Evaluation de l’approche STPA dans le cadre de l’analyse SOTIF d’un système de gestion de contrôle aérien

Evaluation of the STPA approach as part of the SOTIF analysis of an air traffic control management system

HANNI Camillia
THALES Land & Air Systems/SIAM
Rungis
camillia.hanni@thalesgroup.com

MACHROUH Joseph
THALES Land & Air Systems/SIAM
Rungis
joseph.machrouh@thalesgroup.com

KEHREN Christophe
THALES Land & Air Systems/SIAM
Rungis
christophe.kehren@thalesgroup.com

CONSTANT Olivier
THALES Research & Technology
Palaiseau
olivier.constant@thalesgroup.com

I. OBJECTIF

L'objectif de cette étude et de montrer l'intérêt de l'approche STPA (System Theoretic Process Analysis) [2] [3] [4] et ses capacités à améliorer nos analyses de sécurité des systèmes ATM (Air Trafic Management). STPA offre une perspective holistique qui étend les approches standards centrées uniquement sur les défaillances matérielles ou logicielles et se distingue par son approche proactive, plaçant l'accent sur la compréhension des interactions entre les éléments du système plutôt que sur la simple détection de défaillances.

Dans le contexte ATM, cette méthodologie permet d'explorer les dimensions humaines, organisationnelles et technologiques, offrant ainsi une vision complète des facteurs contributifs à la sécurité qui ne sont pas toujours causés par des défaillances, mais potentiellement par des interactions complexes entre les systèmes, les opérateurs et l'environnement.

Une contribution majeure de cette approche réside donc dans sa capacité à anticiper les défaillances potentielles et à proposer des mesures d'atténuation avant qu'un incident ne se produise.

II. METHODOLOGIE

A. Methdologie

1) Introduction STPA

La méthode STPA (System-Theoretic Process Analysis) est une technique basée sur le modèle théorique d'accident et processus STAMP (System Theoretic Accident Model and Process) [4]. Cette dernière part du principe que les accidents sont dus à un manque d'efficacité dans l'application des contraintes de safety sur le comportement du système afin d'éviter les états ou conditions dangereux.

La méthode STPA [3] est une méthode descendante qui utilise un modèle du système qui consiste en un diagramme de contrôle fonctionnel au lieu d'un diagramme des composants physiques utilisé par les méthodes standards d'analyse safety. Selon Ishimatsu et al. [1] ISO/PAS 21448 (2019).

STPA prend en compte les interactions entre les composants du système et considère le système évalué comme un tout. La STPA requiert un diagramme de structure de contrôle pour l'analyse des dangers, qui consiste en des composants d'un système et de leurs voies de contrôle et de rétroaction (voir Figure 1: Méthode STPA (source).

2) Plugin STPA sur CAPELLA

Thales a développé un plugin Capella qui permet l'analyse STPA [6]. Ce plugin fournit un outil d'aide à la STPA basé modèle. Il peut être utilisé pour des analyses STPA autonomes ou en combinaison avec la modélisation classique de Capella. La fonctionnalité liée au STPA est classiquement disponible en tant que "point de vue Capella". En tant que telle, elle doit être activée une fois pour chaque projet Capella concerné.

3) Cas d'étude et target

Dans le cadre de notre analyse des risques STPA, nous avons sélectionné un cas d'étude pertinent afin d'explorer une perspective élargie dans l'analyse des risques, mettant ainsi en valeur la méthode STPA ainsi qu'un outil permettant sa réalisation (le plug-in dédié). Notre choix s'est porté sur un composant de détection de collision intégré dans un système global de gestion du contrôle aérien. Ce composant se distingue par son caractère à la fois « autonome », en traitant des plans de vol

Figure 1: Méthode STPA (source).

en entrée, calculant les conflits grâce à son algorithme et en transmettant les alertes de conflits entre paires d'avions aux contrôleurs aériens, et « online », grâce à la possibilité d'interactions entre les humains et l'interface homme-machine (IHM). Ainsi, ce composant détectera des potentiels conflits en se basant sur la trajectoire prévue des plans de vol d'ici un horizon de 30 minutes. C'est un outil d'aide aux contrôleurs aériens avec une vision proactive des conflits, afin de gérer efficacement ces derniers et anticiper leur charge de travail.

B. Modélisation du cas d'étude :

Dans cette partie nous décrirons les étapes de la modélisation de notre cas d'étude dans le cadre de notre analyse STPA.

1) Le périmètre de l'analyse STPA :

La première étape de STPA vise à cadrer l'analyse en établissant un périmètre d'étude. Cette étape est itérative et ne se réalise pas en une seule fois. Lors de cette première étape, il s'agit d'inscrire toutes les pertes c'est à dire la perte de quoi que ce soit qui a de la valeur aux yeux des parties prenantes de notre système et qui seraient inacceptables. Il est également possible d'inclure des pertes liées à des facteurs environnementaux. Pour identifier les pertes, il peut être utile de passer par une identification préalable des enjeux des parties prenantes. Cette étape intermédiaire permet de comprendre les intérêts des différentes parties prenantes de notre système, ce qui facilite ensuite l'identification des pertes potentielles qui pourraient compromettre ces enjeux.

Dans notre cas, deux pertes (de haut niveau) sont identifiées :

- Pertes de vies humaines.
- Perte de réputation pour l'entreprise.

Comme mentionné précédemment, il est à noter que les pertes peuvent être de tout ordre, ainsi STPA permet de définir une perte liée aux aspects environnementaux. À titre d'illustration, on peut définir la perte suivante :

- Perte de performance environnementale (du fait de la possible résolution automatique de conflits en proposant une solution optimale mais qui sera hors scope de notre analyse STPA).

Les dangers sont de niveau système [4] : Un état du système ou un ensemble de conditions qui, associé à un ensemble particulier de conditions environnementales les plus défavorables, entraînera une perte.
Les dangers identifiés dans notre analyse STPA sont les suivants (certains de ces dangers sont raffinés avec des sous-dangers):

- Le système ne détecte pas de conflits.
- Le système détecte de faux conflits.
- Le système n'affiche pas les conflits aux contrôleurs via l'IHM.
- Le système ne permet pas le transfert des requêtes (commandes des humains vers le composant).
- Le système ne permet pas la prise en compte des mises à jour, création et suppression des plans de vol.
- Le système (composante technique et humaines) ne permet pas de résoudre le conflit.
- Le système fait augmenter la charge de travail du contrôleur.

