Modélisation et Analyse des Systèmes Cyber-physiques : Cas du Système de Surveillance Continue du Glucose

Feryel Benina[1,*], Nadira Benlahrache[2], Faiza Belala[2] and Ahmed Hadj Kacem[1]

[1]*ReDCAD Laboratory, ENIS, University of Sfax, Tunisia*
[2]*University of Constantine 2-Abdelhamid Mehri, LIRE Laboratory, BP : 67A, Constantine, Algeria*

Résumé

Les Systèmes Cyber-Physiques (CPS) sont des systèmes intégrant des composants physiques et des aspects informatiques inter-connectés pour offrir une fonctionnalité avancée. Leur analyse est cruciale pour garantir leur sûreté, leur fiabilité et leur efficacité. Cet article propose une approche novatrice qui combinent les diagrammes SysML et les spécifications AADL (Architecture Analysis and Design Language) afin de faciliter l'analyse formelle des CPS. Une transcription de certains concepts SySML à ceux du langage AADL est proposée et validée à travers une étude de cas de taille réelle, il s'agit d'un système intelligent de surveillance du taux de glucose pour les patients diabétiques.

Mots-clés

AADL, SysML, CPS, CGMS, Continuous Glucose Monitoring System

1. Introduction

Au cœur de la quatrième révolution industrielle, les systèmes embarqués et les technologies cyber-physiques ont pris une place prépondérante, influençant notamment les infrastructures, voitures autonomes et dispositifs médicaux [1]. En fusionnant le monde physique avec le digital, ces systèmes combinent électronique, informatique, et mécanique, d'où l'importance d'outils de modélisation adaptés. SysML, une déclinaison d'UML, est adapté aux systèmes complexes, proposant une vue holistique, tandis qu'AADL s'adresse spécifiquement aux systèmes embarqués critiques, avec un fort accent sur la performance et la fiabilité [2, 3]. Cette recherche vise à explorer la complémentarité de SysML et AADL, en prenant le système de surveillance du glucose comme cas d'étude, et insiste sur l'importance de la validation par des outils tels qu'Osate.

Le document est structuré en cinq sections majeures. Après une introduction aux concepts de base, nous plongeons dans une étude détaillée du CGMS. Ensuite, nous nous concentrons sur le processus de traduction de SysML vers AADL dans le contexte des MCPS (Medical Cyber-Physical System), mettant en évidence les synergies entre ces langages pour finalement conclure notre travail.

2. Concepts de bases SysML et AADL

Dans cette section, nous introduisons SysML; un langage de modélisation axé sur les systèmes, et AADL, dédié aux systèmes embarqués et critiques. Tandis que SysML met en avant une variété de diagrammes couvrant diverses facettes des systèmes, AADL [4] offre un cadre détaillé

TACC 2023: Tunisian-Algerian Joint Conference on Applied Computing, November 6 - 8, Sousse, Tunisia
*Corresponding author.
beninaferiel@gmail.com.tn (F. Benina); nadira.benlahrache@univ-constantine2.dz (N. Benlahrache); faiza.belala@univ-constantine2.dz (F. Belala); ahmed.hadjkacem@fsegs.usf.tn (A. Hadj Kacem)

pour étudier les composants et leurs propriétés, notamment la performance et la sécurité. Grâce à des outils comme Osate, les capacités d'AADL sont renforcées. La compréhension de ces deux langages est essentielle pour la traduction que nous discuterons ultérieurement.

2.1. Systems Modeling Language (SysML)

Le SysML (Systems Modeling Language) est une extension standardisée de l'UML (Unified Modeling Language), développée pour répondre aux défis spécifiques de l'ingénierie des systèmes. Ce langage de modélisation vise à être un outil fondamental pour la description, l'analyse, la vérification et la validation de systèmes complexes [5].

