Dossier de Spécifications

ProSE B1 2024 - CANvengers
Passerelle Android-CAN vers banc CAN réel ou simulé

Responsable du document	Théo BENARD
État du document	En réalisation
Version	1.1
Révision	0

AVERTISSEMENT :

Date	Actions	Auteur	Version	Révision
23/02/2023	Création du document	T. Bénard	0.0	0
02/03/2023	Intégration de l'architecture matérielle et logicielle	E. Declerck	0.0	1
02/03/2023	Intégration CU stratégique nominal	G. Marquette	0.0	2
04/03/2023	Ajout description générale et caractéristiques des acteurs	T. Bénard	0.0	3
07/03/2023	Intégration des interfaces du système	T. Rocher	0.0	4
07/03/2023	Ajout des sous CU	G. Marquette	0.0	5
07/03/2023	Correction de l'Introduction	E. Declerck	0.0	6
07/03/2023	Correction de bugs et de fautes + modification avertissement	T. Bénard	0.0	7
08/03/2023	Révision de l'architecture matérielle et logicielle	E. Declerck	0.0	8
11/03/2023	Correction des interfaces du système	T. Rocher	0.0	9
11/03/2023	Intégrations des IHM	C. Lenne	0.0	10
11/03/2023	Révision des sous-CU	T. Bénard	0.0	11
12/03/2023	Écriture sous-CU manquants	E. Declerck	0.0	12
14/03/2023	Révision des IHM	C. Lenne	0.0	13
15/03/2023	Révision des IHM	E. Declerck	0.0	14
16/03/2023	Relecture et correction	Equipe CANvangers	0.1	0
19/03/2023	Relecture et correction	E. Declerck	0.1	1
20/03/2023	Correction du CU Supprimer un élément	E. Declerck	0.1	2
20/03/2023	Relecture et Correction	G. Marquette	0.1	3
21/03/2023	Correction des CU Ajouter un objet et Reconnecter application	E. Declerck	0.1	4
22/03/2023	Correction des CU Reconnecter l'application CANdroid et table des versions	P. Trémoureux	0.1	5
28/03/2023	Correction des remarques suite aux relectures de chaque partie	T. Bénard	1.0	0

13/05/2023	Correction de l'architecture matérielle et logicielle + Ajout de la définition du RS485 CAN Hat au dictionnaire de domaine	E. Declerck	1.0	1
20/05/2023	Correction de plusieurs variantes dans les CU + modification de la description des interfaces logiques	T. Bénard	1.0	2
23/05/2023	Relecture finale avant l'audit normatif	T. Bénard	1.1	0

TABLE 2 – Table des évolutions et validations internes du document

Table des matières

1 Introduction

1.1 Objet

Ce dossier de spécifications a pour objectif de définir les fonctionnalités et exigences attendues par la société KEREVAL (désignée dans la suite du document comme le Client) pour le développement logiciel du prototype "Passerelle Android-CAN vers banc CAN réel ou simulé". Dans le contexte de développement d'un prototype, ce dossier de spécifications se focalise sur une étape du cycle de vie du produit, à savoir l'envoi et la réception de trames depuis une application Android, CANdroid, vers un Simulateur logiciel (ICSim) ou physique (Banc de test), en n'adressant que les fonctionnalités principales attendues pour ce prototype.

Ce document permettra aux équipes de conception, de réalisation et de test de la société KEREVAL de prendre connaissance de manière précise et détaillée les différentes parties informatiques du prototype. Les fonctionnalités et exigences présentées dans ce document ont été déterminées suite à l'étude du cahier des charges défini avec le Client et des rencontres avec le représentant du Client, A. Ribault (voir [CdC_KEREVAL_2023]).

Ce dossier de spécifications s'inspire de la norme [ISO/IEC/IEEE 29148 : 2018]. Il utilise des schémas et figures respectant la norme [UML_2.5]. De même, il respecte les exigences du Plan d'Assurance Qualité Logicielle (PAQL) défini par la société CANvengers [PAQL_B1_2024].

1.2 Portée

Ce document décrit les fonctionnalités et exigences du Système à l'Étude (SàE) constitué :

- Du logiciel CANgateway déployé sur une Raspberry Pi, permettant de faire le lien entre le Smartphone et Tableau de Bord.
- De l'application Android, CANdroid, permettant d'envoyer et de recevoir des trames grâce au protocole de communication TCP/IP.

1.3 Définitions, acronymes et abréviations

Les abréviations utilisées dans le présent document sont répertoriées et expliquées dans le tableau présenté ci-dessous. Les termes utiles pour interpréter correctement ce dossier de spécifications sont définis dans le dictionnaire de domaine présent dans ce dossier dans la partie 3.2.

Acronymes/Abréviations	Définitions
Client	Société KEREVAL, Numéro SIRET 44278921000030.
CU	Cas d'utilisation.
ID	Identifiant.
IEEE (*Institute of Electrical and Electronics Engineers*)	Association professionnelle internationale définissant des normes dans le domaine informatique et électronique.

IHM (Interface Homme Machine)	Moyens permettant aux utilisateurs de l'application CANdroid d'interagir avec le programme CANgateway.
N.A.	Non Applicable.
OMG (*Object Management Group*)	Consortium international à but non lucratif créé en 1989, dont l'objectif est de standardiser et de promouvoir le modèle objet sous toutes ses formes.
SàE (Système à l'Étude)	Ensemble composé de l'application Android, CANdroid, et du programme en C, CANgateway.
UML (*Unified Modeling Language*)	Notation graphique normalisée, définie par l'OMG et utilisée en génie logiciel.
TCP/IP (*Transmission Control Protocol/Internet Protocol*)	Protocole de communication utilisé pour transmettre des données entre l'application CANdroid et le programme CANgateway.

1.4 Références

Voici un tableau récapitulatif des documents utilisés pour le dossier de spécifications ainsi que les liens permettant d'accéder aux fichiers.

[CdC_KEREVAL_2023]	Société KEREVAL "Cahier des charges : développement d'une Passerelle Android-CAN vers banc CAN réel ou simulé", 2023.
[ISO/IEC/IEEE 29148 : 2018]	International standard, systems and software engineering life cycle processes requirements engineering, 2018, https ://standards.ieee.org/standard/29148-2018.html.
[UML_2.5]	OMG, Unified Modeling Language, version 2.5, 2015.
[Simulateur ICSim]	Simulateur d'un tableau de bord de voiture open source, version 3, 2007, https ://github.com/zombieCraig/ICSim.
[PAQL_B1_2024]	P. Trémoureux et T. Bénard, Plan d'Assurance Qualité Logicielle, 2023, `Git/doc/qualite/PAQL/version/`.
[TEST_B1_2024]	P. Trémoureux et T. Bénard, Plan de test, 2023, `Git/doc/test/plan_test/livrables/`.

1.5 Vue d'ensemble

Ce document de spécifications est structuré en 3 parties :

- La partie I, présente les objectifs et la portée de ce document.
- La partie II, intitulée "description générale", a pour objectif de présenter l'environnement et le contexte de cette passerelle CAN, ainsi que les fonctionnalités principales attendues.
- La partie III, présente en détail les IHM attendues, les fonctionnalités détaillées du projet ainsi que le dictionnaire du domaine.

2 Description générale

KEREVAL, une entreprise spécialisée dans les tests de systèmes embarqués véhicule, souhaite développer un démonstrateur CAN *simulator in the loop* pour permettre à ses équipes de monter en compétences sur le réseau CAN.

L'objectif est également de fournir une visualisation concrète de l'architecture du véhicule, du réseau CAN et du démonstrateur CAN pour les personnes novices dans le métier, tels que les nouveaux salariés lors de leur arrivé chez KEREVAL, ou des étudiants lors des forums.

Pour répondre à ce besoin, KEREVAL souhaite pouvoir utiliser un Simulateur de tableau de bord sur Linux (type ICSim) et/ou un Banc de test physique, connecté(s) à une carte électronique de type Raspberry Pi pour permettre la connexion au réseau CAN.

De plus, KEREVAL souhaite contrôler le système à distance via une application Android sur Smartphone appelée (CANdroid). Le Smartphone sera connecté au système (via la Raspberry Pi) par un réseau TCP/IP. Sur l'application, il sera possible d'ajouter et d'envoyer des trames mais également d'observer toutes les trames diffusées sur le bus CAN. Il sera possible d'enregistrer les trames diffusées sur le bus CAN dans un fichier de logs présent sur la Raspberry Pi.

En outre, l'entreprise souhaite pouvoir injecter des fautes dans Tableau de Bord afin d'en assurer le bon fonctionnement. Elle pourra réaliser cela en envoyant des trames qui seront susceptibles de mettre Tableau de Bord en erreur, et ainsi, en vérifier le comportement.

2.1 Caractéristiques des acteurs

Par le terme d'acteur, nous désignons tout rôle joué par une entité (morale ou physique) qui interagit directement ou non avec le SàE. Cette entité peut être une personne (généralement un utilisateur du système) ou un autre système.

Nous distinguons les acteurs dits directs (qui interagissent directement avec le SàE) et les acteurs dits indirects (qui n'ont pas d'interaction directe avec le SàE) mais qui sont à l'origine d'exigences à respecter par le SàE.

2.1.1 Acteurs directs

Les acteurs directs :

— **Utilisateur** : utilisateur principal du SàE. Celui-ci a la possibilité d'envoyer et de recevoir des trames CAN grâce à l'application CANdroid.

2.1.2 Acteurs indirects

— **N.A**

2.2 Environnement

2.2.1 Architecture matérielle et logicielle

Le diagramme de déploiement UML de la figure 1 du projet représente l'architecture matérielle et logicielle du SàE. Les conventions graphiques utilisées sont explicitées sur la figure 2. Ce diagramme de déploiement identifie les entités matérielles et/ou logicielles avec lesquelles le SàE doit intéragir et permet ainsi de déterminer les principaux échanges qu'il entretient avec son environnement.

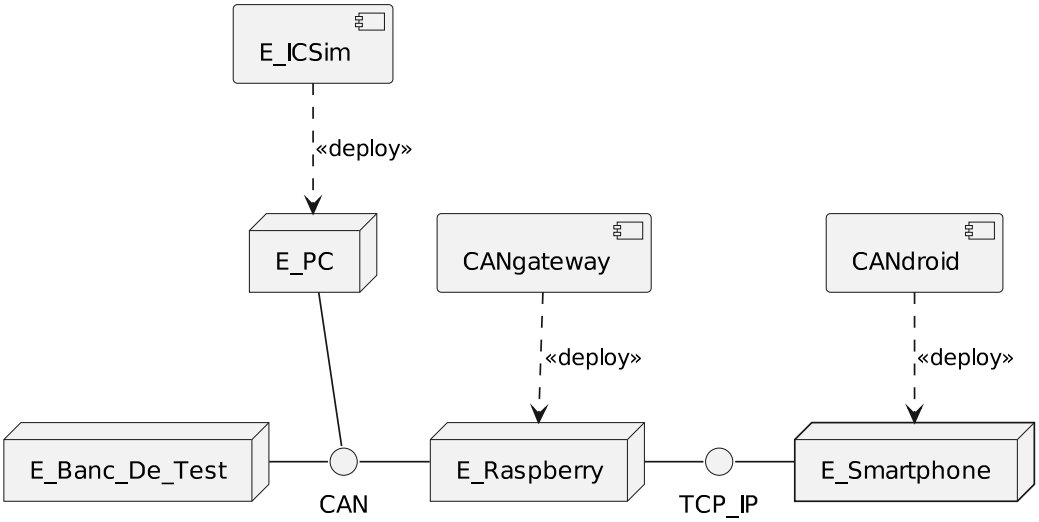

FIGURE 1 – Architecture matérielle et logicielle représentée par un diagramme de déploiement UML

Comme indiqué sur le diagramme, l'application CANdroid est déployée sur E_Smartphone. Le programme CANgateway est déployé sur E_Raspberry. E_Smartphone et E_Raspberry communiquent ensemble grâce au protocole TCP/IP. E_Raspberry récupère les trames CAN émisent par E_PC et/ou E_Banc_De_Test grâce au RS485 CAN Hat qui permet à E_Raspberry d'utiliser le protocole de communication CAN (Voir section 3.2). E_ICSim est un logiciel de simulation déployé sur E_PC. Il permettra aux futurs utilisateurs de se former aux protocoles de communication CAN sans avoir à utiliser un Banc de test encombrant. Par convention, le nom de ces entités est préfixé par les caractères "E_" (E pour Extern). Les caractéristiques de ces entités sont décrites dans le dictionnaire du domaine (voir section 3.2).