Les dangers sont ensuite déclinés en contraintes de niveau système (CNS). Cette étape est grandement facilitée par le plugin STPA, qui permet, sous forme de tableau, de réaliser une traçabilité entre les enjeux, parties prenantes, pertes et dangers.

2) La modélisation de la structure de contrôle :

a) La modélisation : les composants, interactions et responsabilités STPA :

L'une des étapes clés de STPA, mais aussi la plus délicate, concerne la modélisation de la structure de contrôle hiérarchique, qui doit demeurer cohérente avec la délimitation de notre système d'intérêt et des dangers identifiés à l'étape 1. En effet, c'est cette modélisation qui permettra d'identifier les *« unsafe control action »* (UCA) (pouvant conduire à un danger et donc à une ou des pertes), ou celles non ou mal exécutées. Un aspect souvent discuté lors de la réalisation d'une analyse STPA concerne l'identification et la justification du niveau d'abstraction approprié pour la structure de contrôle hiérarchique par rapport au système étudié, ainsi que l'établissement d'un critère d'arrêt pour l'analyse.

Pour notre étude STPA, nous avons identifié deux niveaux d'abstraction distincts :

- Niveau d'abstraction 1 : « boîte blanche » du composant test.
- Niveau d'abstraction 2 : « boîte noire » axé sur les interactions externes du composant test.

De plus, deux modèles Capella ont été utilisés en support :

- Un modèle 1 avec un niveau d'abstraction logique, représentant le système de gestion du contrôle aérien complet intégrant donc notre composant test (détection de conflits).
- Un modèle 2 se focalisant sur les interfaces des composants logiques du composant de détection de conflits.

Et enfin, de la documentation était également à notre disposition.

L'utilisation de l'approche MBSE (Model-Based Systems Engineering) dans le cadre de cette étude STPA a pour but d'exploiter les modèles comme principal moyen de collecte d'informations sur le fonctionnement du système. Cette approche diffère d'une méthode exclusivement basée sur la documentation, et elle permet également de maintenir la cohérence et l'intégration de notre analyse STPA avec le modèle Capella déjà existant, grâce au plug-in présenté dans la section 2)
Le niveau d'abstraction retenu pour la suite de notre analyse est le niveau 1, qui apparaît être le niveau d'abstraction nécessaire aux objectifs de l'analyse STPA compte tenu du ratio coût/bénéfice apporté par l'effort de modélisation. Ce niveau d'abstraction reste à un niveau logique des composants et inclut la représentation des bases de données échangeant avec ces derniers. Cela permet une analyse interne du fonctionnement avec un coût de modélisation moindre grâce au support du modèle Capella se focalisant sur les interfaces.

L'objectif est de capitaliser pleinement les informations contenues dans les modèles logiques et centrés sur les interfaces pour extraire des informations afin de réaliser la structure de contrôle hiérarchique STPA pour notre niveau d'abstraction choisi. Le modèle 1 issu de Capella est à un niveau d'abstraction logique, les échanges entre composants sont des échanges dits fonctionnels et agnostiques autant que possible de toutes solutions d'architectures physiques. La première étape dans l'établissement d'une structure de contrôle conjointe à notre modèle consiste à identifier les composants permettant la détection de conflit dans le modèle Capella du système global de gestion du contrôle aérien. Cette vue d'ensemble (architecture logique)

permet d'identifier les composants qui seront présents dans la modélisation de la structure de contrôle hiérarchique parmi tous les autres. Ces composants doivent réaliser la capacité de détection de conflits. Ils sont tous reliés par les « échanges de composants » qui ont pour objectif de représenter les connexions entre les composants logiques du modèle permettant la circulation d'échanges fonctionnels. Ces « échanges de composant » sont explicitement nommés selon un formalisme (IF_, OU_, P2P_) qui permet de savoir ce qui est échangé (message d'erreurs, plans de vol, conflits…) et de développer une compréhension globale de notre système d'intérêt (humain, IHM et composants effectuant la fonction de détection de conflit), notamment à travers ces interactions et les types d'échanges impliqués.

Le modèle Capella nous fournit les composants logiques suivants qui seront modélisés dans notre structure de contrôle en tant que contrôleur STPA (voir ci-dessous)

Capacité : Détecter des conflits
IHM
Composant_1
Composant_2
Composant_3
« Composant_4 » (utilisateurs opérationnels)

Tableau 1: Composants identifiés dans le modèle Capella

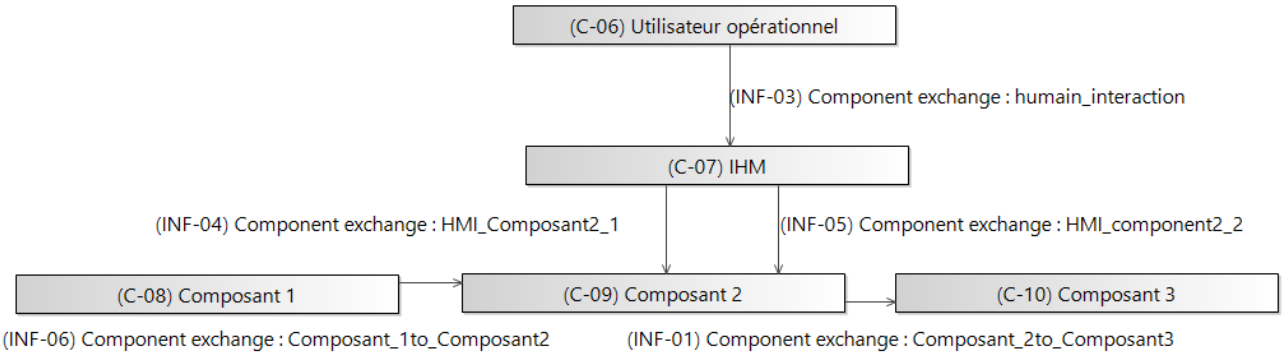

Figure 2: Première ébauche de la structure de contrôle avec les composants identifiés (contrôleur) ainsi que les échanges avec les autres composants.