2.1.1. Diagrammes SysML

L'image 1 est une représentation hiérarchique des différents types de diagrammes inclus dans le langage de modélisation SysML. Voici une brève description de chacun: Les principaux

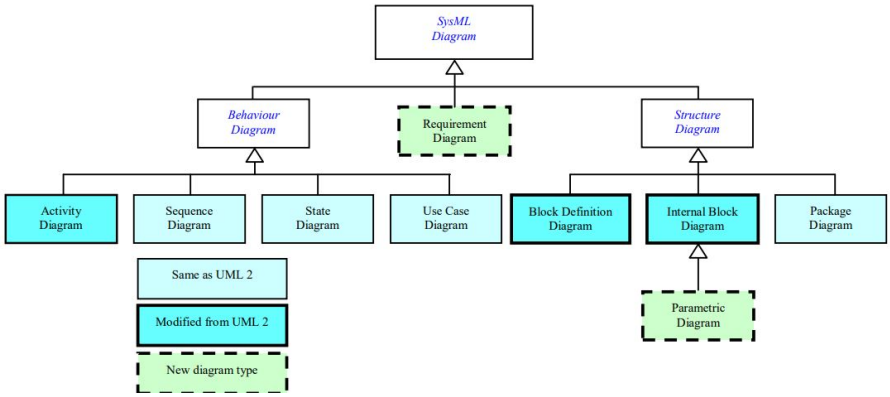

Figure 1: Présentation graphiques de diagrammes SysML [6]

digrammes SysML [7]:

1. **Behavior Diagram**: C'est une catégorie de diagrammes qui décrivent le comportement d'un système, tels que:
    - **Activity Diagram**: Représente le flux d'activités et de contrôle entre différentes entités.
    - **Sequence Diagram**: Illustration des interactions entre différentes entités dans un ordre séquentiel.
    - **State Machine Diagram**: Représente les états d'un objet ou d'une entité et les transitions entre ces états.
2. **Requirement Diagram**: Utilisé pour représenter les exigences d'un système et leurs relations.
3. **Structure Diagram**: C'est une catégorie de diagrammes qui décrivent la structure d'un système, on y trouve:
    - **Block Definition Diagram**: Représente les blocs (entités ou objets) et leurs relations.
    - **Internal Block Diagram**: Illustre l'interconnexion des parties à l'intérieur d'un bloc.
    - **Package Diagram**: Montre la manière dont les éléments sont regroupés et leurs dépendances.

4. **Parametric Diagram**: Utilisé pour définir des relations quantitatives et des contraintes entre les paramètres d'un système.

En somme, SysML est une extension de l'UML spécialement conçue pour les besoins de modélisation des systèmes d'ingénierie.

2.2. Architecture Analysis and Design Language (AADL)

L'Architecture Analysis and Design Language (AADL), établi par la SAE, est un standard pour les systèmes critiques, offrant une description détaillée des architectures. Il est primordial pour analyser performances et sécurité. Utilisé en aérospatiale, automobile et défense, AADL guide les ingénieurs, assurant une conception robuste tout au long du projet [8].

2.3. Les composants AADL

La table 1 présente les composants fondamentaux pour la modélisation de systèmes en utilisant AADL. Chaque élément a un rôle spécifique et ils peuvent être combinés de diverses manières pour créer des modèles de systèmes complexes et critiques [9]:

Type	Élément AADL	Description
Matériel	Processor	Composant utilisé pour l'exécution de threads et de processus.
Matériel	Virtual Processor	Peut être utilisé pour modéliser la distribution des threads sur un processeur matériel.
Matériel	Memory	Peut être physique ou virtuelle utilisée pour le stockage de données et de programmes.
Matériel	Bus	Composant utilisé pour la communication entre les composants matériels.
Matériel	Device	Représente un dispositif matériel, tel qu'un capteur ou un actionneur, qui interagit avec le système.
Logiciel	Thread	Une unité (ou fil) de traitement dans un système.
Logiciel	Process	Une entité logicielle indépendante avec son propre espace d'adressage.
Logiciel	Package	Pour l'organisation des éléments AADL en groupes logiques.
Logiciel	System	C'est le système global et représente l'élément racine de la hiérarchie.
Logiciel	Data	Les données ou les variables utilisées dans le système.
–	Abstract	Un élément abstrait qui peut être utilisé comme base pour d'autres éléments AADL matériels ou logiciels.