FIGURE 2 – Légende du diagramme de déploiement UML

2.2.2 Les interfaces du système

Cette section décrit les entrées et sorties dites « logiques » et « physiques » du SàE. En effet, nous différencions dans cette étude deux grands types d'entrées/sorties :

— Celles dites de haut niveau (dites aussi logiques) qui décrivent les données échangées et événements entre Utilisateur, le SàE et E_TableauDeBord. Les entrées/sorties logiques seront décrites à la section 2.2.2.1.

— Celles dites de bas niveau (dites aussi physiques) qui sont les entrées/sorties réellement échangées entre le SàE et E_TableauDeBord. Les entrées/sorties physiques (ou bas niveau) seront décrites au chapitre 2.2.2.3.

2.2.2.1 Les interfaces logiques

Le SàE est constitué d'une application Android nommée CANdroid et un programme embarqué sur la Raspberry Pi nommé CANgateway.

La figure 5 présente le contexte en faisant figurer les entrées/sorties logiques. Cette représentation du contexte est sous la forme d'un diagramme de communication UML. Les éléments soulignés correspondent à des ensembles de messages qui vont être détaillés.

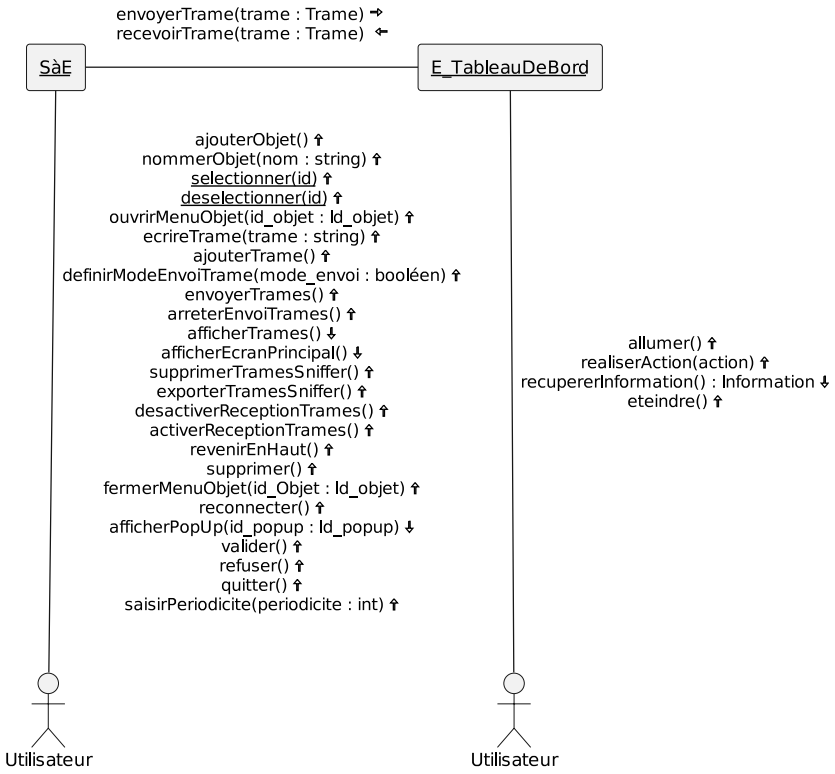

TABLE 5 – Contexte logique représenté par un diagramme de déploiement UML

2.2.2.2 Les interfaces avec les acteurs

L'acteur Utilisateur est en contact direct avec le SàE et Tableau de Bord. Les entrées et sorties entre Utilisateur et le SàE font référence aux fonctionnalités présentes sur l'IHM. Ces fonctionnalités sont décrites en détails dans la partie Exigences spécifiques (cf. section 3).

Les listes d'objets et de trames mentionnées dans les descriptions suivantes sont initialement vides lors du premier lancement de l'application, mais elles sont affichées sur l'IHM. Nous allons maintenant détailler ces entrées et sorties logiques.

2.2.2.2.1 En provenance de Utilisateur

Vers le SàE :

ajouterObjet() — Utilisateur ajoute un nouvel objet, un pop-up s'affiche ensuite pour nommer l'objet et valider l'ajout.
nommerObjet(nom : string) — Utilisateur nomme l'objet créé dans le pop-up qui s'est affiché. Une validation est attendue.
selectionner(id) :

- **selectionnerObjet(id_objet : Id_objet)** — Utilisateur sélectionne un objet d'ID id_objet, il se grise ensuite.
- **selectionnerTrame(id_trame : Id_trame)** — Utilisateur sélectionne une trame d'ID id_trame, elle se grise ensuite.

deselectionner(id) :

- **deselectionnerObjet(id_objet : Id_objet)** — Utilisateur désélectionne un objet d'ID id_objet, il se dégrise ensuite.
- **deselectionnerTrame(id_trame : Id_trame)** — Utilisateur désélectionne une trame d'ID id_trame, elle se dégrise ensuite. Lorsqu'il quitte le "Mode Envoi", les trames se désélectionnent automatiquement.

ouvrirMenuObjet(id_objet : Id_objet) — Utilisateur ouvre le menu déroulant d'un objet d'ID id_objet.
ecrireTrame(trame : string) — Utilisateur écrit une trame dans le menu déroulé d'un objet, le format de la trame est détaillé dans le dictionnaire de domaine (section 3.2).
ajouterTrame() — Utilisateur ajoute la trame écrite, un pop-up s'affiche pour définir le mode d'envoi de la trame.
definirModeEnvoiTrame(mode_envoi : booléen) — Utilisateur définit un mode d'envoi pour la trame à ajouter. Si vrai, l'envoi est ponctuel. Si faux, l'envoi est cyclique. Une validation est attendue.
envoyerTrames() — Utilisateur demande à envoyer la ou les trames préalablement ajoutées(s) et sélectionnée(s). L'application CANdroid passe alors en "Mode Envoi", et Utilisateur ne peut plus supprimer de trames ou ajouter d'objets. L'envoi des trames continue même si Utilisateur met en veille l'application CANdroid.
arreterEnvoiTrames() — Utilisateur arrête l'envoi des trames. L'application CANdroid quitte le "Mode Envoi" et Utilisateur a de nouveau accès aux fonctionnalités de suppression d'élément et d'ajout d'objet.

supprimerTramesSniffer() — Utilisateur supprime l'ensemble des trames affichées dans le sniffer.
exporterTramesSniffer() — Utilisateur demande à exporter l'ensemble des trames affichées dans le sniffer dans un fichier de logs (voir section 3.2).
desactiverReceptionTrames() — Utilisateur désactive la réception de trames, le SàE ne va plus afficher les trames en provenance de Tableau de Bord sur l'IHM.
activerReceptionTrames() — Utilisateur demande à activer la réception des trames, L'application CANdroid affiche les trames en provenance de Tableau de Bord sur l'IHM. La reception des trames continue même si Utilisateur met en veille l'application CANdroid.
revenirEnHaut() — Utilisateur revient en haut du fil de trames affichées dans le sniffer.
supprimer() — Utilisateur supprime les trames et objets sélectionnés. Si Utilisateur a sélectionné un ou plusieurs objets, les trames ajoutées dans le menu de ces objets seront aussi supprimées.
Un pop-up s'affiche pour confirmer ou non la suppression.
fermerMenuObjet(id_objet : Id_objet) — Utilisateur ferme le menu précédemment ouvert d'un objet d'ID id_objet.
reconnecter() — Utilisateur demande à reconnecter l'application CANdroid au programme CANgateway, un pop-up s'affiche et une validation est demandée.
valider() — Utilisateur choisit de valider son action lorsque qu'un pop-up s'affiche.
refuser() — Utilisateur choisit de refuser de réaliser son action lorsque qu'un pop-up s'affiche.
quitter() — Utilisateur quitte l'application CANdroid.
saisirPeriodicite(periodicite : int) — Utilisateur saisit la périodicité de la trame qu'il veut envoyer en mode cyclique.

Vers E_TableauDeBord :

Les fonctions décrites ci-dessous s'exécutent en arrière-plan.

allumer() — Utilisateur allume Tableau de Bord.
realiserAction(action) — Utilisateur réalise une action sur Tableau de Bord.
recupererInformation() : Information — Utilisateur récupère des informations sur Tableau de Bord grâce aux indicateurs visuels.
eteindre() — Utilisateur éteint Tableau de Bord.

2.2.2.2.2 En provenance du SàE

Vers Utilisateur :

afficherEcranPrincipal() — L'application CANdroid affiche l'écran principal.
afficherTrames() — L'application CANdroid affiche toutes les trames du sniffer, c'est-à-dire les trames qu'il a sniffées en provenance de Tableau de Bord, et les trames que Utilisateur a envoyées.
afficherPopUp(id_popup : Id_popup) — L'application CANdroid affiche un pop-up d'ID id_popup correspondant à l'action réalisée précédemment par Utilisateur (voir section 3.2).

Vers E_TableauDeBord :

envoyerTrame(trame : Trame) — Le SàE transmet à Tableau de Bord les trames que Utilisateur a décidé d'envoyer (voir section 3.2 pour les détails sur le type Trame).

2.2.2.2.3 En provenance de E_TableauDeBord

Vers le SàE :

recevoirTrame(trame : Trame) — Le SàE reçoit des trames envoyées par Tableau de Bord (voir section 3.2 pour les détails sur le type Trame).

2.2.2.3 Les interfaces physiques

Ce paragraphe précise les caractéristiques de chaque interface entre le logiciel et les composants matériels du système. Il s'agit des entrées/sorties physiques. Ce sont celles que devra réellement traiter le SàE en les interprétant ou les générant en événement logiques.

La figure 3 représente ce contexte physique avec un diagramme de communication UML.

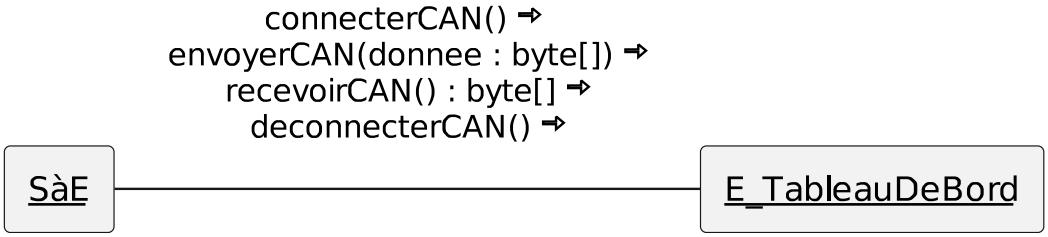

FIGURE 3 – Contexte physique représenté par un diagramme de déploiement UML

2.2.2.3.1 En provenance du SàE

Vers E_TableauDeBord :

connecterCAN() — Le SàE se connecte au bus CAN.
envoyerCAN(donnee : byte[]) — Le SàE envoie des trames sur le bus CAN.
recevoirCAN() : byte[] — Le SàE récupère les trames sortantes de Tableau de Bord sur le bus CAN.
deconnecterCAN() — Le SàE se déconnecte du bus CAN.

2.3 Fonctions principales développées

Ce chapitre présente les principales fonctionnalités développées dans l'incrément 1 et 2 en utilisant une approche basée sur les cas d'usage et les cas d'utilisation (CU). Il met en évidence les fonctionnalités clés implémentées dans ces deux incréments, en les reliant aux cas d'usage et aux cas d'utilisation pertinents.

2.3.1 Rappel sur les cas d'usage

Les cas d'usage recensent les étapes essentielles du cycle de vie d'un produit depuis la fabrication du produit jusqu'à l'élimination ou le recyclage du produit. À chaque étape du cycle de vie correspond un cas d'usage (si cette étape induit des fonctionnalités à définir pour le produit considéré). Pour chaque cas d'usage, plusieurs cas d'utilisation distincts peuvent être définis.

