Note de cadrage

I) Introduction

Dans le cadre de notre formation en IUT informatique, nous travaillons avec l'entreprise TMM groupe sur un projet de client SNMP sous Android.

L'entreprise TMM est le leader en France dans le domaine des logiciels de santé à usage des soignants et des patients. Elle est implantée partout en France mais aussi à l'étranger, comme en Belgique ou en Allemagne. TMM crée des solutions d'accompagnement et de divertissement pour le milieu hospitalier disponibles sur différents supports (tablettes numériques, box TV…), sous le système d'exploitation Android.

Dans le cadre de l'amélioration de leurs produits et de la capacité à intervenir rapidement en cas de problème sur leur parc grandissant l'entreprise souhaite développer un service Android utilisant le protocole SNMP pour envoyer à un serveur Centreon des « TRAP » (messages d'erreur du protocole SNMP) indiquant que le dispositif rencontre un problème technique.

Le service devra récolter un maximum d'informations utiles concernant le problème technique et les transmettre au serveur Centreon afin de trouver la cause du problème (lié à l'utilisateur, au logiciel ou au matériel).

II) La portée et les limites de l'étude

- Les utilisateurs

L'application étant autonome elle ne possédera pas d'utilisateur physique quand elle fonctionnera sur le parc. En revanche, lorsque nous testerons l'application sur les différents terminaux de TMM, nous utiliserons une interface graphique de débogage. Les utilisateurs avec lesquels l'application va communiquer seront donc:

- Le serveur Centreon: il est autonome, sont but est de traiter les TRAP que les terminaux clients enverront.
- L'API Android : son rôle est de permettre l'accès aux informations à propos des composants physiques mais aussi logiciels de la tablette afin de déceler des problèmes techniques.
- L'API de l'application TMM : c'est l'interface de l'application qui nous permettra de vérifier l'état de ses modules (téléphone…).
- Le débogueur : c'est l'acteur utilisant l'interface, qui ne sera pas accessible dans la version finale, qui permettra de déboguer l'application sur les différents terminaux de l'entreprise (lancer manuellement les tests, envoyer manuellement des TRAP).

- **Objectifs utilisateurs :**
  - Analyser le terminal pour détecter des problèmes :
    - de type hardware
    - de type software
  - Envoyer des données au serveur :
    - préparer des TRAP
    - émettre des TRAP
  - Déboguer l'application

Les charges utilisateurs

- Gérer le débogage :
  - lancer manuellement les différentes analyses traitées par le service et retourner le succès ou l'échec du test
  - permettre d'envoyer manuellement le résultat du test dans une TRAP.

Les charges du système

- Gérer l'analyse du terminal : l'application devra être autonome pour savoir si elle est en mode ROOT et ne pas analyser trop souvent le matériel ce qui causerait un ralentissement de la tablette.
- Gérer l'analyse du Hardware : l'application devra utiliser l'API Android pour vérifier si le matériel est opérationnel et ne présente pas d'utilisation anormale.
- Gérer l'analyse du software : l'application devra déceler les dysfonctionnements possibles au niveau logiciel sans en créer de nouveaux.
- Gérer la préparation de TRAP : l'application devra mettre en forme les TRAP pour contenir l'identification du problème et des informations le concernant.
- Gérer l'envoi de TRAP : l'application devra envoyer les TRAP du protocole SNMP en utilisant le réseau du terminal.