Ensuite, nous procédons à l'analyse des fonctions logiques du système global de gestion du contrôle aérien pour nous focaliser sur la fonction logique principale de notre système de détection de conflits, à savoir la détection de conflits. Cette dernière sera la fonction logique mère.

Le découpage de cette dernière en plusieurs sous-fonctions logiques filles nous permet d'établir, via le navigateur sémantique de Capella, son allocation à un composant de notre structure de contrôle, comme décrit dans le Tableau 1. Ce découpage en sous-fonctions filles permet d'établir des responsabilités STPA sur la base de ces allocations de fonctions filles. Nous avons donc alloué à chacun des contrôleurs (STPA) une responsabilité liée aux fonctions filles. Il est cependant essentiel de s'assurer que les responsabilités assignées garantissent effectivement le respect (ou du moins la tentative) des contraintes systèmes déclinées des dangers identifiés lors de l'étape 1 de STPA afin de ne modéliser uniquement que ce qui est nécessaire à notre analyse STPA. Ce filtrage est nécessaire dans notre cas, et nous permet d'éliminer des responsabilités, et donc des fonctions logiques, qui sont alors hors du scope de l'analyse (par exemple une fonction logique de résolution de conflit qui aurait conduit à une affectation d'une responsabilité à un composant non humain).

En revanche, l'exploration de cette fonction mère uniquement ne permet pas d'établir la responsabilité d'un composant, celle-ci ayant été identifiée en analysant une autre fonction mère liée à la distribution des plans de vol. Aussi, on ne retrouve pas certaines sous-fonctions logiques filles directement rattachées à la fonction mère de détection de conflit pour l'affectation de certaines responsabilités (l'envoi des messages d'erreurs par exemple ou l'envoi des plans de vol en entrée) mais qui en revanche sont identifiables grâce à l'architecture logique du système en étudiant les échanges de composants et les fonctions logiques allouées à ces derniers.

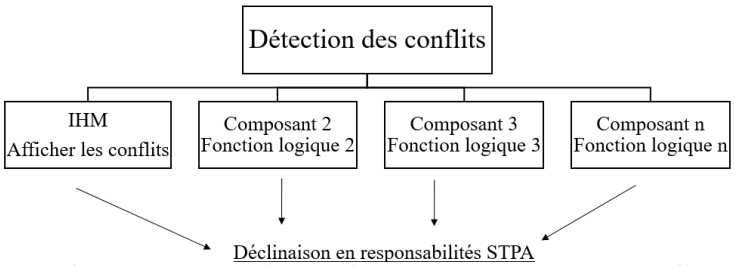

Figure 3 : Découpage de la fonction mère principale en sous fonctions filles pour en extraire les responsabilités STPA

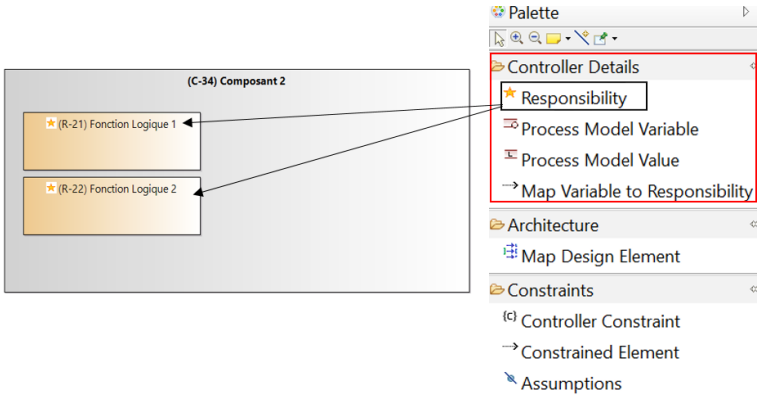

Figure 4: Création des responsabilités avec le plug in sur un composant

	Name	System-Level Constrai...	Control Actions	Feedback	Information	Process Model
(R-21)	Fonction Logique 1	[SC-01]	[]	[]	[]	[]
(R-22)	Fonction Logique 2	[SC-04]	[]	[]	[]	[]

Figure 5: Lien et traçabilité avec l'exercice d'une responsabilité pour l'application des CNS établie à l'étape 1.

b) Identification des actions de contrôle feedbacks et finalisation de la structure de contrôle :

La partie concernant la modélisation des composants, des interactions et des responsabilités STPA est complétée avec la modélisation des actions de contrôle et des feedbacks sur la base des responsabilités établies plus haut, comme le suggère le manuel. Le support premier est l'exploitation des échanges fonctionnels (c'est un échange unidirectionnel d'informations ou de matières entre deux fonctions logiques) entre les fonctions logiques identifiées en lien avec la fonction mère de détection de conflits. Dans notre approche basée sur le modèle, le but était de tester l'établissement d'un possible transfert des chaînes fonctionnelles (séquence de fonctions qui travaillent ensemble pour accomplir une tâche spécifique) en « chemin de contrôle » en établissant un pont entre des échanges fonctionnels et une action de contrôle STPA, ce qui a pu être réalisé sur certaines chaînes fonctionnelles mais pas toutes.

Malheureusement, il n'a donc pas été possible d'établir ce lien systématique et universel entre les échanges fonctionnels entre deux fonctions logiques du modèle et une action de contrôle.