Table 1
Description de quelques éléments AADL organisés par type (matériel ou logiciel) avec leurs descriptions

2.4. Editeur d'Osate

OSATE (Open Source AADL Tool Environment) est un environnement de développement intégré destiné à soutenir la modélisation, l'analyse et la génération de code pour AADL [10].

- **Édition :** Osate fournit un éditeur de texte avec coloration syntaxique, complétion automatique et navigation pour faciliter la création et la modification de modèles AADL.

- **Validation :** L'outil peut détecter et signaler les erreurs dans les spécifications AADL, aidant ainsi les concepteurs à vérifier la cohérence et la validité de leurs modèles.
- **Analyse :** Osate est extensible et peut être intégré avec divers plugins pour effectuer des analyses spécialisées sur les modèles AADL, tels que la vérification de propriétés, l'analyse de latence, et bien d'autres.

Osate, de nature open-source avec une base de code dynamique, s'adapte constamment aux demandes de la communauté AADL.

3. Objectif de la proposition

Le domaine des systèmes cyber-physiques médicaux (MCPS) [11] est à la pointe de la transformation en médecine contemporaine. Ces systèmes, qui marient les aspects numériques et physiques, jouent un rôle pivot dans les avancées médicales. Cependant, assurer leur fiabilité et leur intégrité s'avère être un challenge considérable. Reconnaissant l'importance de la modélisation CPS [12] dans la compréhension et la prévision des interactions au sein de ces systèmes, ce papier avance une méthode qui fusionne les diagrammes SysML avec le langage AADL. Cette intégration donne lieu à une représentation précise des MCPS. En s'appuyant sur cette alliance, nous menons des évaluations formelles qui confirment la sûreté et les propriétés essentielles des MCPS. En fin de compte, nous visons à ce que les MCPS, tout en étant à la pointe de l'innovation, soient également fiables et sans risques pour les utilisateurs.

3.1. Originalité de l'approche

Notre approche se distingue nettement dans le paysage actuel de la modélisation des systèmes par ces principaux éléments:

- **Étape 1 - Mapping de SysML vers AADL:** Le cœur de notre approche repose sur le mapping précis des éléments SysML vers AADL. Ce processus de cartographie assure que chaque composant, relation et attribut en SysML trouve son équivalent approprié dans AADL.
- **Étape 2 - Analyse Formelle Post-Mapping:** Une fois le mapping effectué, une analyse formelle du modèle AADL est entreprise. Cette étape est cruciale pour s'assurer que le modèle transposé est non seulement correct, mais qu'il conserve aussi la sémantique du modèle SysML original.
- **Résultats de l'Analyse:** L'analyse formelle met en lumière plusieurs propriétés essentielles du modèle. Le mapping entre SysML et AADL est bien plus qu'une simple conversion. Il s'agit d'une fusion stratégique qui, lorsqu'elle est bien effectuée, débouche sur un modèle robuste, sécurisé et conforme aux attentes. Notre approche vise à perfectionner ce processus de mapping pour garantir des résultats optimaux. comme le montre la figure 2 ci-dessous.

3.2. Pertinence pour les systèmes cyber-physiques médicaux

En mettant l'accent sur les systèmes cyber-physiques médicaux, ce travail reconnaît l'importance cruciale de la sûreté dans les applications médicales. Les erreurs ou défaillances dans ces systèmes peuvent avoir des conséquences catastrophiques. [13]Par conséquent, notre approche est conçue pour offrir une assurance supplémentaire en termes de sûreté, répondant ainsi aux normes élevées requises dans le domaine médical.