Parmi les cas d'usage, on retrouve généralement ceux de fabrication du produit (comprenant ou non les activités de test du produit fabriqué), de conditionnement (paramétrage éventuel, expédition et transport), de commercialisation (paramétrage éventuel, mode de démonstration, installation sur site...), d'utilisation (par le ou les utilisateurs), de maintenance (SAV ou diagnostique), de désinstallation et de recyclage (élimination ou valorisation).

2.3.2 Rappel sur les cas d'utilisation

Un cas d'utilisation (CU) représente un ensemble d'interactions entre un ou des acteurs et le système à spécifier. Ce cas d'utilisation est souvent lié à un ou parfois plusieurs cas d'usage. Un CU est principalement décrit par un scénario d'utilisation (nommé scénario nominal), scénario d'une utilisation représentative du système. Ces CU sont ensuite détaillés jusqu'à un niveau de décomposition suffisant pour décrire les fonctions attendues du système.

2.3.2.1 Représentation graphique des CU

Les CU peuvent être représentés sous forme graphique, voir la figure 4 pour un exemple. Les acteurs directs sont représentés sous forme de petits personnages. Dans les bulles, sont représentés les cas d'utilisation. Un trait entre un acteur et un CU indique que l'acteur participe à ce CU. Les liens hachurés entre CU, étiquetés par le mot «use» (ou «include»), indiquent que ce CU fait appel à l'autre CU — on parle alors de sous-cas d'utilisation. Les liens hachurés entre CU, étiquetés par le mot «extend», indique qu'il s'agit d'une extension d'un CU : un CU qui ne se déclenche que sous certaines conditions.

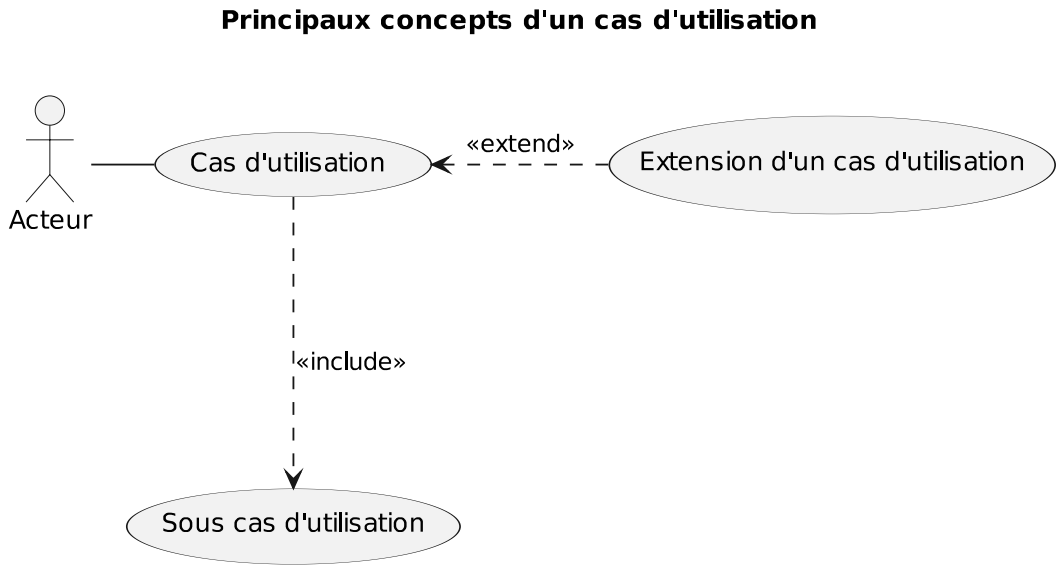

FIGURE 4 – Légende explicative d'un diagamme de cas d'utilisation UML

2.3.2.2 Représentation textuelle des CU

La description textuelle des cas d'utilisation est souvent présentée sous forme d'un tableau constitué de champs suivants :

Titre	Rappel en quelques mots l'objectif principal du cas d'utilisation.
Résumé	Décris brièvement le comportement du cas d'utilisation.
Portée	Définis la portée de conception du CU (étendue spatiale).
Niveau	Niveau de granularité du cas d'utilisation (stratégique, utilisateur ou sous-fonction).
Acteurs directs	Acteurs qui participent au CU.
Acteurs indirects	Acteurs qui ne participent pas au CU, mais qui ont un intérêt dans sa réalisation.
Préconditions	Ensemble des conditions qui doivent être vérifiées avant le déroulement du CU. Les préconditions, sans mention contraire explicite, des CU parents au CU doivent toujours être vérifiées.
Garanties minimales	Définis ce qui est garanti par le système à l'étude même en cas d'échec du cas d'utilisation.
Garanties en cas de succès	Définis les garanties en cas de succès (par le scénario nominal ou par ses variantes).

Scénario nominal	C'est un scénario représentatif de l'utilisation du système où tout se passe bien. Il se termine par la réussite des objectifs. Il peut être constitué d'une condition déclenchant le scénario, d'un ensemble d'étapes, d'une condition de fin, et éventuellement d'extensions ou de variantes. Une étape peut être une interaction entre acteurs, une étape de validation, ou un changement interne.
Variantes	Lorsqu'il y a plusieurs façons de procéder à une même étape sans remise en cause du scénario nominal.
Extensions	Définissent les autres scénarios que le scénario nominal (par exemple ceux qui se terminent par un échec). Ils se déclenchent sur des conditions spécifiques détectées par le SàE.
Informations complémentaires	Informations diverses nécessaires à la compréhension du CU.

2.3.3 CU Échanger des trames CAN

2.3.3.1 Description graphique

FIGURE 5 – CU Échanger des trames CAN

2.3.3.2 Description textuelle

Titre	Échanger des trames CAN.
Résumé	Utilisateur peut échanger des trames CAN avec Tableau de Bord.
Portée	SàE.
Niveau	Stratégique.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Tableau de Bord et la Raspberry Pi sont fonctionnels. — Le programme CANgateway est installé sur la Raspberry Pi. — L'application CANdroid est téléchargée sur le Smartphone. — Le matériel est fonctionnel.
Garanties minimales	— L'application CANdroid est utilisable.
Garanties en cas de succès	— Utilisateur peut échanger des trames CAN entre Tableau de Bord et l'application CANdroid. — Utilisateur est capable d'observer les trames CAN reçues et d'en envoyer des nouvelles. — Utilisateur est capable d'ajouter des objets ainsi que d'ajouter des trames à envoyer.

Scénario nominal	1. Utilisateur démarre le SàE. 2. Le SàE commence à recevoir des trames. 3. Utilisateur demande à ajouter un objet. 4. Utilisateur demande à ajouter une trame. 5. Utilisateur demande à envoyer des trames. 6. Le SàE diffuse les trames sur le bus CAN. 7. Utilisateur demande à arrêter l'envoi des trames. 8. Utilisateur demande à supprimer un élément. 9. Utilisateur stoppe le SàE.

Variantes	**2,5 [La connexion n'est pas établie et Utilisateur ne souhaite pas se reconnecter]** 2,5.a.1. Va en 3. **2,5 [La connexion n'est pas établie & Utilisateur souhaite se reconnecter]** 2,5.b.1. Utilisateur demande à reconnecter l'application CANdroid au programme CANgateway. 2,5.b.2. Va en 2. **3,4,5,7,8 [Utilisateur souhaite stopper le SàE]** 3,4,5,7,8.a.1. Va en 9. **3,4,5,7,9 [Un élément existe & Utilisateur souhaite supprimer un élément]** 3,4,5,7,9.a.1. Va en 8. **3,4,5,8,9 [Des trames sont en cours d'envoi & Utilisateur souhaite arrêter l'envoi de trames]** 3,4,5,8,9.a.1. Va en 7. **3,4,7,8,9 [Au moins une trame existe, aucune trame n'est en cours d'envoi & Utilisateur souhaite envoyer une trame]** 3,4,7,8,9.a.1. Va en 5. **3,5,7,8,9 [Utilisateur souhaite ajouter une trame & au moins un objet existe]** 3,5,7,8,9.a.1. Va en 4. **4,5,7,8,9 [Utilisateur souhaite ajouter un objet]** 4,5,7,8,9.a.1. Va en 3. **5 [Il n'y a pas de trames à envoyer]** 5.a.1. Va en 8. **7 [Il n'y a pas de trames en cours d'envoi]** 7.a.1. Va en 8. **8 [Il n'y a pas d'éléments à supprimer]** 8.a.1. Va en 9.

Extensions	**2-8 [Quitter]** 2-8.a.1. Utilisateur stoppe le SàE. 2-8.a.2. Fin du CU.
Informations complémentaires	N.A.

2.3.4 CU Démarrer le SàE

2.3.4.1 Description graphique

Voir figure 5.

2.3.4.2 Description textuelle

Titre	Démarrer le SàE.
Résumé	Utilisateur démarre le SàE et les différents périphériques dont il a besoin.
Portée	SàE.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Utilisateur à accès aux différents éléments du SàE ainsi que des différents périphériques dont il a besoin.
Garanties minimales	N.A.
Garanties en cas de succès	— Utilisateur peut utiliser l'application CANdroid librement.
Scénario nominal	1. Utilisateur met en fonctionnement le Simulateur ICSim. 2. Utilisateur connecte la Raspberry Pi au bus CAN. 3. Utilisateur met sous tension la Raspberry Pi. 4. Le programme CANgateway essaye de se lancer. 5. La Raspberry Pi informe Utilisateur que le programme CANgateway est lancé. 6. Utilisateur démarre l'application CANdroid. 7. L'application CANdroid affiche EcranPrincipal. 8. L'application CANdroid se connecte au programme CANgateway. 9. L'application CANdroid met à jour EcranPrincipal.

Variantes	**1 [Mise en fonctionnement du Banc de test uniquement]** 1.a.1. Utilisateur allume le Banc de test. 1.a.2. Va en 2. **1 [Mise en fonctionnement du Simulateur ICSim et du Banc de test]** 1.b.1. Utilisateur allume le Simulateur ICSim. 1.b.2. Utilisateur allume le Banc de test. 1.b.3. Va en 2. **1 [Utilisateur ne souhaite pas utiliser Tableau de Bord]** 1.c.1. Va en 2. **1-3 [Utilisateur ne souhaite pas utiliser la Raspberry Pi]** 1-3.a.1. Va en 6. **5 [Le programme CANgateway ne s'est pas lancé]** 5.a.1 Utilisateur démarre l'application CANdroid. 5.a.2 L'application CANdroid affiche EcranPrincipal. 5.a.3 Fin du CU. **8 [La connexion échoue & Utilisateur ne souhaite pas se reconnecter]** 8.a.1. Fin du CU. **8 [La connexion échoue & Utilisateur souhaite se reconnecter]** 8.b.1. Utilisateur demande à reconnecter l'application CANdroid au programme CANgateway. 8.b.2. Fin du CU.
Extensions	N.A.
Informations complémentaires	N.A.

2.3.5 CU Reconnecter l'application CANdroid

2.3.5.1 Description graphique

Voir figure 5.

2.3.5.2 Description textuelle

Titre	Reconnecter l'application CANdroid.
Résumé	Utilisateur peut reconnecter l'application CANdroid au programme CANgateway.
Portée	Application CANdroid.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— L'application CANdroid est déconnectée du programme CANgateway.
Garanties minimales	N.A.
Garanties en cas de succès	— L'application CANdroid est de nouveau connectée au programme CANgateway. — Utilisateur peut utiliser l'application CANdroid librement.
Scénario nominal	1. Utilisateur demande la reconnexion de l'application CANdroid au programme CANgateway. 2. L'application CANdroid affiche PopupDemandeReconnexion. 3. Utilisateur valide la reconnexion. 4. L'application CANdroid affiche EcranPrincipal. 5. L'application CANdroid se reconnecte au programme CANgateway. 6. L'application CANdroid met à jour EcranPrincipal.

Variantes	**3 [Utilisateur souhaite abandonner la reconnexion]** 3.a.1. Utilisateur annule la reconnexion. 3.a.2. L'application CANdroid affiche EcranPrincipal. 3.a.3. Fin du CU. **5 [La connexion échoue & Utilisateur ne souhaite pas se reconnecter]** 5.a.1. Fin du CU. **5 [La connexion échoue & Utilisateur souhaite se reconnecter]** 5.b.1. Va en 1.
Extensions	N.A.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.3.6 CU Recevoir des trames

2.3.6.1 Description graphique

Voir figure 5.