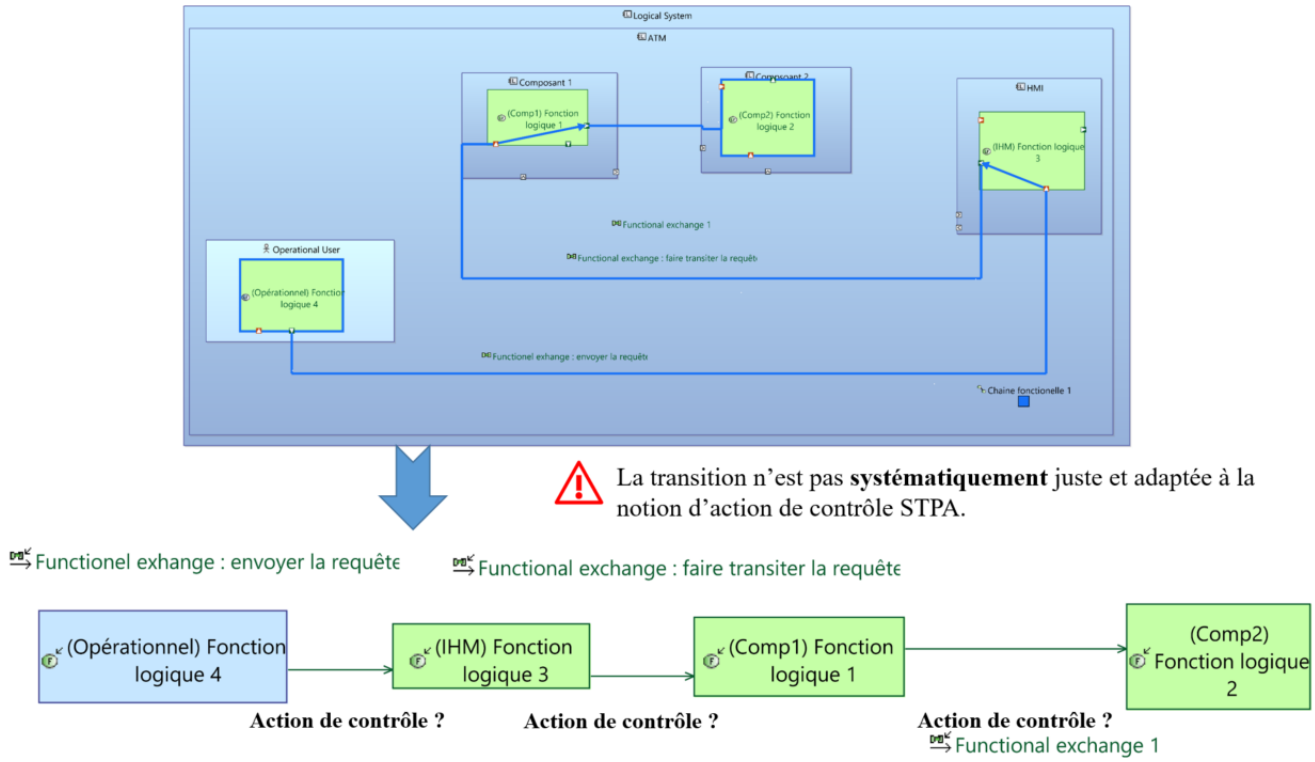

Figure 6: Processus d'exploration des chaines fonctionnelles pour un potentiel passage en modélisation STPA.

Nous avons donc complété l'identification des actions de contrôle en utilisant en parallèle la documentation associée. Les process models, feedbacks et informations sont identifiés d'une part grâce au modèle centré sur les interfaces et le système, ce qui permet d'avoir à disposition toutes les variables et types de données échangés entre les composants et nos bases de données, et d'autre part avec des échanges avec un ingénieur système. Cette étape de réflexion sur les process models et donc l'identification des feedbacks/informations nécessaires sont tous reliés à l'exercice d'une responsabilité allouée à un contrôleur (STPA).

Concernant la réalisation de la structure de contrôle avec le plug-in, ce dernier permet d'avoir une vue se focalisant sur les contrôleurs de manière individuelle tout en représentant leurs interactions avec les autres composants (du fait des actions de contrôle). Cette fenêtre a pour but de compléter la modélisation de la structure de contrôle et ainsi renseigner les process models et l'identification des feedbacks. L'utilisation du modèle Capella nous a permis de bien comprendre le fonctionnement du système pour la réalisation de la structure de contrôle, cependant nous avons eu besoin d'autres informations non présentes dans les modèles. C'est avec la documentation que nous avons complété la modélisation. Une fois la structure de contrôle réalisée, l'analyste humain, afin d'identifier des scénarios de pertes, doit énoncer des « unsafe control actions » qui violeraient une CNS établie dans l'étape 1 de l'analyse, pour ensuite décliner une contrainte sur le contrôleur (STPA).

Voici un exemple d'UCA que nous avons identifié dans notre analyse :

Tableau 1: Exemples d'UCA

UCA :	**Contrainte de contrôleur (STPA) :**
UCA_1 : Publier un élément sur un espace d'une base de données **alors que ce dernier est déjà occupée.**	Le composant ne doit jamais envoyer l'écriture sur un espace de la base de données si cet espace est déjà occupé.
UCA_2 : Le composant 3 ne publie pas un statut d'éligibilité **pour un plan de vol alors que celui-ci** respecte les conditions requises pour l'être.	Le contrôleur (STPA) doit publier le statut éligible du plan de vol dans la base de données si ce plan de vol est éligible à passer à la détection de conflit.
UCA_3 : Le superviseur opérationnel **envoie une requête pour activer plusieurs volumes alors que le composant** 2 **ou/et** 3 **sont défaillants**.	Le superviseur opérationnel ne doit pas avoir la possibilité d'activer ou désactiver des volumes alors que ces composants ne fonctionnent pas.
UCA_4 : Le contrôleur ne fait pas de résolution manuelle d'un conflit alors que ces avions ne respectent pas les distances de sécurité.	Le contrôleur doit selon sa charge de travail actuelle, résoudre un conflit qui apparait et prioriser selon la criticité du conflit.

UCA_5 : Le superviseur opérationnel n'envoie pas la requête d'activation d'un volume de détection alors que le contexte opérationnel le requiert	Le superviseur opérationnel doit bien paramétrer les activations de volumes pour le contexte opérationnel à surveiller.

Les UCA 1 et 2 sont liées à une action de contrôle qui décrit le fonctionnement technique du système tandis que les UCA 3, 4 et 5 sont liées aux interactions possibles des humains avec l'IHM.

Il est également essentiel de noter que certaines contraintes de contrôleur, si examinées individuellement, peuvent ne pas suffire à garantir la sécurité de notre système par une simple application isolée. En effet, c'est la combinaison de toutes ces contraintes de contrôleur (STPA) qui doit être prise en compte. Le comportement du contrôleur doit être conforme à l'ensemble des contraintes. Pour illustrer, supposons qu'un plan de vol satisfasse toutes les conditions requises pour être publié selon la contrainte 2, mais que tous les emplacements dans la base de données où ce dernier doit être publié soient déjà occupés. Dans ce cas, le plan de vol ne devrait pas être publié en vertu de la contrainte 1, même si la contrainte 2 est respectée. Ainsi, c'est bel et bien le "ET" de toutes les contraintes qui doit être pris en compte pour garantir un comportement sûr du système.