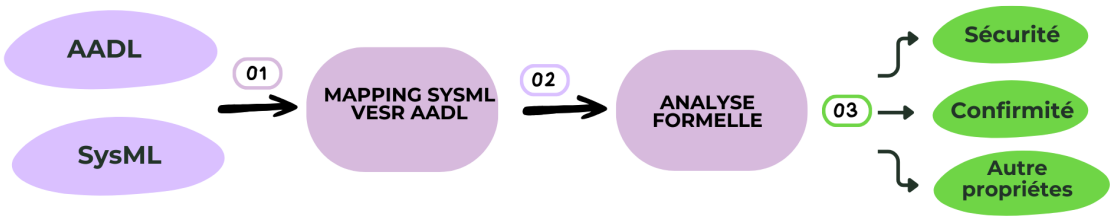

Figure 2: Représentation graphique de l'approche proposée

3.3. Application à l'étude de cas

Pour illustrer la valeur ajoutée de notre approche, nous avons étudié le système de contrôle de la glycémie. Cette étude pratique permet de démontrer la pertinence de notre contribution dans des situations réelles, tout en mettant en avant ses aspects spécifiques via la modélisation SysML et sa correspondance avec AADL.

3.4. Choix du Système

Le *Continuous Glucose Monitoring System* (CGMS) est conçu pour aider les patients diabétiques en surveillant constamment leur taux de glycémie et en fournissant des recommandations pour une meilleure gestion de la maladie [14]. La fiabilité et la sécurité de ce système sont d'une importance capitale. Afin d'atteindre ces standards, l'utilisation d'un langage comme AADL est essentielle. Ce dernier garantit une spécification minutieuse, en mettant l'accent sur les aspects de sécurité, veillant à une prise en charge efficace des patients.

3.5. Composants du CGMS

Le système CGMS est un ensemble de composants (fig. 3) interagissant entre eux pour assurer une supervision efficace des patients:

1. **Capteur de glucose:** Inséré sous la peau du patient, il mesure en continu le niveau de glucose dans le liquide interstitiel.
2. **Transmetteur:** Collecte les données du capteur et les transmet au dispositif récepteur ou à un smartphone.
3. **Récepteur/Dispositif d'affichage:** Peut être un appareil distinct ou une application mobile. Il affiche les niveaux actuels de glucose et les tendances.
4. **Application mobile:** Si utilisée, elle offre des fonctionnalités supplémentaires comme le partage de données et des notifications personnalisées.

4. Méthodologie de traduction SysML à AADL

La traduction entre SysML et AADL nécessite une compréhension approfondie des deux langages et une méthode systématique pour assurer une correspondance précise. [15] L'objectif est de capturer efficacement la sémantique des systèmes modélisés dans SysML et de les refléter correctement dans AADL.

4.1. Comparaison des travaux sur l'intégration de SysML et AADL

La table 2 compare deux approches liées à la traduction de modèles SysML en modèles AADL. Elle examine les différents aspects, y compris les objectifs principaux, les domaines d'application,

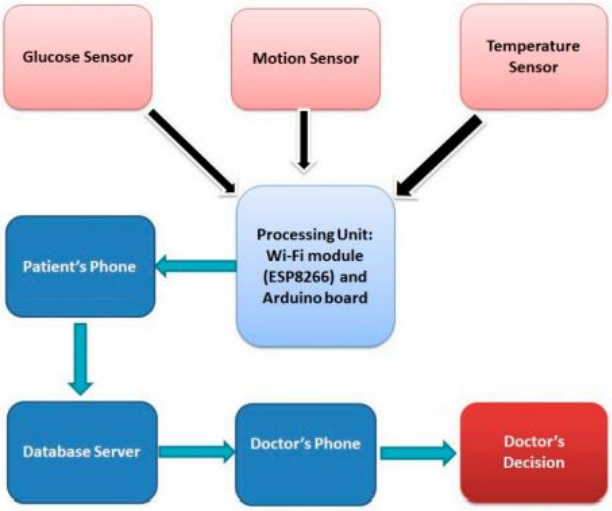

Figure 3: Présentation graphique de fonctionnement de CGMS [14]

les méthodologies, les niveaux d'abstraction, les contributions, les résultats majeurs, les cas d'utilisation spécifiques et les outils utilisés.