2.3.6.2 Description textuelle

Titre	Recevoir des trames.
Résumé	L'application CANdroid reçoit et affiche les trames du réseau CAN.
Portée	Application CANdroid.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré. — L'application CANdroid est connectée au programme CANgateway. — La Raspberry Pi est connectée au bus CAN. — Tableau de Bord est connecté au bus CAN.
Garanties minimales	N.A.
Garanties en cas de succès	— Les trames sont affichées sur le sniffer de l'application CANdroid.
Scénario nominal	1. Utilisateur commande un actionneur sur Tableau de Bord. 2. Le SàE récupère les trames du réseau CAN. 3. L'application CANdroid met à jour EcranPrincipal. 4. Utilisateur demande à interagir avec le sniffer.

Variantes	**1,2 [La connexion échoue & Utilisateur ne souhaite pas se reconnecter]** 1,2.a.1. Va en 3. **1,2 [La connexion échoue & Utilisateur souhaite se reconnecter]** 1,2.b.1. Utilisateur demande à reconnecter l'application CANdroid au programme CANgateway. 1,2.b.2. Va en 1. **4 [Utilisateur souhaite commander un autre actionneur]** 4.a.1. Va en 1.
Extensions	**2-3 [Utilisateur a demandé l'arrêt de la réception de trames]** 2-3.a.1. Fin du CU.
Informations complémentaires	N.A.

2.3.7 CU Interagir avec le sniffer

2.3.7.1 Description graphique

Voir figure 5.

2.3.7.2 Description textuelle

Titre	Interagir avec le sniffer.
Résumé	Utilisateur interagit avec le sniffer.
Portée	Application CANdroid.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré. — L'application CANdroid est connectée au programme CANgateway. — La Raspberry Pi est connectée au bus CAN. — Tableau de Bord est connecté au bus CAN.
Garanties minimales	N.A.
Garanties en cas de succès	— Utilisateur est capable d'exporter et de supprimer les trames du sniffer. — Utilisateur est capable d'arrêter la mise à jour du sniffer en temps réel et de revenir en haut du fil lorsqu'il fait défiler les trames.

Scénario nominal	1. Utilisateur demande de revenir en haut du fil. 2. L'application CANdroid va en haut du fil. 3. Utilisateur demande d'effacer les trames. 4. L'application CANdroid efface les trames affichées. 5. Utilisateur demande l'arrêt de la réception des trames. 6. L'application CANdroid stoppe l'affichage de nouvelles trames. 7. Utilisateur demande d'enregistrer les trames reçues sur le Smartphone. 8. L'application CANdroid enregistre les trames dans un fichier de log (voir section 3.2). 9. Utilisateur demande la reprise de la réception des trames. 10. L'application CANdroid reprend l'affichage de nouvelles trames.
Variantes	**1,3,5,7 [La réception de trames est arrêtée & Utilisateur souhaite reprendre la réception de trames]** 1,3,5,7.a.1. Va en 9. **1,3,5,7,9 [Utilisateur ne souhaite rien faire]** 1,3,5,7,9.a.1. Fin du CU. **1,3,9 [La réception de trames est en cours & Utilisateur souhaite arrêter la réception de trames]** 1,3,9.a.1. Va en 5. **1,3,9 [La réception de trames est en cours & Utilisateur souhaite enregistrer les trames reçues]** 1,3,9.b.1. Va en 5. **1,3,9 [La réception de trames est arrêtée & Utilisateur souhaite enregistrer les trames reçues]** 1,3,9.c.1. Va en 7. **1,5,7,9 [Utilisateur souhaite effacer les trames]** 1,5,7,9.a.1. Va en 3. **3,5,7,9 [Utilisateur souhaite aller en haut du fil]** 3,5,7,9.a.1. Va en 1.

Extensions	N.A.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.3.8 CU Ajouter un objet

2.3.8.1 Description graphique

Voir figure 5.

2.3.8.2 Description textuelle

Titre	Ajouter un objet.
Résumé	Utillisateur ajoute un objet sur l'application CANdroid.
Portée	Application CANdroid.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— L'application CANdroid est démarrée.
Garanties minimales	N.A.
Garanties en cas de succès	— Un objet est ajouté.
Scénario nominal	1. Utilisateur demande l'ajout d'un objet. 2. Le SàE choisit un Nom d'objet par défaut. 3. L'application CANdroid affiche PopupAjoutObjet. 4. Utilisateur saisit un nom d'objet. 5. Utilisateur valide l'ajout. 6. L'application CANdroid met à jour EcranPrincipal.

Variantes	**1 [Nombre maximal d'objets atteint & Utilisateur souhaite ajouter un objet]** 1.a.1. L'application CANdroid affiche PopupErreurNombreObjet. 1.a.2. Utilisateur ferme PopupErreurNombreObjet. 1.a.3. Fin du CU. **4 [Utilisateur ne souhaite pas saisir de nom]** 4.a.1. Va en 5. **6 [Utilisateur souhaite saisir un nom qui existe déjà]** 6.a.1. L'application CANdroid affiche PopupErreurAjoutObjet. 6.a.2. Va en 4.
Extensions	**4-5 [Utilisateur souhaite annuler]** 4-5.a.1. Utilisateur ferme PopupAjoutObjet. 4-5.a.2. L'application CANdroid affiche EcranPrincipal.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.3.9 CU Ajouter une trame

2.3.9.1 Description graphique

Voir figure 5.

2.3.9.2 Description textuelle

Titre	Ajouter une trame.
Résumé	Utilisateur ajoute une trame CAN associée à un objet avec son mode d'envoi.
Portée	Application CANdroid.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré. — Un objet existe.
Garanties minimales	N.A.
Garanties en cas de succès	— Une trame est créée dans le bon objet.
Scénario nominal	1. Utilisateur déroule le menu de l'objet. 2. L'application CANdroid met à jour EcranPrincipal. 3. Utilisateur saisit la trame. 4. Utilisateur valide la saisie de la trame. 5. L'application CANdroid affiche PopupModeEnvoiTrame. 6. Utilisateur choisit le mode d'envoi ponctuel. 7. Utilisateur valide le mode d'envoi. 8. L'application CANdroid met à jour EcranPrincipal.

Variantes	**1 [L'objet est déjà déplié]** 1.a.1. Va en 3. **4 [Nombre maximal de trames atteint & Utilisateur souhaite ajouter une trame]** 4.a.1. L'application CANdroid affiche PopupErreurNombreTrame. 4.a.2. Utilisateur ferme PopupErreurNombreTrame. 4.a.3. Fin du CU. **5 [Utilisateur souhaite ajouter une trame qui n'a pas le bon format]** 5.a.1. L'application CANdroid affiche PopupErreurSaisieTrame. 5.a.2. Utilisateur ferme PopupErreurSaisieTrame. 5.a.3. Va en 3. **6 [Utilisateur choisit le mode cyclique & souhaite changer la périodicité]** 6.a.1. Utilisateur choisit le mode cyclique. 6.a.2. Utilisateur saisit la périodicité. 6.a.3. Va en 7. **6 [Utilisateur choisit le mode cyclique & souhaite garder la périodicité par défaut]** 6.b.1. Utilisateur choisit le mode cyclique. 6.b.2. Va en 7.
Extensions	**6-8 [Utilisateur souhaite annuler]** 6-8.a.1. Utilisateur ferme PopupModeEnvoiTrame. 6-8.a.2. L'application CANdroid affiche EcranPrincipal.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.3.10 CU Envoyer des trames

2.3.10.1 Description graphique

Voir figure 5.

2.3.10.2 Description textuelle

Titre	Envoyer des trames.
Résumé	Utilisateur utilise l'application CANdroid afin d'envoyer une trame CAN au Tableau de Bord.
Portée	SàE.
Niveau	Utilisateur
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré. — Un ou plusieurs objets existent. — Une ou plusieurs trames existent. — L'application CANdroid est connectée au programme CANgateway. — La Raspberry Pi est connectée au bus CAN. — Tableau de Bord est connecté au bus CAN.
Garanties minimales	N.A.
Garanties en cas de succès	— La trame CAN est envoyée à Tableau de Bord.

Scénario nominal	1. Utilisateur déroule le menu d'un objet. 2. L'application CANdroid met à jour EcranPrincipal. 3. Utilisateur sélectionne une trame. 4. L'application CANdroid met à jour EcranPrincipal. 5. Utilisateur demande à envoyer la trame. 6. Le SàE diffuse la trame sur le bus CAN. 7. L'application CANdroid affiche EcranPrincipal.
Variantes	**1 [L'objet est déjà déplié]** 1.a.1. Va en 3. **3 [Une trame est sélectionnée & Utilisateur ne souhaite pas sélectionner de trames]** 3.a.1. Va en 5. **3 [Aucune trame n'est sélectionnée & Utilisateur ne souhaite pas sélectionner de trames]** 3.b.1. Va en 7. **5 [Utilisateur souhaite sélectionner plusieurs trames]** 5.a.1. Va en 1. **5 [Utilisateur ne souhaite plus envoyer de trames]** 5.b.1. Utilisateur désélectionne les trames grisées. 5.b.2. L'application CANdroid met à jour EcranPrincipal. 5.b.3. Fin du CU. **5 [Utilisateur souhaite désélectionner une trame]** 5.c.1. Utilisateur désélectionne une trame. 5.c.2. Va en 3. **6 [Une ou plusieurs trames sélectionnées sont paramétrées avec le mode d'envoi cyclique]** 6.a.1. Le SàE commence l'envoi des trames sélectionnées sur le bus CAN. 6.a.2. Va en 7.
Extensions	N.A.

Informations complémentaires	N.A.

2.3.11 CU Arrêter l'envoi des trames

2.3.11.1 Description graphique

Voir figure 5.

2.3.11.2 Description textuelle

Titre	Arrêter l'envoi des trames.
Résumé	Utilisateur utilise l'application CANdroid afin d'arrêter l'envoi des trames à Tableau de Bord.
Portée	SàE.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré. — L'application CANdroid est connectée au programme CANgateway. — La Raspberry Pi est connectée au bus CAN. — Tableau de Bord est connecté au bus CAN. — Un ou plusieurs objets existent. — Une ou plusieurs trames existent. — Une ou plusieurs trames sont en cours d'envoi.
Garanties minimales	N.A.
Garanties en cas de succès	— Aucune trame CAN n'est envoyée à Tableau de Bord.
Scénario nominal	1. Utilisateur demande à stopper l'envoi des trames. 2. L'application CANdroid affiche PopupArretEnvoi. 3. Utilisateur confirme l'arrêt de l'envoi des trames. 4. Le SàE arrête de diffuser les trames sur le bus CAN. 5. L'application CANdroid affiche EcranPrincipal.

Variantes	**3 [Utilisateur ne souhaite plus arrêter l'envoi des trames]** 3.a.1. Va en 5.
Extensions	N.A.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.3.12 CU Supprimer un élément

2.3.12.1 Description graphique

Voir figure 5.

2.3.12.2 Description textuelle

Titre	Supprimer un élément.
Résumé	Utilisateur utilise l'application CANdroid pour supprimer une trame ou un objet .
Portée	SàE.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré. — Un ou plusieurs objets existent. — Une ou plusieurs trames existent. — Aucune trame n'est en cours d'envoi.
Garanties minimales	N.A.
Garanties en cas de succès	— L'élément sélectionné est supprimé de l'application.
Scénario nominal	1. Utilisateur sélectionne un objet. 2. L'application CANdroid met à jour EcranPrincipal. 3. Utilisateur demande la suppression de la sélection. 4. L'application CANdroid affiche PopupSuppressionElement. 5. Utilisateur valide la suppression de la sélection. 6. L'application CANdroid met à jour EcranPrincipal.