Aussi, c'est pour cette raison que la modélisation de la structure de contrôle et de son niveau d'abstraction est essentielle. En effet, si une action de contrôle n'est pas représentée, il devient impossible d'identifier comment cette action peut mener à UCA, violant ainsi les contraintes système. C'est pourquoi nous avons choisi d'enrichir notre structure de contrôle avec le modèle des interfaces, afin de représenter les différents éléments de stockage.

En réalisant une analyse STPA, on peut prendre en compte de très (voire trop) nombreux contextes. Le guide STPA [4] donne des exemples de contextes potentiellement non sûrs à prendre en compte. La quantification formelle du nombre d'UCA possible n'a pas été établie, ce nombre dépendant de l'analyste humain effectuant l'étude STPA. Par conséquent, il n'est pas possible de garantir une couverture exhaustive de tous les contextes potentiels dans lesquels cette action de contrôle pourrait être non sûre. Cependant, il est envisageable d'établir une checklist des différents contextes à prendre en compte dans l'étude, puis de les examiner tout en justifiant tant leurs traitements que leurs non-traitements. On peut traiter une UCA partiellement, en déclinant une contrainte sur un contrôleur ou en allant plus loin en explorant l'UCA pour identifier un scénario de perte.

Tableau 2: Contexte à prendre en compte pour décliner une UCA

Contexte à prendre en compte pour décliner des UCA :	**Exemple d'UCA :**
Dans l'environnement nominal/ dégradé	La fonction 1 n'est pas transmise alors que [contexte nominal de fonctionnement]
Cas d'indisponibilité d'un composant.	La fonction 1 est transmise alors que le composant n°1 est [défaillant/saturé/en train de réaliser une autre tâche]
Une action réalisée précédemment	La fonction 1 est transmise alors que [l'action de contrôle n° 2 n'as pas été effective/en cours/pas envoyée]
La valeur d'un paramètre d'un ***process model***	La fonction 1 est envoyée alors que [état **réel** d'un *process model*]

Nous nous sommes intéressés aux UCA qui nous semblaient les plus pertinentes pour notre objectif, en identifiant les scénarios de pertes liés à la réalisation de ces UCA.

3) Lien entre l'approche SOTIF et STPA

C'est à ce stade que la réflexion sur l'aspect SOTIF (Safety Of The Intended Functionality) [4] (ISO/PAS 21448) peut être initiée. C'est en réalisant l'analyse STPA que nous allons réaliser le processus des activités SOTIF demandées dans ISO PAS. En partant du principe de base selon lequel chaque action de contrôle implémentée dans le système peut potentiellement être non sûre, même si elle permet à un contrôleur d'exercer une responsabilité STPA.

Ici, les Fonctionnalités attendues sont, dans notre cas d'étude, les actions de contrôle STPA. Ces dernières doivent être fournies à un instant donné pour répondre à leurs responsabilités. L'idée est donc d'identifier le contexte et les scénarios dans lesquels l'action de contrôle se retrouve à être non sûres et d'y apporter les mesures nécessaires. Réaliser une étude STPA permettra de se concentrer sur les zones 1, 2 et 3 (voir Figure 7). L'objectif est de maximiser (ou de maintenir) les scénarios sûrs de la fonctionnalité et connus (correspondant à la zone n°1), tout en réduisant au minimum les scénarios sûrs et inconnus ainsi que les scénarios non sûrs et inconnus (correspondant respectivement aux zones 2 et 3).

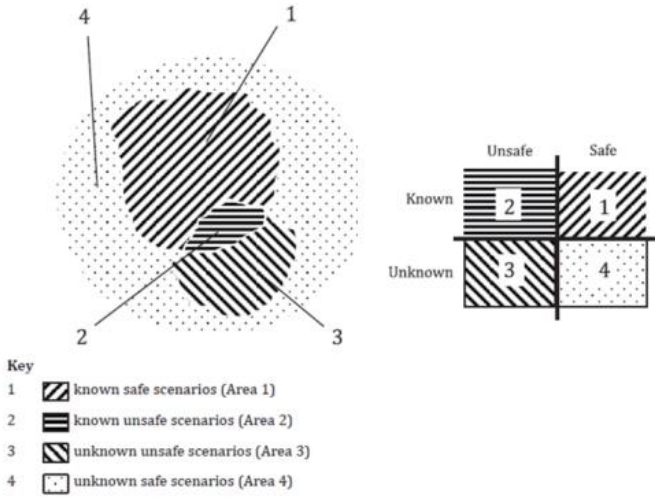

Figure 7 : Visualisation des aires connues/inconnues et sûres/non sûres (source [6])

On réduit les scénarios non sûrs et inconnus avec STPA (zone 3) ci-dessus en identifiant UCA dans un contexte particulier pour ensuite décliner un scénario de perte. Ensuite, pour chacun des scénarios de pertes identifiés, celui-ci se retrouve dans la zone n°2 : nous connaissons nos scénarios non sûrs liés à une action de contrôle (fonctionnalité). Enfin, nous minimisons l'aire n°2 en y apportant certaines contre-mesures de type : limitation ou amélioration des fonctionnalités lorsque ce contexte se produit. Ce processus fait passer le scénario identifié comme étant non sûr dans la zone 1, ce qui rend le système plus robuste. En revanche, STPA ne permet aucune quantification d'occurrence de scénarios de pertes (pas à l'état actuel), le passage de la zone 2 à la zone 1 doit être prouvé par des procédures de vérification.
Pour illustrer ce cheminement, nous allons prendre un exemple tiré de notre analyse, en nous appuyant sur l'Annexe A et le diagramme des activités SOTIF à réaliser présent dans la norme ISO/PAS 21448 (source [4])