Aspect/Caractéristique	[15]	[16]
Objectif principal	Intégration SysML/AADL: conception, validation, implémentation de systèmes critiques.	Pont SysML-AADL: Traduction bidirectionnelle entre SysML et AADL.
Domaine d'application	Systèmes critiques: aviation, automobile.	Migration modèle: SysML vers AADL.
Méthodologie	Modèles SysML/AADL cohérents.	Conversion automatisée SysML vers AADL.
Niveau d'abstraction	Conception système et logiciel.	Traduction modèle SysML vers AADL.
Contribution	Intégration SysML/AADL: modélisation complète.	Transition simplifiée: SysML vers AADL.
Résultats	Cohérence et précision améliorées.	Réutilisation modèle SysML en AADL.
Cas d'utilisation	Systèmes critiques: sûreté.	Migration modèle: SysML vers AADL.
Outils	Eclipse Papyrus (SysML), OSATE (AADL).	Outil de conversion SysML vers AADL.

Table 2
Comparaison entre les travaux existants

Deux approches sont comparées : [15] qui se concentre sur une intégration transparente pour des systèmes critiques tandis que [16] propose un outil de traduction automatisée pour migrer des modèles SysML vers AADL. Chacun de ces travaux adopte des méthodologies distinctes pour répondre à des objectifs spécifiques d'applications.

Ainsi nous constatons que la traduction SysML vers AADL est une tâche complexe nécessitant une compréhension approfondie des deux langages. L'objectif est de faciliter la conception de systèmes critiques en capturant efficacement la sémantique des modèles SysML et en les reflétant correctement en AADL.

4.2. Importance de la Traduction entre SysML et AADL

La traduction entre SysML et AADL est essentielle pour fusionner les avantages des deux langages. SysML offre une perspective de haut niveau de la conception du système, [16]tandis qu'AADL fournit des détails techniques pour l'analyse des systèmes. Cette traduction permet d'assurer une conception et une mise en œuvre cohérentes, de la phase initiale de modélisation jusqu'à l'analyse et la validation.

4.3. Mapping SysML - AADL

Bien que SysML soit un langage de modélisation puissant et flexible, adapté à la description de systèmes complexes à différents niveaux d'abstraction, il présente certaines limitations lorsqu'il s'agit de décrire le comportement des systèmes en temps réel et d'analyser des propriétés spécifiques telles que les performances, la sûreté et la sécurité [7]. Plus précisément, les objectifs du mapping de SysML vers AADL sont les suivants :

- **Clarté et Compréhension** : Permettre aux utilisateurs de comprendre comment les concepts SysML se rapportent à ceux d'AADL et vice versa. Cela aide à éviter les erreurs d'interprétation et les incohérences lors de la transition entre les langages.
- **Facilitation de la Traduction** : Fournir un guide pratique pour la traduction des modèles SysML en modèles AADL. Les correspondances et les équivalences sont mises en évidence, ce qui accélère le processus de traduction.
- **Réutilisation de Modèles** : Identifier les opportunités de réutilisation de modèles SysML existants dans un environnement AADL. Cela peut permettre de gagner du temps et des ressources lors de la transition.

4.3.1. Correspondance des éléments du diagramme des Blocs SysML

La table 3 décrit les principaux choix faits concernant le passage de SysML vers AADL pour les éléments d'un diagramme de blocs SysML. Ainsi, nous avons opté pour:

1. **Paquetage ↔ Package:** En SysML, un paquetage sert à regrouper des éléments liés. Dans AADL, un `package` remplit une fonction similaire. L'exemple montre un paquetage nommé "HealthSystem" défini en AADL.
2. **Bloc ↔ Process/Thread/Device/Memory/processor/Bus:** Un bloc SysML représente un ensemble cohérent d'informations ou de composants. Dans AADL, cette notion est représentée par des éléments comme les `process`, `thread` ou `device`. L'exemple illustre un "device" nommé "CapteurDeGlucose".
3. **Block Interfaces↔ Features:** Les "Interface Blocks" en SysML définissent les points d'interaction d'un bloc. Dans AADL, ce concept est capturé par les `features`. L'exemple montre des features associées au processus "CapteurDeGlucose".
4. **Port ↔ Data/Event Port:** Les ports en SysML permettent les interactions entre les blocs. En AADL, les ports (qu'ils soient de données ou d'événements) servent à la même fin. L'exemple décrit un "device" appelé "Transmetteur" avec des ports d'entrée et de sortie pour transmettre des données.
5. **Relations/composition ↔ Subcomponents:** En SysML, les relations et compositions définissent comment les blocs sont reliés ou composés. En AADL, les `subcomponents` jouent ce rôle. L'exemple détaille la composition du "device CapteurDeGlucose" avec des sous-composants tels que "sensorUnit" et "processingUnit".
6. **Flux ↔ Connections:** Les flux en SysML décrivent les échanges d'informations entre blocs. En AADL, les `connections` remplissent cette fonction. L'exemple illustre les connexions entre les sous-composants du "device CapteurDeGlucose".

Élément SysML	Équivalent AADL	Exemple de code AADL

Paquetage	Package	`package HealthSystem`
		`public`
		`...`
		`end HealthSystem;`

Bloc	Process/Thread/Device	`Device CapteurDeGlucose`
		`...`
		`end CapteurDeGlucose;`

Interface Blocks	Features	`process CapteurDeGlucose`
		`features`
		`...`
		`end CapteurDeGlucose;`

Port	Data/Event Port	`device Transmetteur`
		`features`
		`inputDataFromCapteur: in data port GlucoseData;`
		`outputDataToSmartphone: out data port GlucoseData;`
		`end Transmetteur;`

Relations/ composition	Subcomponents	`device implementation CapteurDeGlucose.impl`
		`subcomponents`
		`sensorUnit: abstract SensorUnitType;`
		`processingUnit: abstract ProcessingUnitType;`
		`end CapteurDeGlucose.impl;`

flux	Connections	`device implementation CapteurDeGlucose.impl`
		`subcomponents`
		`sensorUnit: abstract SensorUnitType;`
		`connections`
		`conn1: port sensorUnit.dataOutput -> processingUnit.dataInput;`
		`conn2: port processingUnit.dataOutput -> outputGlucoseData;`
		`end CapteurDeGlucose.impl;`

propriétés	Property Set	`property set HealthProperties is`
		`Latency: aadlinteger;`
		`end HealthProperties;`

Table 3
Correspondance diagramme de Bloc SysML vers AADL

7. **Propriétés ↔ Property Set:** Les propriétés en SysML caractérisent les spécifications des blocs. En AADL, les `property sets` sont utilisés pour définir des propriétés spécifiques. L'exemple fournit un ensemble de propriétés "HealthProperties" qui spécifie la propriété *latence*.

4.3.2. Correspondance des éléments du diagramme d'exigences SysML

Propriété ↔ Value (valeur): Dans le contexte d'un système, une **propriété** sert à définir une caractéristique ou un attribut particulier de ce système. En AADL, une `value` (valeur) est utilisée pour assigner une quantité ou une donnée spécifique à une propriété (voir table 4). Pour illustrer, considérer un système AADL nommé "MySystem" défini avec une propriété "HealthProperties" ayant une valeur de 50. Cela pourrait suggérer que "MySystem" est censé

maintenir un niveau de santé de 50. Toutefois, l'interprétation exacte dépendrait du contexte du système et de la signification attribuée à "HealthProperties".

Élément SysML	Équivalent AADL	Exemple de code AADL

Exigence spécifique	Exigence (value)	`HealthProperties`
		`Properties:Value => 50;`
		`end MySystem;`

Table 4
Correspondance du diagramme d'exigences SysML vers AADL

4.3.3. Correspondance des éléments du diagramme d'états SysML

En SysML, un **état** représente une condition spécifique dans laquelle une entité (comme une machine ou un processus) se trouve à un moment donné. L'équivalent direct en AADL est le `mode`. Soit un dispositif AADL nommé "CapteurDeGlucose" (voir table 5). Ce dispositif peut avoir un mode "Actif", qui est défini comme son *mode initial* et pour lequel, on peut définir plusieurs autres modes.