Variantes	**1 [Utilisateur souhaite sélectionner une trame]** 1.a.1. Utilisateur déroule le menu d'un objet. 1.a.2. Le SàE met à jour EcranPrincipal. 1.a.3. Utilisateur sélectionne une trame. 1.a.4. Va en 2. **3 [Utilisateur souhaite sélectionner un autre élément]** 3.a.1. Va en 1. **5 [Utilisateur ne souhaite plus supprimer d'élément]** 5.a.1. Utilisateur annule la suppression d'élément. 5.a.2. Va en 6.
Extensions	N.A.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.3.13 CU Stopper le SàE

2.3.13.1 Description graphique

Voir figure 5.

2.3.13.2 Description textuelle

Titre	Stopper le SàE.
Résumé	Permet d'arrêter convenablement le SàE.
Portée	SàE.
Niveau	Utilisateur.
Acteurs directs	Utilisateur.
Acteurs indirects	N.A.
Préconditions	— Le SàE est démarré.
Garanties minimales	N.A.
Garanties en cas de succès	— Le SàE est arrêté.
Scénario nominal	1. Utilisateur quitte l'application CANdroid. 2. Utilisateur met hors tension la Raspberry Pi. 3. Utilisateur éteint le Simulateur ICSim.

Variantes	**2-3 [Uniquement application CANdroid en fonctionnement]** 2-3.a.1. Fin du CU. **3 [Utilisateur utilise le Simulateur et le Banc de test]** 3.a.1. Utilisateur éteint le Simulateur ICSim. 3.a.2. Utilisateur éteint le Banc de test. 3.a.3. Utilisateur débranche le Banc de test. 3.a.4. Fin du CU. **3 [Utilisateur utilise uniquement le Banc de test]** 3.b.1. Utilisateur éteint le Banc de test. 3.b.2. Utilisateur débranche le Banc de test. 3.b.3. Fin du CU. **3 [Aucun Tableau de Bord n'est allumé]** 3.c.1. Fin du CU.
Extensions	N.A.
Informations complémentaires	Cette partie sera développée lors de l'incrément 2.

2.4 Contraintes

2.4.1 Politiques réglementaires

Le code source ainsi que les conceptions de ce produit ne sont pas soumis à une obligation de confidentialité. Il n'y a aucune restriction ou politique réglementaire qui s'applique à ces informations.

2.4.2 Contraintes matérielles

Le produit est livré fonctionnel sur un Smartphone Samsung Galaxy A20e fonctionnant sous Android 9, ainsi que sur une Raspberry Pi 3B+. La connexion est fonctionnelle avec l'utilisation du module de communication CAN PCAN.USB IPEH-002021 175459 de la marque PEAK, ainsi qu'un module CAN RS485.

2.4.3 Exigences de fiabilité

L'équipe CANvengers s'engage à déployer les meilleurs efforts pour respecter les exigences suivantes :

— Les actionneurs doivent répondre en moins d'une seconde suite à une sollication via l'application CANdroid.

— Une information d'un capteur doit être affichée en moins d'une seconde sur l'application CANdroid.

Cela signifie que ces exigences seront respectées sous les conditions suivantes :

— Un seul client TCP doit être connecté au serveur.

— La connexion n'est pas brouillée.

2.4.4 Exigences de maintenabilité

L'application CANdroid est conçue pour être modulable, afin de pouvoir ajouter des capteurs et des actionneurs. Le code source doit également être lisible et commenté, afin de faciliter les évolutions futures.

2.4.5 Exigences de disponibilité

N.A.

3 Exigences spécifiques

3.1 Interfaces hommes machines

3.1.1 Généralités

Lorsque Utilisateur démarre l'application CANdroid, la connexion entre celle-ci et le programme CANgateway se fait automatiquement. Cependant, si la connexion échoue, Utilisateur est notifié et peut décider de se reconnecter.

Il peut ajouter ou supprimer des objets fictifs auxquels il peut ajouter des trames associées. Lors de l'ajout de la trame, Utilisateur doit choisir le mode d'envoi de la trame (ponctuel ou cyclique). Si la trame ne lui convient plus, il peut la supprimer. Utilisateur peut décider de commencer ou arrêter l'envoi des trames à Tableau de Bord.

Pour finir, l'application CANdroid est équipée d'un sniffer CAN que Utilisateur peut sauvegarder ou vider. Il peut également arrêter ou lancer la réception de trames.

3.1.2 Les actions utilisateurs

Avec l'application CANdroid, Utilisateur peut réaliser diverses actions :

- Se reconnecter au programme CANgateway.
- Démarrer et arrêter l'envoi de trames.
- Ajouter un objet.
- Supprimer un objet ou une trame.
- Déplier ou replier le menu de l'objet.
- Saisir une trame.
- Ajouter une trame.
- Arrêter ou relancer la réception des trames dans le sniffer.
- Revenir en haut du sniffer.
- Exporter les trames du sniffer.
- Supprimer les trames du sniffer.

La figure 6 montre le nom des boutons et des champs de texte de EcranPrincipal.

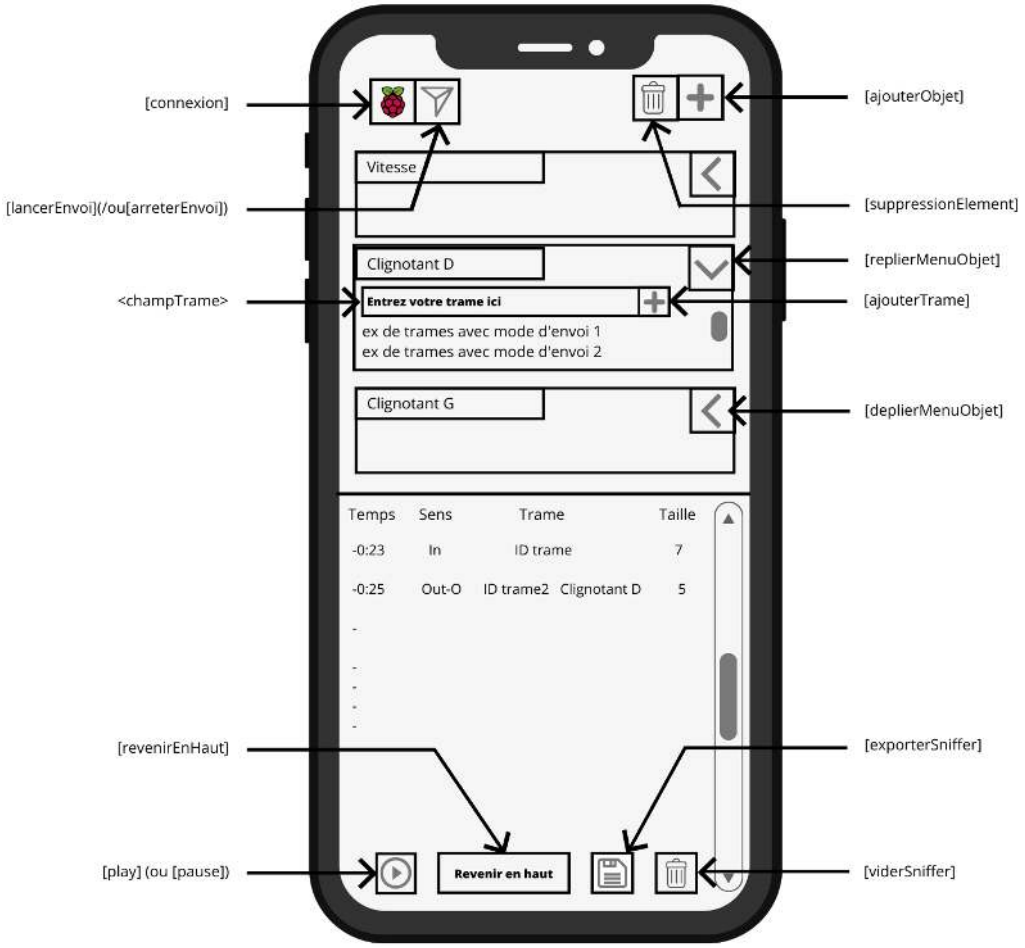

FIGURE 6 – EcranPrincipal avec légende des boutons et des champs

L'ensemble des figures suivantes représente les noms des boutons et des champs de texte de tous les différents pop-up utilisés dans l'application CANdroid.

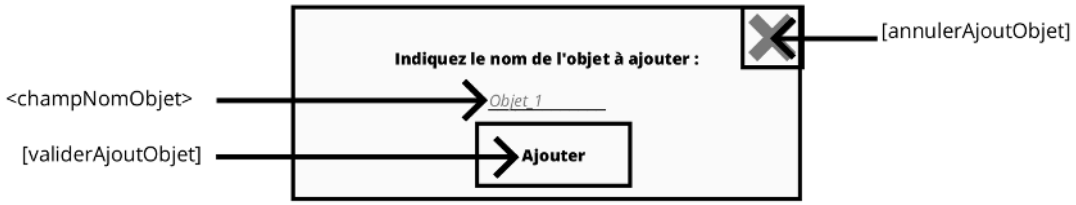

FIGURE 7 – PopupAjoutObjet avec nom des boutons et du champ de saisie

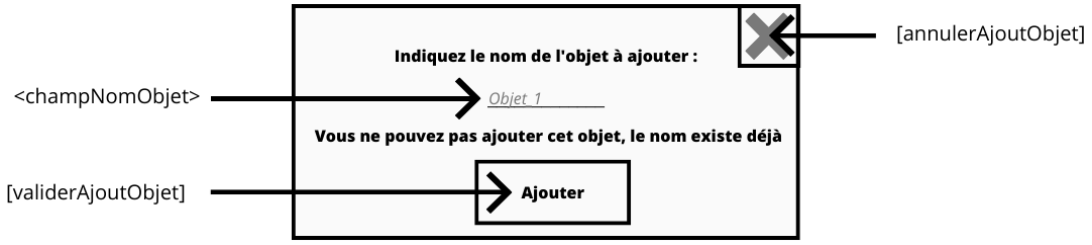

FIGURE 8 – PopupErreurAjoutObjet avec nom des boutons et du champ de saisie

FIGURE 9 – PopupErreurNombreObjet avec nom du bouton

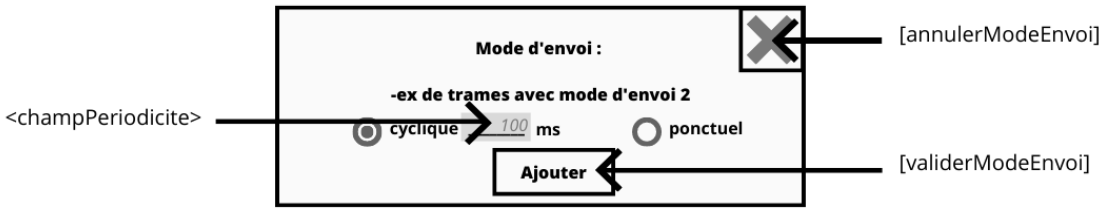

FIGURE 10 – PopupModeEnvoiTrame avec nom des boutons et du champ de saisie de la périodicité