Tableau 3: Application d'activités SOTIF supportées par STPA

Positionnement le digramme d'activité d'analyse SOTIF	**Cas d 'étude :**	**Notion STPA à utiliser :**	**Zone de SOTIF**
Spécification de la fonction. ***Clause 5***	Action de contrôle étudiée (de l'humain ver l'IHM): Activation et désactivation des volumes de détection ou d'inhibition pour effectuer la détection de conflits ou limiter le calcul de conflits dans certains volumes.	Action de contrôle STPA pour répondre à une responsabilité qui répond aux spécifications de CNS.	N/A
SOTIF : Identification du danger et évaluation du risque. ***Clause 6***	Contexte : l'un des composants 3 ou 4 ne fonctionne pas. Potentiel danger : le système ne peut pas résoudre un conflit car l'activation des volumes dans ce cas peut donner l'impression que ces volumes sont pris en charge par la détection automatique de conflits, alors que ce n'est pas le cas.	Identification des UCA liées à cette fonction (action de contrôle) qui enfreignent une CNS conçue pour prévenir un danger selon STPA.	Réduction de l'inconnu en zone n°3 Placement au niveau de la zone n°2
Risque de *préjudice* acceptable ? ***Clause 6***	Chaque UCA est liée à un danger relié à une perte STPA (un fait non acceptable par une partie prenante) pouvant causer des dommages à l'humain.	STPA ne permet pas de quantifier la notion de risque car elle est intrinsèquement liée à des probabilités, ce que STPA ne fournit pas.	Réduction de l'inconnu en zone n°3 Placement au niveau de la zone n°2

Identification et évaluation des évènements déclencheurs. ***Clause 7***	Evénement déclencheur : Au niveau de l'IHM : feedback et informations inadéquates* pour l'humain sur la défaillance des composants 2 ou 3.	Analyse d'un scénario de perte STPA incluant les facteurs causaux liés à l'algorithme de contrôle ou processus de décision pour les prises de décision de l'humain, la boucle de contrôle liée au feedback (temps de réponse, retard, informations fausses, mesures imprécises, inadéquates), et le comportement attendu des autres composants.	Réduction de l'inconnu en zone n°3 Placement au niveau de la zone n°2
Modification de la fonctionnalité afin de réduire les risques associés à SOTIF, conformément à la Clause 8.	Modification de la fonction : Une restriction de la fonction d'interaction entre l'IHM et l'humain. On désactive la possibilité d'interagir pour l'activation/ désactivation de volumes dans le cas où une défaillance du composant 3 ou 4 est détecté.	Contres mesures à la fois du scénario de perte et sur les contraintes du contrôleur.	On se retrouve dans la zone n°1 des scénarios à présent sûr et connus grâce à la modification de la fonctionnalité

*En effet, si le composant 2 ou 3 est défaillant, la vérification afin de valider la requête de l'humain passe par un autre composant qui lui est fonctionnel. Il accepte la requête et envoie une confirmation au superviseur à travers de l'IHM indiquant que sa requête est acceptée. Cependant, le retour affiché à l'écran indique des volumes activés qui ne le sont pas pour la détection de conflits, ce qui constitue un retour inadéquat.

STPA permet donc d'accompagner les clauses 5, 6, 7 et 8 partiellement lors de la réalisation des activités SOTIF (l'évaluation et la quantification ne pouvant être fournies par STPA, la clause 12 ne peut être fournie, concernant les clauses 9, 10 et 11 concernant la vérification des mesures apportées, cela est hors-scope de STPA).

III. LES RESULTATS ET CONCLUSION

A. Résultats de l'analyse STPA

Dans cette partie, nous décrirons uniquement les résultats de STPA indépendamment de l'analyse standard

L'analyse STPA a pu être entièrement réalisée avec le plug-in intégré à Capella. Cette analyse a permis de décliner des scénarios de pertes liés à la réalisation d'une UCA et d'autres de type actions de contrôle non exécutées ou mal exécutées. C'est donc un total de 27 scénarios de pertes qui ont été identifiés, pour un peu plus de 50 contraintes ou contre-mesures.
Voici un échantillon des scénarios identifiés :

Tableau 4 : Présentation de quelques scenarios

Identifiant du scénario	**Action de contrôle**	**UCA**	**Scénario**
LS-21	Publication d'un statut d'éligibilité	Composant_2 publie un statut d'éligibilité alors que le plan de vol n'est pas éligible	Le composant 2 ne publie pas le statut d'éligibilité pour des plans de vol car il croit incorrectement que ces derniers ne sont pas éligibles en raison de *son process model* incorrect. En effet, il ne reçoit pas les mises à jour des plans de vol, ou alors il les reçoit, mais en raison d'un problème interne, les plans de vol sont considérés comme non consistants. Par conséquent, il ne passe pas à ce statut. Ces plans de vol restent donc exclus de la détection de conflit. En conséquence, le système ne peut détecter de conflits liés à ces plans de vol. Le composant 2 ne publie pas le statut d'éligibilité pour des plans de vol car il croit incorrectement que ces derniers ont déjà été mis à jour en raison de *process model* incorrect sur l'état de la base de données stockant les statuts des plans de vol. En effet, la base de données

