used to allocated various assurance levels in addition to the DAL. This extension could also be used to consider additional constraints encoding the IDAL and FDAL allocation rules. Concerning the cost, a simple enhancement would be to enable lexicographic optimization on user defined costs or Pareto-front exploration.

ACKNOWLEDGEMENTS

ONERA work takes place in the PHYDIAS2 project about means of compliance for safe unmanned aerial systems. PHYDIAS2 is funded by DGAC in the framework of the plans "France Relance" and "NextGeneration EU". Laura Chambert's work took place at DGA TA in the framework of a "Contrat Armée Jeunesse" .

REFERENCES

[1] SAE, "Aerospace Recommended Practices 4754a - ed-79a - Development of Civil Aircraft and Systems," 2010.
[2] ——, "Aerospace Recommended Practices 4761- ed-135 - guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment," 1996.
[3] P. Bieber, R. Delmas, and C. Seguin, "Dalculus–theory and tool for development assurance level allocation," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2011, pp. 43–56.
[4] ISO, "ISO-26262 -Road vehicles – Functional safety," 2010.
[5] RTCA, Inc / EUROCAE, "DO-178 ED-12C - Software Considerations in Airborne Systems and Equipment Certification," 2011.
[6] ——, "DO-254 / ED-80 Design Assurance Guidance for Airborne Electronic Hardware," 2005.
[7] C. Baron and V. Louis, "Towards a continuous certification of safety-critical avionics software," *Computers in Industry*, vol. 125, p. 103382, 2021.
[8] JARUS, "Amc rpas.1309 issue 2: Safety assessment of remotely piloted aircraft systems."
[9] A. Arnold, G. Point, A. Griffault, and A. Rauzy, "The altarica formalism for describing concurrent systems," *Fundamanta Informaticae*, vol. 40, no. 2-3, pp. 109–124, 1999.
[10] EASA, "Easy access rules for unmanned aircraft systems," 2021.
[11] Y. Gheraibia, S. Kabir, K. Djafri, and H. Krimou, "An overview of the approaches for automotive safety integrity levels allocation," *Journal of failure analysis and prevention*, vol. 18, pp. 707–720, 2018.
[12] Z. Lu, L. Zhong, S. Haijing, and Z. Jia, "An optimization method for development assurance level assignment of airborne system," *Systems Engineering & Electronics*, vol. 44, no. 8, 2022.
[13] X. Li, Z. Lu, and J. Wang, "An optimization approach for dal assignments," *Aircraft Engineering and Aerospace Technology*, vol. 90, no. 2, pp. 328–335, 2018.
[14] Y. Wei, Y. Le, G. Xie, and L. Zhang, "Development cost optimization for multi-functional mixed-criticality embedded systems," *IEEE Access*, vol. 7, pp. 88 949–88 959, 2019.
[15] M. Safar, "Asil decomposition using smt," in *2017 Forum on Specification and Design Languages (FDL)*. IEEE, 2017, pp. 1–6.
[16] Y. Zhou, S. Samii, P. Eles, and Z. Peng, "Asil-decomposition based routing and scheduling in safety-critical time-sensitive networking," in *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2021, pp. 184–195.
[17] V. Liew, P. Beame, J. Devriendt, J. Elffers, and J. Nordström, "Verifying properties of bit-vector multiplication using cutting planes reasoning," in *# PLACEHOLDER_PARENT_METADATA_VALUE#*, vol. 1. TU Wien Academic Press, 2020, pp. 194–204.
[18] I. E. Commission *et al.*, "Iec 61508: Functional safety of electrical," 1999.