FIGURE 11 – PopupErreurSaisieTrame avec nom du bouton

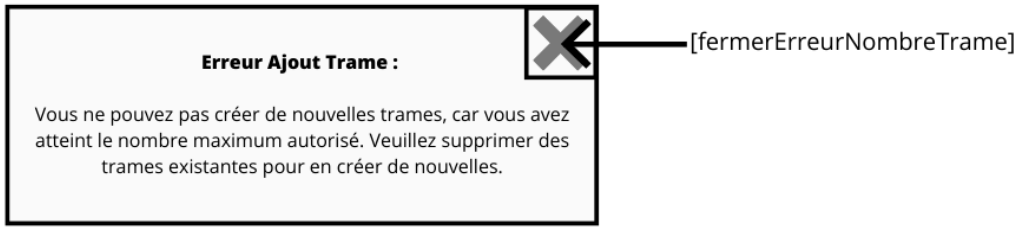

FIGURE 12 – PopupErreurNombreTrame avec nom du bouton

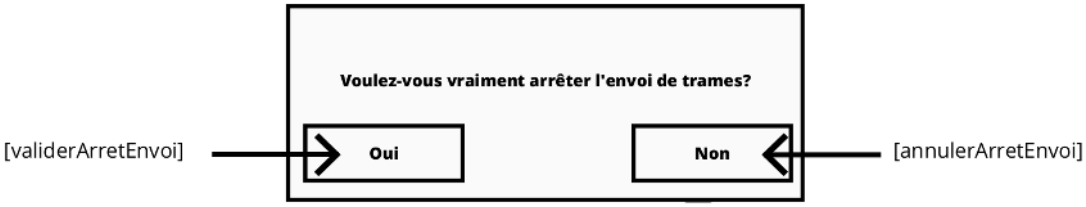

FIGURE 13 – PopupArretEnvoi avec nom des boutons

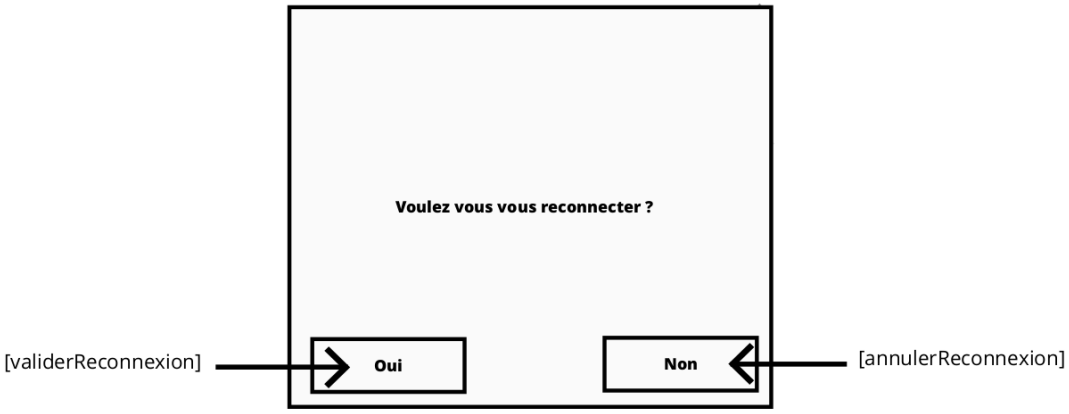

FIGURE 14 – PopupDemandeReconnexion avec nom des boutons

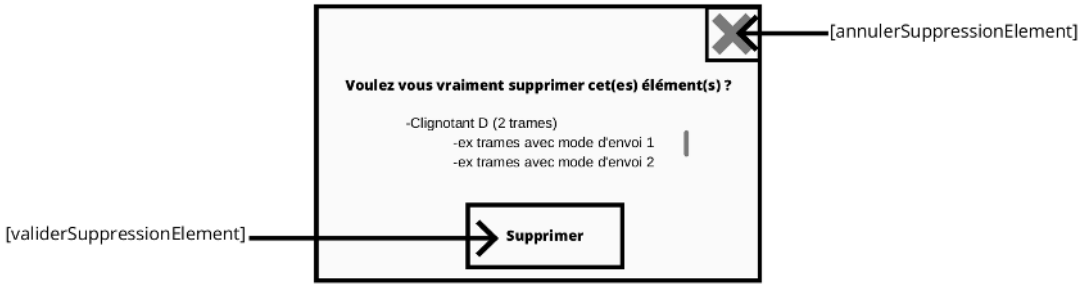

FIGURE 15 – PopupSuppressionElement avec nom des boutons

3.1.3 Les écrans

3.1.3.1 Vue générale

Utilisateur peut interagir avec Tableau de Bord via l'application CANdroid. L'application CANdroid peut transmettre des informations à Utilisateur par l'intermédiaire de l'écran du Smartphone.

La figure 16 représente les navigations possibles entre les différents écrans proposés par l'IHM. Ces enchaînements sont représentés par un diagramme d'états-transitions UML.
Chaque écran est représenté par un état (rectangle arrondi sur la figure). Les transitions indiquées par les flèches représentent une navigation d'un écran à l'autre en précisant l'événement logique du contexte (cf. chapitre 2.2.2.1) qui active la transition. Cela correspond généralement à des interactions de Utilisateur sur l'IHM qui génère l'événement correspondant.

Certaines transitions ne sont pas franchies sur des événements logiques, ce sont :

— Les événements de temporisation (le mot-clef « after » est alors noté sur la transition) : la transition doit être sensibilisée pendant une certaine durée pour être franchie.

— Des conditions qui deviennent vraies (la condition est alors exprimée entre crochets).

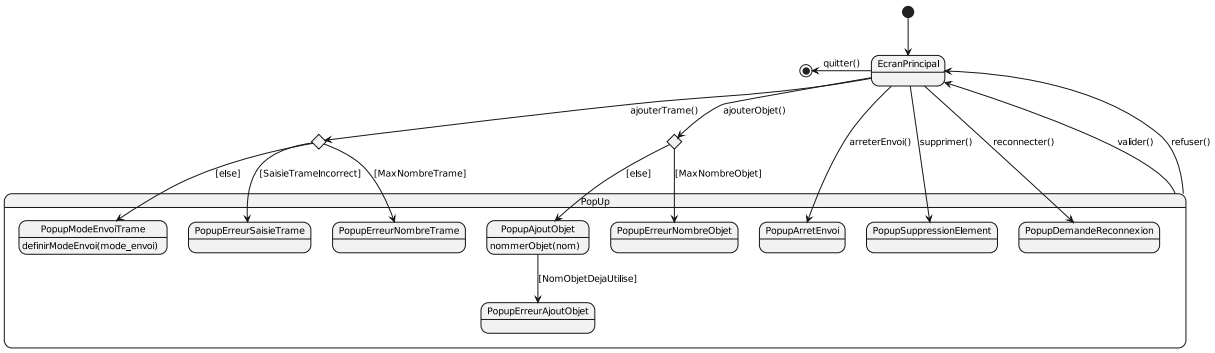

FIGURE 16 – Enchaînement entre les écrans de l'IHM représenté par un diagramme états-transitions UML

L'état initial de création de l'IHM est représenté par le disque de couleur noire. Après le lancement du système, l'IHM démarre avec l'écran "EcranPrincipal". Le disque de couleur noire entouré par un cercle correspond à la fin d'un enchaînement. Lorsqu'un enchaînement est fini, l'IHM s'éteint.

À partir de EcranPrincipal, Utilisateur peut réaliser différentes actions, qui déclencheront l'apparition de pop-up (afficherPopUp(id_popup)). Utilisateur peut à tout moment revenir à EcranPrincipal (refuser()).

Dans le cas d'un ajout d'objet (ajouterObjet()), l'IHM affiche PopupAjoutObjet (afficherPopUp(id_popup)). Utilisateur nomme l'objet (nommerObjet(nom)), il doit appuyer sur le bouton "Ajouter" nommé [validerAjoutObjet] dans la figure 7. Cela correspond à la fonction "valider()" de la figure 16. Si Utilisateur ne souhaite pas choisir de nom, l'application CANdroid en fournit un par défaut (voir section 3.2).

Dans le cas où Utilisateur saisit un nom d'objet qui est déjà utilisé par un autre objet, l'IHM affiche le pop-up d'erreur PopUpErreurAjoutObjet (afficherPopUp(id_popup)). Ce pop-up contient une phrase qui informe Utilisateur de la source de l'erreur. Tant que Utilisateur n'aura pas modifié le nom de l'objet, il ne pourra pas l'ajouter à la liste de l'application CANdroid.

Dans le cas d'un ajout de trame (ajouterTrame()), l'IHM affiche PopupModeEnvoiTrame (afficherPopUp(id_popup)) pour définir le mode d'envoi (definirModeEnvoi(mode_envoi)) de la trame à ajouter. Utilisateur doit appuyer sur le bouton "Ajouter" nommé [validerModeEnvoi] dans la figure 10. Cela correspond à la fonction "valider()" de la figure 16 qui permet à Utilisateur de confirmer son choix.

Dans le cas où Utilisateur saisit une trame qui ne correspond pas au format utilisé par l'application CANdroid, l'IHM affiche le pop-up d'erreur PopupErreurSaisieTrame (afficherPopUp(id_popup)). Pour fermer ce popup, Utilisateur doit appuyer sur le bouton [fermerErreurTrame]. Il pourra par la suite modifier le format de la trame saisie afin de pouvoir l'ajouter correctement.

Dans le cas où le nombre maximum de trames ou d'objets est atteint, l'IHM affiche respectivement les pop-ups d'erreur PopupErreurNombreTrame (afficherPopUp(id_popup)) et PopupErreurNombreObjet (afficherPopUp(id_popup)). Ces pop-ups contiennent un message d'erreur qui informe Utilisateur de la nature de l'erreur. Utilisateur peut simplement fermer le pop-up en cliquant sur la croix située en haut à droite. Cette action le ramènera sur EcranPrincipal. Dans la figure 12, la croix est identifiée par [fermerErreurNombreTrame], tandis que dans la figure 9 elle est identifiée par [fermerErreurNombreObjet].

Dans le cas d'une suppression d'élément (supprimer()), l'IHM affiche PopupSuppressionElement (afficherPopUp(id_popup)). Le pop-up affiche la liste des éléments sélectionnés, Utilisateur doit appuyer sur le bouton "Supprimer" nommé [validerSuppressionElement] dans la figure 15. Cela correspond à la fonction "valider()" de la figure 16 qui permet à Utilisateur de confirmer son choix.

Dans le cas d'un arrêt de l'envoi de trames (arreterEnvoiTrames()), l'IHM affiche PopupArretEnvoi (afficherPopUp(id_popup)). Utilisateur doit appuyer sur le bouton "Oui" nommé [validerArretEnvoi] dans la figure 13, ou sur le bouton "Non" nommé [annulerArretEnvoi] dans la même figure. Cela correspond respectivement aux fonctions "valider()" et "refuser()" de la figure 16 qui permet à Utilisateur de confirmer ou annuler son choix.

Dans le cas d'une perte de connexion, si Utilisateur souhaite se reconnecter (reconnecter()), l'IHM affiche PopupDemandeReconnexion(afficherPopUp(id_popup)). Utilisateur choisit de se reconnecter en appuyant sur le bouton "Oui" nommé [validerReconnexion] dans la figure 14, ou sur le bouton "Non" nommé [annulerReconnexion] dans la même figure. Cela correspond respectivement aux fonctions "valider()" et "refuser()" de la figure 16 qui permet à Utilisateur de confirmer ou annuler son choix.

3.1.3.2 EcranPrincipal

FIGURE 17 – Affichage général de EcranPrincipal

EcranPrincipal se décompose en deux parties. La partie haute permet de déclencher des interactions avec Tableau de Bord tandis que la partie basse correspond au sniffer CAN.

3.1.3.3 Partie haute de l'écran

La partie "envoi" (partie haute de l'écran) modère les interactions avec Tableau de Bord. On y observe l'état de la connexion au programme CANgateway. On peut ajouter ou supprimer des objets, et dans ces mêmes objets, ajouter ou supprimer des trames. Enfin, on peut y ordonner l'envoi de trames.

Connexion :

Lors du démarrage de l'application CANdroid, la connexion avec le programme CANgateway s'initialise automatiquement et continue en tâche de fond de l'application. Une connexion valide est représentée par l'icône de framboise (bouton [connexion]) en haut à gauche. Si la connexion ne s'est pas correctement déroulée, l'icône est grisée et barrée. Ainsi, au lancement de l'application CANdroid, l'icône est d'abord grisée et barrée (voir figure 18), puis, elle se colore si la connexion aboutit en moins de 3 secondes (voir figure 20). Si la connexion n'aboutit pas, Utilisateur peut demander à recommencer en cliquant sur le bouton [connexion] (dont l'icône est grisée et barrée). Un pop-up s'affiche et demande si l'on souhaite une reconnexion ou non au programme CANgateway (voir figure 19). L'application CANdroid peut être utilisée sans connexion. Dans ce cas, l'icône de framboise reste grisée et barrée. Le bouton [lancerEnvoi] est de couleur noire sur fond gris foncé pour indiquer qu'il est impossible d'envoyer des trames sur le bus CAN. De même pour le sniffer où le bouton [play] n'est pas cliquable. S'il est vide alors un message nous informe qu'il n'y a pas de trames sur le bus (voir figure 18).