- Le contenu

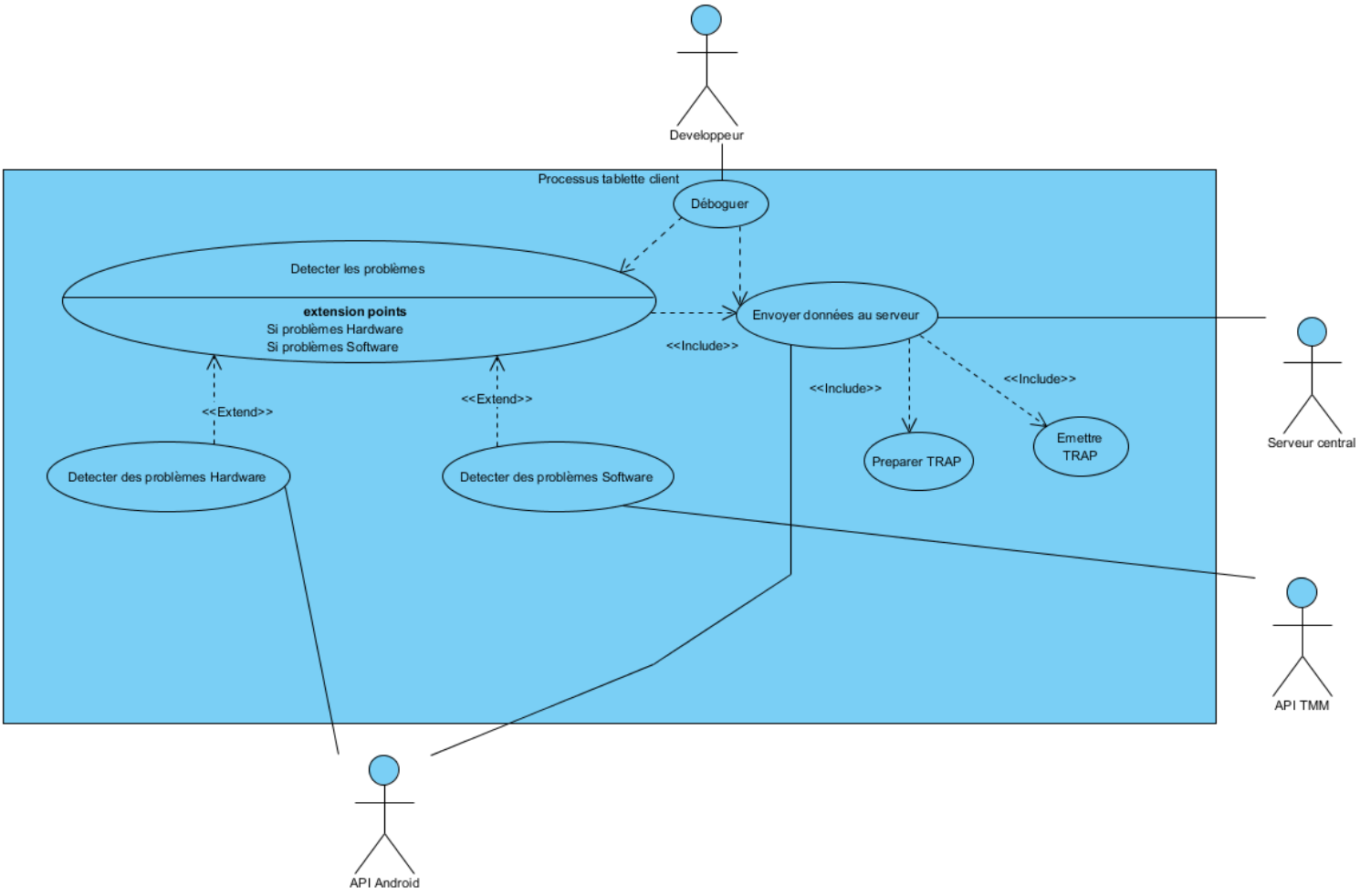
Pour le bon déroulement de l'application principale de la tablette cliente, nous allons devoir créer un processus invisible, léger portatif sur différents matériels et architectures.

Son rôle sera d'analyser la tablette et de détecter des problèmes qui peuvent être tout aussi bien hardware (surchauffe du processeur, utilisation anormale de la RAM, arrêt inopiné de la tablette) que software (fermeture de l'application principale, dysfonctionnement de certaines applications comme le téléphone ou la télécommande). Quand l'une des ces erreurs sera rencontrée, notre application enverra une TRAP SNMP au serveur central qui sera géré par l'entreprise TMM et qui interviendra aussitôt.

Notre application sera développée en JAVA et utilisera le protocole SNMP.

Dans le cadre du bon développement de notre application, nous allons devoir créer une surcouche qui va posséder une interface graphique et nous permettra de déboguer notre programme. Ainsi nous pourrons voir étape par étape si notre application principale (qui ne possédera en version finale aucune interface) réagira correctement aux attentes que nous aurons fixées et si elle pourra accéder aux API nécessaires pour les tests, dans l'éventualité où nous ne serions pas ROOT.