Élément SysML	Équivalent AADL	Exemple de code AADL

État	Mode	`device CapteurDeGlucose`
		`features outputGlucoseData: out data port;`
		`...`
		`modes Actif: initial mode;`
		`...`
		`end CapteurDeGlucose;`

État de transition	Transition mode	`device CapteurDeGlucose`
		`features outputGlucoseData: out data port;`
		`...`
		`modes Actif: initial mode;`
		`...`
		`end CapteurDeGlucose;`

Table 5
Correspondance du diagramme d'état SysML vers AADL

En SysML, un **état de transition** décrit le mouvement ou le passage d'un état à un autre. Son équivalent en AADL est la `transition de modes`. Bien que l'exemple fourni (table 5) ne montre pas explicitement de transition de mode, en pratique, AADL permet de spécifier les conditions ou événements qui déclenchent le passage d'un mode à un autre.

4.3.4. Correspondance des éléments de diagramme d'activité SysML

En SysML, une **action** représente une opération ou une activité spécifique effectuée par une entité ou un composant du système. L'équivalent direct en AADL est le *thread*. Un thread en AADL est conçu pour exécuter une action ou une série d'actions, souvent en réponse à des entrées et en générant des sorties. Soit un thread AADL nommé "AnalyseDonneesThread" (voir table 6) qui est conçu pour analyser des données. Ce thread possède une *feature* (caractéristique) "inputData" qui est un port d'entrée pour recevoir des données à analyser.

Élément SysML	Équivalent AADL	Exemple de code AADL

Action	Thread	`thread AnalyseDonneesThread`
		`features`
		`inputData: in data port;`
		`...`
		`end AnalyseDonneesThread;`

Table 6
Correspondance des éléments du diagramme d'activité SysML vers AADL

5. Perspectives et applications futures

La relation entre **SysML** et **AADL**, axée principalement sur le mappage, s'avère cruciale dans le cadre de la modélisation et de la conception des systèmes **Cyber-Physiques (CPS)**. Cependant, notre démarche ne se restreint pas seulement à cette étape intermédiaire. En réalité, notre vision de la recherche embrasse une gamme plus étendue de domaines et d'utilisations, dans le but d'optimiser l'emploi de l'**AADL** et de stimuler l'essor des **CPS**. Un des principaux axes de cette approche est d'intensifier l'étude et la caractérisation des **MCPS** via l'**AADL**. Ces systèmes, incarnant d'importantes intégrations logiciel-matériel, se distinguent par leur complexité et s'insèrent dans des environnements opérationnels rigoureux. En adoptant l'**AADL** comme langage de modélisation privilégié, nous aspirons à cerner avec une plus grande précision les caractéristiques essentielles des **MCPS**, spécifiquement en termes de sécurité, de fiabilité, et d'efficacité. Cette démarche s'inscrit dans une perspective plus vaste ayant pour dessein de fusionner les forces de **SysML** et d'**AADL** pour la représentation et l'appréciation des **CPS**, à l'instar du système de surveillance continue du glucose. Nous sommes résolus à mettre en avant des méthodologies et des outils améliorés pour l'élaboration et l'évaluation de tels **CPS**, renforçant ainsi leur solidité et efficience.

6. Conclusion

En conclusion, ce travail a mis en lumière notre proposition de mapping du langage SysML vers le langage AADL pour les systèmes cyber-physiques (CPS), avec comme étude de cas, un système de surveillance en temps réel de la glycémie. Cette démarche de liaison entre SysML et AADL est essentielle pour assurer la sûreté des CPS, particulièrement dans des domaines sensibles comme la santé.

Toutefois, nous reconnaissons les limites de notre étude. Malgré nos progrès, les CPS regorgent de challenges non résolus. Notre méthode, bien que robuste, requiert des optimisations et pourrait nécessiter des adaptations pour d'autres contextes MCPS. Le cadre de la surveillance glycémique, en tant qu'exemple, pourrait ne pas être universellement transposable à tous les CPS. Il s'impose donc de continuer à adapter et peaufiner notre approche pour différentes applications CPS.