FIGURE 18 – Affichage de EcranPrincipal non connecté

FIGURE 19 – Affichage de PopupDemandeReconnexion

Objet :

Les objets sont matérialisés sur l'application CANdroid par un rectangle (voir figure 17). Ce rectangle comprend un sous-rectangle avec le nom de l'objet en haut à gauche et, à droite, une flèche déroule le menu de l'objet (bouton [deplierMenuObjet] ou [replierMenuObjet] pour l'effet inverse) pour afficher les trames déjà ajoutées dans cet objet.

Les objets sont conservés d'une utilisation à l'autre sur l'application CANdroid. Ils sont triés du plus récent en haut au plus ancien. Lorsqu'il n'y a pas d'objet, un texte l'indique par un message d'information (voir figure 20).

FIGURE 20 – Affichage de EcranPrincipal sans objet

L'ajout d'objet se concrétise en cliquant sur l'icône "+" (bouton [ajouterObjet]) en haut à droite de l'écran. PopupAjoutObjet s'affiche alors pour que Utilisateur insère un nouveau nom d'objet (voir figure 21). Lorsque le nom est entré, Utilisateur clique sur "Ajouter" (bouton [validerAjoutObjet]) pour valider l'ajout de l'objet et l'afficher automatiquement dans la liste des objets présents sur l'application CANdroid.

Si Utilisateur ne remplit pas le champ du nom d'objet lors de la création d'un nouvel objet dans l'application CANdroid, celle-ci va automatiquement créer un objet avec un nom d'objet par défaut prérempli. Ce nom d'objet par défaut est visible en italique et en gris dans le champ nommé <champNomObjet> (voir section 3.2).

FIGURE 21 – Affichage de PopupAjoutObjet

Cependant, Utilisateur ne peut pas utiliser un nom déjà existant lorsqu'il ajoute un objet. Si le nom est déjà pris, un message d'erreur "Vous ne pouvez pas ajouter cet objet, le nom existe déjà" s'affiche (voir figure 22).

De plus, Utilisateur est limité dans le nombre d'objets qu'il peut créer sur l'application CANdroid. Si cette limite est atteinte et que Utilisateur souhaite ajouter un nouvel objet, une fenêtre d'erreur appelée PopupErreurNombreObjet s'affichera (voir figure 23).

FIGURE 22 – Affichage de PopupErreurAjoutObjet

FIGURE 23 – Affichage de PopupErreurNombreObjet

Pour supprimer un objet, Utilisateur clique sur l'objet, la case entière se grise puis Utilisateur clique sur la poubelle en haut à droite de EcranPrincipal (bouton [suppressionElement] sur la figure 24). PopupSuppressionElement s'affiche pour demander confirmation à Utilisateur. La suppression d'un objet entraîne la suppression des trames associées à cet objet. La liste des éléments sélectionnés à supprimer est indiquée sur le pop-up et Utilisateur clique sur "Supprimer" (bouton [validerSuppressionElement]) pour valider la suppression (voir figure 15).

FIGURE 24 – Affichage de EcranPrincipal avec l'objet "Clignotant D" sélectionné

FIGURE 25 – Affichage de PopupSuppressionElement

Trames :

Chaque trame est associée à un objet. Les trames sont visibles dans le rectangle de l'objet associé à la suite du nom de l'objet et dans l'ordre de leur ajout. Si possible, la trame est affichée dans sa globalité. Une trame possède un mode d'envoi : cyclique ou ponctuel. Pour visualiser la liste des trames, Utilisateur clique sur la flèche de l'objet (bouton [deplierMenuObjet]) pour dérouler le menu de l'objet et ainsi voir ses trames. Voir sur la figure 17 où le menu de l'objet "Clignotant D" est déroulé et on peut y voir ses trames, contrairement à l'objet "Vitesse".

FIGURE 26 – Affichage de EcranPrincipal pour la saisie de trames

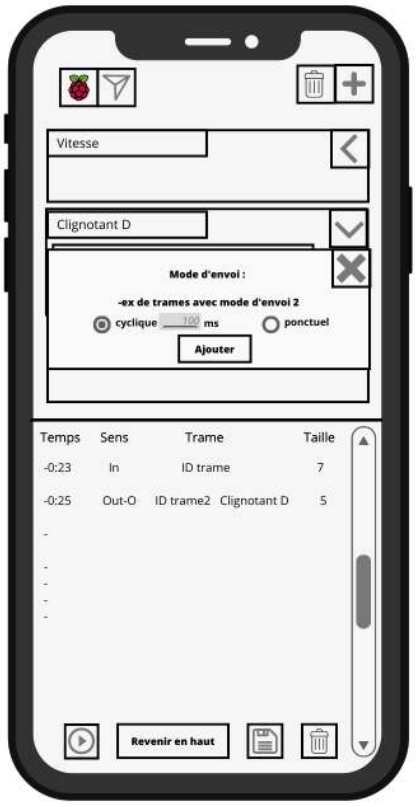

FIGURE 27 – Affichage de PopupModeEnvoiTram

Pour ajouter une nouvelle trame, Utilisateur se place dans l'objet auquel il veut l'associer. Il saisit la trame sous le format "#id$size@message" dans champ <champTrame>. C'est à Utilisateur d'écrire à la main les séparateurs. Cette syntaxe permet d'éviter à Utilisateur d'écrire les zéros non-significatifs. Utilisateur clique ensuite sur le bouton [ajouterTrame]. Si Utilisateur ne saisit pas la trame sous le bon format, PopupErreurSaisieTrame s'affiche (voir figure 28). Tant que Utilisateur ne saisit pas une trame sous le bon format, l'ajout ne pourra pas se faire.

Lorsque Utilisateur entre le bon format défini précédemment pour les trames, Popup-

ModeEnvoiTrame s'ouvre (voir figure 27) et demande à Utilisateur de choisir le mode d'envoi de la trame, pour cela, il doit cliquer sur le bouton radio correspondant au mode d'envoi souhaité. Un radio bouton est un élément qui permet à Utilisateur de sélectionner le mode d'envoi ponctuel (bouton [radioBoutonPonctuel]) ou le mode d'envoi cyclique (bouton [radioBoutonCyclique]). Lorsqu'un bouton radio est sélectionné, le cercle le définissant se remplit, et l'autre bouton radio est automatiquement désélectionné, le cercle le définissant se vide. Par défaut, une trame est périodique de périodicité 100ms mais Utilisateur peut la modifier grâce au champ <champPeriodicite>. Utilisateur clique sur "Ajouter" (bouton [validerModeEnvoi]) pour valider l'ajout de la trame (voir figure 27).

De la même manière que pour l'ajout d'objets, il existe également une limite pour le nombre de trames que Utilisateur peut ajouter. Si cette limite est atteinte et que Utilisateur tente d'ajouter une nouvelle trame, PopupErreurNombreTrame s'affiche (voir figure 29).

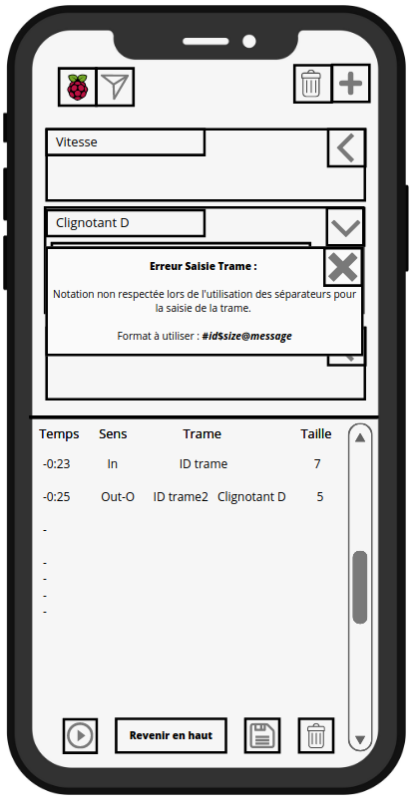

FIGURE 28 – Affichage de PopupErreurSaisieTrame

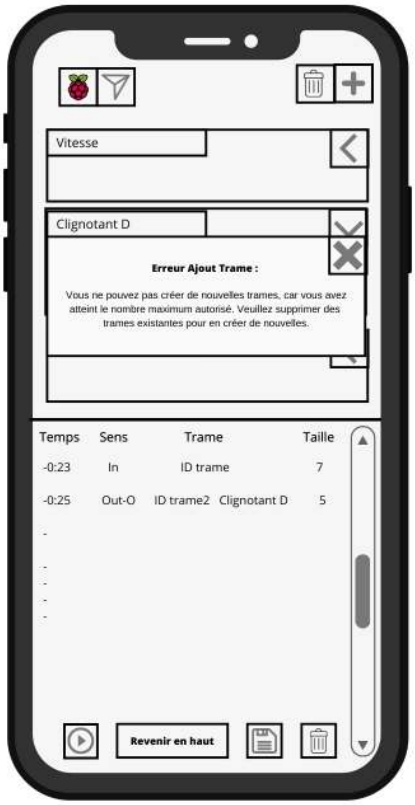

FIGURE 29 – Affichage de PopupErreurNombreTrame

Pour envoyer une trame, Utilisateur clique sur une trame à envoyer. S'il souhaite en envoyer plusieurs, la sélection est successive. Une trame est considérée comme sélectionnée si elle est grisée claire (voir figure 30). Si Utilisateur souhaite désélectionner une trame, il lui suffit de re-cliquer dessus. Lorsque la sélection est effectuée, Utilisateur clique sur l'icône d'envoi en haut à gauche de l'écran (bouton [lancerEnvoi]). Une trame en cours d'envoi se distingue par un surlignage gris foncé et l'écriture en blanc (voir figure 31). Le bouton d'envoi est également blanc sur fond gris foncé pour indiquer que des trames sont en cours d'envoi. Dans cet état-là, les boutons de suppression et d'ajout (trames comme objets) sont en noir sur fond gris foncé pour montrer que les outils correspondants sont désactivés.

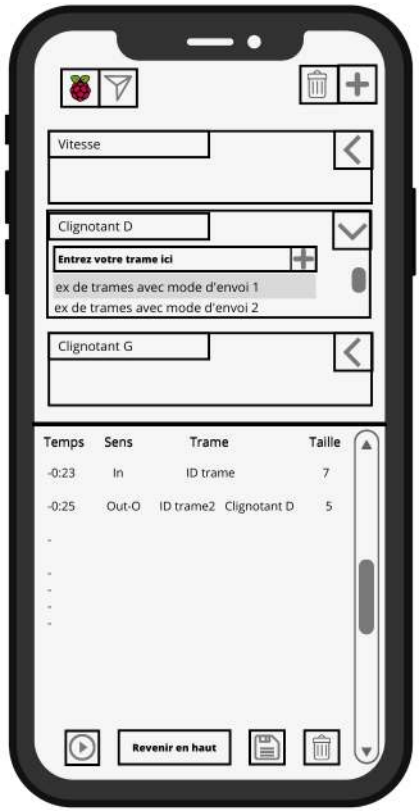

FIGURE 30 – Affichage de EcranPrincipal avec une trame de l'objet "Clignotant D" sélectionnée

FIGURE 31 – Affichage de EcranPrincipal lorsque des trames sont en cours d'envoi

Pour arrêter l'envoi des trames, Utilisateur clique à nouveau sur le bouton d'envoi (bouton [arreterEnvoi]), PopupArretEnvoi apparaît pour demander la confirmation à Utilisateur (voir figure 32). Utilisateur valide ce changement d'état en cliquant sur "Oui" (bouton [validerArretEnvoi]), dans le cas inverse Utilisateur clique sur "Non" (bouton [annulerArretEnvoi]) et le mode d'envoi continue.

FIGURE 32 – Affichage de PopupArretEnvoi

De même que pour supprimer un objet, pour supprimer une trame, Utilisateur clique sur la trame qu'il souhaite supprimer, la ligne se grise puis Utilisateur clique sur la poubelle (bouton [suppressionElement]) en haut à droite de EcranPrincipal (voir figure 30). La liste des éléments sélectionnés à supprimer est indiquée sur PopupSuppressionElement et Utilisateur clique sur "Supprimer" (bouton [validerSuppressionElement]) pour valider la suppression (voir figure 33).