- Le diagramme de cas d'utilisation



- Détail des cas d'utilisations

Détecter les problèmes :

- Préconditions : la tablette est lancée, l'application TMM est lancée, notre application est lancée.
- Scénario : nous effectuons une vérification pour savoir si l'application est lancée en mode ROOT. Puis nous lançons de manière récurrente les test à effectuer en fonction du mode ROOT. Lorsqu'une erreur est détectée nous appelons l'« envoi des données au serveur » avec les informations collectées sur l'origine du problème.
- Postconditions : le service est fermé, la tablette s'éteint.

Détecter des problèmes Hardware :

- Préconditions : le cas « Détecter les problèmes » nous envoie une demande de vérification hardware.
- Scénario : nous recherchons un problème de type Hardware grâce à l'API Android (état de la RAM, du CPU, de la ROM et du réseau). Si nous découvrons un problème matériel, nous collectons les informations à son propos puis nous les mettons en forme pour les remonter.
- Postconditions : les informations collectées sont remontées, la vérification est terminée.

Détecter des problèmes Software :

- Préconditions : le cas « Détecter les problèmes » nous envoie une demande de vérification software.
- Scénario : nous recherchons un problème de type Software grâce à l'API TMM (état des modules de TMM, application TMM toujours allumée, état de l'application TMM), si nous découvrons un problème software, nous collectons les informations à son propos puis nous les mettons en forme pour les remonter.
- Postconditions : les informations collectées sont remontées, la vérification est terminée.

Envoyer données au serveur :

- Paramètres : un ensemble d'informations relatives au problème rencontré.
- Préconditions : un problème a été détecté et une demande d'envoi de données a été initiée
- Scénario : nous lançons la préparation d'une TRAP avec nos données, puis nous envoyons la TRAP retournée.
- Postconditions : nous savons si la TRAP a été envoyée ou non.

Préparer TRAP :

- Paramètres : un ensemble d'informations relatives au problème rencontré.
- Préconditions : un problème a été détecté, les informations relatives au problème ont été transmises
- Scénario : nous mettons les informations en forme pour qu'elles soient contenues dans le format d'une TRAP. Nous retournons cette TRAP.
- Postconditions : la TRAP est prête et retournée.

Émettre TRAP :

- Paramètre : une TRAP SNMP.
- Préconditions : une TRAP a été crée, l'application demande l'envoi de la TRAP au serveur central.
- Scénario : nous émettons la TRAP. Si elle l'envoi est correct nous retournons un « succès », le cas échéant nous retournons un « échec ».
- Postconditions : le résultat a été retourné.

Déboguer :

- Préconditions : notre application est lancée en mode « débogage ».
- Scénario : lancement manuel des vérifications, affichage des résultats : vérification effectuée et son résultat ou non effectuée et la cause. Préparation et envoi d'une TRAP SNMP au serveur central.
- Postconditions : débogage terminé, application arrêtée ou passée en mode « service ».

- Les risques

- L'accès au ROOT : certains terminaux utilisés par TMM ne sont pas accessible en mode ROOT, ce qui peut empêcher d'accéder à certaines fonctionnalités de l'API d'Android ou à des processus.
- Fermeture de notre application : en cas de fermeture de notre application pour une raison quelconque, la tablette cliente ne pourra plus émettre de TRAP SNMP et donc déceler des problèmes sur le terminal.
- Application trop lourde : si c'est le cas elle ralentira la tablette et empêchera le bon fonctionnement de l'application TMM.
- Installation universelle : le parc de terminaux géré par TMM étant très larges et composé de différentes versions d'Android, marques et d'architecture de processeurs. Il faudra donc créer une application universelle.
- Serveur inaccessible : en cas d'une coupure du serveur principal pour une maintenance ou pour une autre raison, des TRAP SNMP pourront être perdues.
- Coupure de réseau : en cas d'une coupure du réseau, nous devons être capable à la reprise de la connexion, de détecter les problèmes rencontrés entre temps.