Remerciements

Ce travail a été partiellement soutenu par le projet LABEX-TA intitulé : "Méthodes Formelles pour le Génie Logiciel".

References

[1] L. Esterle, R. Grosu, Cyber-physical systems: challenge of the 21st century, Springerlink.com (2016). URL: http://www.springerlink.com, received August 26, 2016, accepted October 25, 2016, published online November 3, 2016.

[2] B. Fontan, Méthodologie de conception de systèmes temps réel et distribués en contexte UML/SysML, Réseaux et télécommunications [cs.ni], Université Paul Sabatier - Toulouse III, 2008. URL: URL_of_the_dissertation_on_HAL_OPEN_SCIENCE, nNT: tel-00258430.

[3] P. H. Feiler, J. Hansson, The Architecture Analysis & Design Language (AADL): An Introduction, Technical Report, Carnegie Mellon University, 2007.

[4] P. H. Feiler, D. P. Gluch, J. J. Hudak, The Architecture Analysis & Design Language (AADL): An Introduction, Technical Report, Performance-Critical Systems, 2006.

[5] P. de Saqui-Sannes, L. Apvrille, Avatar/ttool : un environnement en mode libre pour sysml temps réel, in: Proceedings of the [Nom de la conférence], Springer, Lieu de la conférence, 2022, pp. numéro de page de début–numéro de page de fin.

[6] O. M. Group, OMG Systems Modeling Language (OMG SysML), 2010. Copyright © 2006 by Object Management Group. Published and used by INCOSE and affiliated societies with permission.

[7] J. Smith, J. Doe, Limitations of sysml in real-time system modeling, Journal of Systems Engineering (2019).

[8] R. Varona-Gómez, E. Villar, Aads: Aadl simulation and performance analysis in systemc, Journal of Embedded Systems (2011).

[9] A.-E. Rugina, K. Kanoun, M. Kaâniche, ModÉlisation de la sÛretÉ de fonctionnement de systÈmes À partir du langage aadl, HAL OPEN SCIENCE (2006).

[10] P. Feiler, The open source aadl tool environment (osate), Carngie Mellon University (2019).

[11] I. Ghernaout, L. Elmhadhbi, M. Karray, B. Archimède, Un systÈme socio-cyber physique basÉ sur un rÉseau-infocentrÉ pour la collecte mobile de sang, in: 13ème CONFERENCE INTERNATIONALE DE MODELISATION, OPTIMISATION ET SIMULATION (MOSIM2020), AGADIR, Maroc, 2020, pp. 1–7. URL: https://hal.science/hal-03192809, conference held virtually in AGADIR.

[12] I. Graja, S. Kallel, N. Guermouche, S. Cheikhrouhou, A. H. Kacem, Modelling and verifying time-aware processes for cyber-physical environments, IET Cyber-Physical Systems: Theory & Applications (2018). URL: www.ietdl.org.

[13] M. Poursoltan, Cadre méthodologique et opérationnel pour la validation et l'amélioration des Systèmes Cyber-Physiques et Humains basé sur la modélisation et la simulation, Phd thesis, Université de Bordeaux, École Doctorale des Sciences Physiques et de l'Ingénieur, Bordeaux, 202X. Specialité: Automatique, Productique, Signal et Image, Ingénierie cognitique.

[14] A. Rghioui, J. Lloret, M. Harane, A. Oumnad, Un système intelligent de surveillance de la glycémie pour les patients diabétiques, Technical Report, Equipe de Recherche en Smart Communication-ERSC-Centre de Recherche E3S, EMI. Université Mohamed V de Rabat, 2020.

[15] P. de Saqui-Sannes, J. Hugues, Combining sysml and aadl for the design, validation and implementation of critical systems, Annals of Telecommunications (2012).

[16] H. Shackleton, S. Vestal, SysML to AADL Bridge: Automating the Translation of SysML into AADL for Analysis and Refinement, Technical Report, Adventium Labs, 2021.