FIGURE 33 – Affichage de PopupSuppressionElement
avec suppression de trames seulement

3.1.3.4 Partie basse de l'écran

La partie sniffer (partie basse de l'écran) présente les trames CAN envoyées et reçues par la Raspberry Pi lorsqu'il y en a. Lorsque la Raspberry Pi n'est pas encore connectée, un commentaire indique qu'il n'y a pas de trames actuellement "Pas de trame actuellement" (voir figure 20).
Les trames arrivent par le haut du terminal donc la trame la plus récente sera celle du haut. Pour chaque trame CAN, on précise la trame (id + donnée), l'objet auquel elle est associée (si la trame est envoyée), sa taille, son sens ("In" correspond à la réception sur l'application CANdroid et "Out" correspond à l'émission depuis l'application CANdroid) et un repère temporel. L'envoi d'une trame cyclique se remarque par l'ajout du symbole O dans son sens (Out-O). La trame est au format "#id$size@message" avec les caractères "#", "$" et "@" comme séparateurs.
Le repère temporel est en millisecondes, et est initialisé à zéro lorsque la première trame est sniffée sur le bus CAN.

FIGURE 34 – Affichage de EcranPrincipal avec le sniffer en mode "pause"

Utilisateur peut mettre en pause le sniffer en cliquant sur le bouton pause (bouton [pause]) en bas à gauche de l'écran (voir figure 34). Lorsque le sniffer est en pause, Utilisateur peut exporter les trames présentes dans le terminal vers un fichier de log, en cliquant sur l'icône "enregistrer" (bouton [exporterSniffer]) en bas à droite de l'écran. Il peut à la suite remettre le sniffer en écoute en cliquant sur l'icône play (bouton [play] sur la figure 34). Lorsque le sniffer est en cours de lecture, le bouton pour enregistrer le sniffer est verrouillé d'où le fait qu'il soit en noir sur fond gris foncé (voir figure 35).

FIGURE 35 – Affichage de EcranPrincipal avec le sniffer en mode "play"

L'enregistrement contient toutes les informations des trames visibles sur le sniffer. Les trames sont exportées dans un fichier de log (voir section 3.2).

Utilisateur peut revenir en haut de la liste des trames en cliquant sur le bouton [revenirEnHaut], et peut vider le sniffer des trames actuelles en cliquant sur la poubelle en bas de l'écran à droite sur le bouton [viderSniffer] (voir la figure 35).

3.2 Dictionnaire de domaine

— **Boutons :**

— [**ajouterObjet**] : Fait apparaître PopupAjoutObjet. Permet d'ajouter de nouveaux objets. Ce bouton fait appel à la fonction ajouterObjet() de la figure 5.

— [**ajouterTrame**] : Fait apparaître PopupModeEnvoiTrame. Permet d'ajouter une trame associée à un objet. Le bouton fait appel à la fonction ajouterTrame() de la figure 5.

— [**annulerAjoutObjet**] : Permet de sortir de PopupAjoutObjet.

— [**annulerArretEnvoi**] : Permet d'annuler l'arrêt d'envoi de trames et de sortir de PopupArretEnvoi.

— [**annulerModeEnvoi**] : Permet de sortir de PopupModeEnvoiTrame. Le nom de la trame reste cependant dans le champ de texte (<champTrame>, figure 6).

— [**annulerReconnexion**] : Permet d'annuler la reconnexion et quitter PopupDemandeReconnexion, de manière à utiliser l'application CANdroid en mode hors connexion.

— [**annulerSuppressionElement**] : Permet d'annuler la suppression des éléments sélectionnés et sortir de PopupSuppressionElement. Les éléments restent sélectionnés.

— [**arreterEnvoi**] : Permet l'arrêt de l'envoi des trames en cours d'envoi.

— [**connexion**] : Fait apparaître PopupDemandeReconnexion. Permet de reconnecter CANdroid et CANgateway.

— [**deplierMenuObjet**] : Permet de déplier le menu de l'objet. Ce bouton fait appel à la fonction ouvrirMenuObjet() de la figure 5.

— [**exporterSniffer**] : Permet d'exporter toutes les trames du sniffer dans un fichier de log. Ce bouton fait appel à la fonction exporterTramesSniffer() de la figure 5.

— [**fermerErreurTrame**] : Permet de sortir de PopupErreurSaisieTrame.

— [**fermerErreurNombreObjet**] : Permet de sortir de PopupErreurNombreObjet.

— [**fermerErreurNombreTrame**] : Permet de sortir de PopupErreurNombreTrame.

— [**lancerEnvoi**] : Permet d'envoyer les trames sélectionnées avec le mode ponctuel et de débuter l'envoi des trames sélectionnées avec le mode cyclique.

— [**play**]/[**pause**] : Met en marche ou en pause le sniffer. La mise en pause fait appel à la fonction desactiverReceptionTrames(). La reprise de la lecture fait appel à la fonction activerReceptionTrames() de la figure 5.

— [**replierMenuObjet**] : Permet de replier le menu de l'objet. Ce bouton fait appel à la fonction fermerMenuObjet() de la figure 5.

— [**revenirEnHaut**] : Permet de revenir en haut du sniffer. Ce bouton fait appel à la fonction revenirEnHaut() de la figure 5.

— [**suppressionElement**] : Fait apparaître PopupSuppressionElement. Permet de supprimer les éléments sélectionnés. Ce bouton fait appel à la fonction supprimer() de la figure 5.

- [**validerAjoutObjet**] : Permet de confirmer la création d'un nouvel objet et permet de sortir de PopupAjoutObjet.
- [**validerArretEnvoi**] : Permet de valider la demande d'arrêt d'envoi de trames et de sortir de PopupArretEnvoi. Ce bouton fait appel à la fonction arreterEnvoiTrames() de la figure 5.
- [**validerModeEnvoi**] : Permet de sauvegarder le mode d'envoi et de valider l'ajout de la nouvelle trame.
- [**validerSuppressionElement**] : Permet de valider la suppression des éléments sélectionnés et de sortir de PopupSuppressionElement.
- [**validerReconnexion**] : Permet de relancer une nouvelle procédure de reconnexion et de sortir de PopupDemandeReconnexion.
- [**viderSniffer**] : Permet de supprimer toutes les trames du sniffer. Ce bouton fait appel à la fonction supprimerTramesSniffer() de la figure 5.

- **CAN** : Controller Area Network, il s'agit d'un protocole de communication série utilisé pour connecter par exemple des capteurs et des actionneurs.

- **Champs de textes** :
  - **<champNomObjet>** : Ce champ de texte correspond à la saisie du nom de l'objet. Par défaut, ce champ est pré-rempli par le Nom d'objet par défaut. Il fait appel à la fonction nommerObjet() de la figure 5.
  - **<champPeriodicite>** : Ce champ de texte correspond à la saisie de la périodicité lorsque le mode cyclique est activé. Il fait appel à la fonction saisirPeriodicite() de la figure 5.
  - **<champTrame>** : Ce champ de texte correspond à la saisie de la trame sous le format souhaité. Il fait appel à la fonction ecrireTrame() de la figure 5.

- **Écrans utilisés** :
  - **EcranPrincipal** : Affichage principal de l'application. C'est sur cet écran que Utilisateur fait la majorité de ses interactions (ajouter un objet, supprimer une trame, consulter le sniffer, etc.).
  - **PopupAjoutObjet** : Pop-up d'ajout d'objet. Il sert à demander à Utilisateur le nom de l'objet qu'il souhaite ajouter.
  - **PopupArretEnvoi** : Pop-up de confirmation de demande d'arrêt d'envoi de trames.
  - **PopupDemandeReconnexion** : Pop-up de confirmation de demande de reconnexion entre CANdroid et CANgateway.
  - **PopupErreurAjoutObjet** : Pop-up d'ajout d'objet avec le message "Vous ne pouvez pas ajouter cet objet, le nom existe déjà" lorsque Utilisateur écrit un nom d'objet qui existe déjà.

  - **PopupErreurSaisieTrame** : Pop-up qui informe Utilisateur que le format qu'il a utilisé pour écrire la trame n'est pas le bon. PopupErreurSaisieTrame informe également sur le format à utiliser pour créer une trame.
  - **PopupErreurNombreObjet** : Pop-up pour informer Utilisateur qu'il a atteint le nombre maximum d'objets qu'il peut créer.
  - **PopupErreurNombreTrame** : Pop-up pour informer Utilisateur qu'il a atteint le nombre maximum de trames qu'il peut créer.
  - **PopupModeEnvoiTrame** : Pop-up de sélection du mode d'envoi de la trame à ajouter. Utilisateur peut choisir le mode ponctuel, ou le monde cyclique et saisir ou non la périodicité d'envoi de la trame.
  - **PopupSuppressionElement** : Pop-up de confirmation de suppression des éléments sélectionnés.

- **Fichier de logs** : fichier contenant les trames du sniffer, sauvegardé sur la Raspberry Pi. Le titre du fichier est de la forme trames_date_heure.log.
  - Date est sous la forme : JJMMAAAA (J correspond à jour, M correspond à mois, A correspond à années).
  - Heure est sous la forme : hhmm (h correspond à heure, m correspond à minutes).

- **La fin du fil** : correspond au trames reçues en dernier sur le sniffer de l'application CANdroid.

- **Nom d'objet par défaut** : Le nom par défaut est le nom donné lorsque Utilisateur ne spécifie pas de nom pour l'ajout d'un objet. Le nom par défaut sera "Objet_" suivi de l'identifiant le plus grand déjà utilisé pour un objet, augmenté de 1. Par exemple, si les identifiants des derniers objets créés sont 10, 11 et 12, le prochain objet créé aura le nom par défaut "Objet_13".

- **Format de la trame** : afin d'éviter à Utilisateur de taper tous les zéros non significatifs lors de la saisie d'une trame, les trames doivent être saisies avec des séparateurs de la forme suivante :
  - #id$size@message

- **PC (correspond à E_PC)** : Il s'agit d'un ordinateur fonctionnant sous Linux. Cet ordinateur dispose du SimulateurICSim installé.

- **Raspberry PI (correspond à E_Raspberry)** : Ordinateur monocarte créé par la Fondation Raspberry Pi.

- **RS485 CAN Hat** : D'après le site marchant du module (https ://www.waveshare.com/wiki/RS485_CAN_HAT), le RS485 CAN Hat permet à une Raspberry PI de communiquer avec d'autres appareils de manière stable

sur longue distance via les fonctions RS485/CAN. Dans notre cas d'utilisation, nous n'utilisons que la fonction CAN.

— **Smartphone (correspond à E_Smartphone)** : Téléphone portable sous système Android.

— **Sniffer** : Un sniffer est un programme qui capture tous les paquets circulant dans le réseau. Dans notre cas, le terme sniffer est utilisé pour désigner le terminal d'affichage des trames du réseaux CAN de l'application CANdroid.

— **Tableau de Bord** : représente l'un des (ou les) deux systèmes ci-dessous :
  — **Simulateur ICSim (correspond à E_ICSim)** : Simulateur d'un tableau de bord de voiture.
  — **Banc de test (correspond à E_Banc_De_Test)** : Banc de test d'un tableau de bord de voiture

  Il est connecté au SàE afin d'envoyer et recevoir des trames CAN. Dans tout le dossier, on emploie le terme Tableau de Bord (correspond à E_TableauDeBord) au singulier, car le scénario nominal du SàE n'utilise que le SimulateurICSim.

— **Types utilisés** :
  — **booléen** : est un type correspondant à un booléan, c'est-à-dire soit vrai (valeur non nulle), soit faux (valeur nulle).
  — **byte** : un ensemble de 8 bits, correspondant à l'unité de stockage d'un emplacement mémoire. C'est la plus petite unité adressable par un programme sur un ordinateur.
  — **string** : est un type correspondant à une chaîne de caractères.
  — **Id_objet** : identifiant des objets.
  — **Id_trame** : identifiant des trames (selon leur ordre de création, donc différent de l'ID de la trame CAN).
  — **Id_popup** : identifiant du pop-up choisit (cf. liste des écran utilisés).
  — **Informations** : informations visuelles visibles sur Tableau de Bord. Cela peut être un clignotant, la vitesse de la voiture, etc.
  — **Trame** : tableau de 12 bytes.