			remonte une mauvaise information sur le statut d'un autre plan de vol, ou alors les informations sont erronées de manière à ce que le composant 3 pense incorrectement que le plan de vol a déjà subi la mise à jour. Par conséquent, il ne passe pas à la transition. Ces plans de vol restent donc exclus de la détection. En conséquence, le système ne peut détecter de conflits liés à ces plans de vol.
LS-19	Envoyer un conflit à l'IHM	Composant_3 n'envoie pas le conflit avec la dernière mise à jour des données sur le conflit alors que le conflit non mis à jour est affiché	Un conflit doit être mis à jour et ensuite être également mis à jour sur l'IHM du contrôleur aérien. Du fait d'un *process model* incorrect au niveau des plans de vol bruts (utilisés pour effectuer la détection par l'algorithme de calcul interne) qui font croire au composant 3 qu'il n'a pas à recalculer une mise à jour de ces conflits, car ce dernier n'a pas reçu la mise à jour des plans de vol et donc ne peut pas procéder à ces nouveaux calculs (sur la base des nouveaux plans de vol bruts). Par conséquent, l'IHM n'affichera pas la dernière mise à jour du conflit avec les dernières informations. Un conflit doit être mis à jour et ensuite envoyé à l'IHM du contrôleur aérien. Dû à un *process model* incorrect au niveau de la base de données stockant les conflits, en raison d'un retour incorrect, le composant 3 a de mauvaises données sur le conflit (priorité du conflit, géométrie du conflit, altitude minimale) en le confondant avec un autre conflit. Par conséquent, son algorithme de contrôle décide de ne pas publier cette mise à jour de conflit à l'IHM car cela ne respecte pas les conditions nécessaires pour modifier une donnée affichée par le conflit. Par conséquent, l'IHM n'affichera pas la dernière mise à jour du conflit avec les dernières informations. Un conflit doit être mis à jour. Lorsque le composant 3 calcule la mise à jour du conflit, celui-ci n'arrive pas à calculer les données mises à jour (criticité ou type de conflit, par exemple) en raison d'un problème interne algorithmique. Son algorithme de contrôle ne publie donc pas cette mise à jour. Par conséquent, l'IHM n'affichera pas la dernière mise à jour du conflit avec les dernières informations.
LS-27	Publication d'un statut	Publie un statut d'exclusion d'un plan de vol alors qu'il est en train de subir une transition	Un plan de vol est éligible et est en train de réaliser un processus de transition vers l'état d'éligibilité, sachant que son statut actuel est celui d'un plan de vol forcé manuellement dans la base de données. Le composant 3 reçoit à ce moment une demande de dé forçage du plan de vol. Celui-ci pense incorrectement que le statut est encore celui de forcé manuellement car le statut d'éligibilité n'est pas encore mis à jour dans la base de données stockant les statuts des plans de vol. En effet, il est en cours de transition vers son nouveau statut lié à l'éligibilité. C'est donc le statut actuel qui est remonté au composant 3, ce qui constitue un feedback inadéquat à ce moment précis de la part de la base de données. Du fait de l'algorithme de contrôle du composant 3 qui est inadéquat (car ce dernier prend la décision sur la base de sa connaissance du statut sans vérifier si ce plan de vol est en cours de transition), le plan de vol est basculé vers un statut d'exclusion du plan de vol alors qu'il était en train de subir une transition pour devenir éligible. De

			ce fait, le système ne détectera pas de conflits liés à ce plan de vol.
LS-32	Demander l'activation de volumes de détection	Envoyer des activations de volumes de détection alors que les volumes sont mal définis.	Le superviseur émet une requête pour activer des volumes de détection. Cependant, en raison d'un *process model* incorrect du superviseur, il n'est pas au courant que les volumes sont mal définis, soit par le système lui-même (par le biais de messages d'erreur), soit par le superviseur technique. Cette information n'est pas remontée à l'interface homme-machine (IHM), ce qui laisse l'opportunité au superviseur d'envoyer sa requête. De plus, les requêtes précédentes du superviseur ont été acceptées en raison d'un retour d'information incorrect qui a validé ses demandes antérieures. Ces validations erronées renforcent la croyance du superviseur que les volumes sont correctement définis en offline. Par conséquent, cela conduit à des volumes mal définis, compromettant la capacité du système à détecter les conflits.

B. Comparaison avec les analyses safety standards

Nous avons voulu comparer les résultats des analyses *safety* standards afin d'évaluer la complémentarité de STPA avec celles-ci. Cette comparaison stricte s'avère relativement difficile étant donné que les deux approches ne sont pas les mêmes, même dans la déclinaison des dangers. Nous avons à notre disposition l'arbre de défaillance, les coupes minimales ainsi que les contre-mesures pour chaque danger. Nous avons décidé de réaliser une comparaison dans un sens unique : un danger STPA identifié dans un scénario de pertes doit être recherché dans l'arbre de défaillance.

Lors de la comparaison avec les résultats de l'analyse standard, nous avons réussi à identifier un scénario de perte lié à la non-détection de conflits qui n'était pas présent dans l'analyse standard. Ce scénario (non présent dans le tableau 4**Erreur ! Source du renvoi introuvable.**) est lié à une transition réalisée par le composant 3 qui exclut la détection d'un ou plusieurs plans de vol dans un contexte particulier. Ce scénario a été identifié en remettant en cause toutes les actions de contrôle permettant l'exercice d'une responsabilité en cherchant des contextes où elle pourrait être non sûre. Il est intéressant de noter que STPA a permis d'identifier la cause de ce potentiel scénario en remettant en question dès le début de l'analyse l'envoi même de la requête par l'humain lors de ce contexte précis, sans entrer dans des actions de contrôle liées au fonctionnement technique du composant, mais seulement celles liées à l'interaction possibles de l'humain avec l'IHM. La contrainte de contrôleur (STPA) haut niveau proposée était de ne pas laisser l'opportunité dans ce cas précis aux contrôleurs de demander cette action ou alors d'afficher le statut du plan de vol pour connaître son état actuel vis-à-vis de la détection de conflit (inclus dans la détection, exclu de la détection ou dans un état forcé manuellement). Pour la deuxième, cela n'a pas été implémenté car cela alourdirait l'affichage de l'IHM aux contrôleurs.

Un seul scénario a été identifié en plus de l'analyse classique ; cependant, cela ne signifie pas que STPA n'apporte aucune valeur ajoutée aux analyses classiques. La méthode STPA a tout de même réussi à couvrir, bien que partiellement en raison du niveau d'abstraction moins détaillé au niveau physique et logiciel (middleware, nœud, etc.), des scénarios de pertes identifiés dans l'analyse classique. À ce niveau, il est également pertinent de se questionner si cela constitue réellement un défaut. En effet, l'un des objectifs de la *safety* est d'établir des contraintes de niveau système. Une approche moins détaillée au niveau physique et logiciel peut offrir une vision plus holistique des dangers potentiels conduisant à des contraintes *safety* plus robustes et applicables à l'ensemble du système. Cela évite d'entrer dans des détails techniques logiciels qui pourraient ne pas être pertinents pour l'analyse *safety* globale, ce qui pourrait alourdir la structure de contrôle et son exploration. Les scénarios de types « action de contrôle non exécutée ou mal exécutée » dysfonctionnels sont aisés à identifier avec STPA dès lors que l'action de contrôle est modélisée.

Ce qui ajoute une plus-value et une complémentarité réside dans l'identification des scénarios de pertes en lien avec les facteurs causaux liés aux feedbacks, le fait d'avoir la structure de contrôle hiérarchique incluse dans l'analyse STPA qui nous permet de remettre en question toutes les actions de contrôle aussi bien d'origine composant ou humaines, d'établir une traçabilité des contre-mesures associées à chaque scénario de perte. En effet, le questionnement systématique concernant la présence nécessaire de feedbacks, les types de données émises en tant que feedback et l'influence des délais de remontée de ces derniers permettent d'identifier des facteurs causaux qui ne sont pas forcément évidents à prendre en considération. STPA est donc une méthode qui permet d'être proactive dans la conception d'une architecture système (si appliquée tôt dans le développement) en prenant en compte les aspects *safety* dès le début de la conception. Ainsi, elle permet d'adapter et de justifier certains choix d'architecture.

C. Les avantages, inconvénients et perspectives futures pour STPA :

L'un des avantages de STPA réside dans son analyse holistique du système, où les humains sont considérés au même niveau d'analyse que les autres composants. Notamment, STPA permet d'examiner de manière proactive la possibilité pour les humains d'envoyer des requêtes inappropriées vers un composant, même si ces requêtes seraient initialement rejetées (dans le cas nominal). Ce processus de prévention proactive des erreurs vise à anticiper et à empêcher de telles situations, en s'efforçant de créer un environnement (soit par des contraintes ou contre-mesures de type procédure à suivre) où les humains comprennent ce qui est approprié dans un contexte donné et agissent en conséquence, évitant ainsi le besoin même de rejeter des requêtes après qu'elles aient été soumises. Ceci est possible grâce à la structure de contrôle et à la notion d'UCA qui permet de remettre en question n'importe quelle action de contrôle dans un contexte particulier.

Nous notons également l'avantage de pouvoir réaliser l'analyse STPA avec le support d'une approche basée modèle, ce qui peut grandement faciliter la modélisation d'une structure de contrôle. Néanmoins, nous sommes dépendants de l'exactitude du modèle utilisé. Cependant, la structure de contrôle peut se modéliser sans aucun support de modèle, l'approche STPA étant indépendante. Un des inconvénients (amélioré par le plug-in) est le format de communication d'un scénario de pertes aux parties prenantes. Alors qu'un arbre de défaillance et des coupes minimales bénéficient de leur formalisme et de leur visualisation, d'une simplicité dans la compréhension de la réalisation d'un évènement redouté, la description d'un scénario de perte de STPA se fait sous la forme d'une rédaction narrative de l'analyste. Ce processus peut être fastidieux à relire et à comprendre pour une personne devant valider ou comprendre un scénario de pertes. C'est pourquoi nous avons voulu tester la mise en place d'un pont entre la réalisation d'un arbre de défaillance pour illustrer un scénario STPA. Il en ressort que les UCA ne se prêtent pas forcément toujours au formalisme des arbres de défaillance. Le plug-in permet de dépasser cette limitation en apportant un propre « formalisme de présentation STPA » dans un scénario de perte avec une arborescence tirée de la traçabilité établie par l'analyste humain.

L'un des défis rencontrés lors de la réalisation de l'étude STPA réside dans le contexte où elle est effectuée sur une architecture préexistante et validée. Dans de telles situations, STPA n'adopte pas nécessairement la position optimale, étant idéalement déployée en amont du cycle de conception pour établir des contraintes d'architecture de haut niveau. De plus, l'expérience tirée de cette étude suggère que STPA est particulièrement adaptée aux systèmes nécessitant une forte collaboration simultanée entre humains, ainsi qu'aux produits exigeant une interaction étroite entre l'homme et la machine. Aussi, il apparaît que réaliser une étude STPA à un niveau d'abstraction plus bas demanderait un effort non négligeable pour intégrer la technicité des logiciels (par exemple, middleware, modélisation des global locks). En outre, il est essentiel que la méthode STPA soit mise en œuvre par une équipe pluridisciplinaire, impliquant non seulement des experts *safety*, mais également d'autres disciplines pertinentes. Des cycles de validation doivent être établis pour garantir une approche organisationnelle efficace lors de la réalisation d'une étude STPA.

Enfin, nous souhaitons poursuivre ces travaux en étudiant comment STPA nous permettrait d'intégrer des contraintes supplémentaires telles que celles de l'Ecodesign, ainsi qu'une évaluation des coûts de réalisation de cette approche qui est complémentaire aux approches classiques de sûreté de fonctionnement.

REMERCIEMENTS

Les auteurs souhaitent remercier M. Tessier et M. Dreyer pour avoir répondu à nos différentes questions.

REFERENCES

[1] ISO/PAS 21448 (2019).

[2] Leveson, N. G., and Thomas, J. P., STPA Handbook, MIT Partnership for Systems Approaches to Safety and Security (PSASS) (2018) Materials.

[3] Leveson, N.G., Fleming, C.H., Spencer, M., Thomas, J., Wilkinson, C, Safety Assessment of Complex, Software-Intensive Systems. SAE International Journal of Aerospace 5 (1) (2012), 233-244.

[4] Leveson, N.G.. Engineering a safer world: systems thinking applied to safety. Cambridge: The MIT Press (2012).

[5] Thibault, M., Patrice, R., Sébastien, M., Alexandre, T., Gauthier, E., & de Kierzkowski, J. P. (2022, October). Assessment of STPA methodology and first feedback on use cases. In Congrès Lambda Mu 23 «Innovations et maîtrise des risques pour un avenir durable»-23e Congrès de Maîtrise des Risques et de Sûreté de Fonctionnement, Institut pour la Maîtrise des Risques.

[6] Constant, O., Ledinot, E., Le Noir, J., Towards Model-Based Support for STPA as a Capella Add-On, 11th European Congress on Embedded Real-Time Systems - ERTS'22 (2022).

[7] IEEE Architecture Working group, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, IEEE Std 1471-2000, IEEE, 2000.

[8] Ishimatsu, T, Leveson, N.G., Thomas, J. Miyamoto, Y. Nakao, H. "Modeling and Hazard Analysis Using Stpa", Proceedings of the 4th IAASS Conference, Making Safety Matter, 19–21 May 2010, Huntsville, Alabama, USA SP-680 (September 2010)