



Haute école d'ingénierie et d'architecture Fribourg  
Hochschule für Technik und Architektur Freiburg

***TIC – Filière Informatique***

**Projet de semestre 6**

**2018**

# **Rapport**

---

## **S.A.S**

**Développement d'une application Web de surveillance  
et de soutien contre l'addiction en ligne**

---



Logo

**Nicolas Fuchs  
Grégory Ducrey**

---

Superviseurs : **Dr Sandy Ingram**

**Dr Houda Chabbi**

Mandant : **Dr. Claire Korkmaz**

**Dr. Sariah Haddad**

---

Fribourg, Mars 2018

## Sommaire

<b>Historique des versions .....</b>	<b>4</b>
<b>1. Introduction .....</b>	<b>5</b>
1.1. Contexte .....	5
1.2. But du projet .....	6
1.3. Document.....	6
<b>2. Analyse des besoins et des technologies .....</b>	<b>7</b>
2.1. Introduction .....	7
2.2. Idéation .....	8
2.3. Etat de l'art.....	9
2.3.1. Self control .....	10
2.3.2. Contrôle parental Mac os .....	11
2.3.3. Rescue Time .....	12
2.3.4. Moment .....	13
2.3.5. Contrôle parental Swisscom .....	14
2.3.6. Internet Security Swisscom.....	15
2.3.7. Synthèse.....	16
2.4. Tests technologiques.....	17
2.4.1. Analyse des processus .....	17
2.4.2. Analyse des sites web visités .....	19
2.5. Conclusion .....	22
<b>3. Conception .....</b>	<b>23</b>
3.1. Introduction .....	23
3.2. Cas d'utilisation .....	24
3.2.1. Se logger.....	25
3.2.2. Enregistrer les activités desktop et web.....	26
3.2.3. Générer un fichier de log .....	27
3.2.4. Afficher la liste des utilisateurs .....	28
3.2.5. Consulter les statistiques .....	29
3.2.6. Créer un objectif.....	30
3.2.7. Ajouter un utilisateur.....	31
3.2.8. Supprimer un utilisateur.....	32
3.2.9. Accepter le contrôle .....	33
3.2.10. Consulter la barre d'utilisation.....	34
3.2.11. Consulter ses statistiques .....	35
3.3. Structure du fichier Json.....	36
3.4. Conclusion .....	37
<b>4. Implémentation .....</b>	<b>38</b>
4.1. Introduction .....	38
4.2. Prototype 1.....	39
4.3. Prototype 2.....	40
4.4. Prototype 3.....	41
4.5. Conclusion .....	42

<b>5. Validation.....</b>	<b>43</b>
5.1. Introduction .....	43
5.2. Conclusion .....	44
<b>6. Conclusion.....</b>	<b>44</b>
6.1. Conclusions de groupe .....	44
6.2. Conclusions personnelles .....	44
6.2.1. Grégory Ducrey .....	44
6.2.2. Nicolas Fuchs .....	44
<b>7. Annexes .....</b>	<b>44</b>
7.1. Déclaration d'honneur .....	44
7.2. Bibliographie .....	44

## Historique des versions

Cet historique indique à quelles dates ont eu lieu les modifications du cahier des charges :

Version	Semaine	Date	Personnes	Chapitres	Modification
0.1	P3	05.03.18	G. Ducrey N. Fuchs	1, 1.1, 1.2, 1.3	Création de la table des matières, Introduction
0.2	P4	12.03.18	G. Ducrey N. Fuchs	2, 2.4	Introduction à l'analyse, état de l'art et synthèse
0.3	P5	19.03.18	G. Ducrey N. Fuchs	2.4, 2.5	Etat de l'art et tests technologiques
0.4	P6	29.03.18	G. Ducrey N. Fuchs	3	Conception
0.5	P7	09.04.2018	G. Ducrey N. Fuchs	3	Conception
0.6	P8	16.04.2018	G. Ducrey N. Fuchs	3	Correction de la doc selon les retours
0.7	P9	23.04.2018	G. Ducrey N. Fuchs	3,4	Fin de la conception et début de l'implémentation

# 1. Introduction

Ce chapitre présente les aspects suivants du projet :

- Le contexte et motivations de réalisation du projet
- Les objectifs
- Le document

## 1.1. Contexte

Selon le monitoring suisse de l'addiction, un peu plus de 7% des jeunes Suisses entre 15 et 19 ans utilisent le Web de manière compulsive et problématique (addiction aux jeux de rôle en ligne, jeux d'argent, négligence des devoirs et des activités hors-ligne). L'objectif du projet est de concevoir et développer une application minimisant l'effet des stimuli addictifs, et redonnant le contrôle aux "consommateurs". Pour cela, un monitoring automatique des activités du consommateur (avec son consentement) doit être mise en place suivi de mesures de sensibilisation proposées en temps-réel.

Pour détecter si une personne est dépendante aux jeux vidéos, il y a plusieurs critères qui sont les suivants selon wikipédia [1] :

- Monopolisation de la pensée par le projet de comportement addictif
- Intensité et durée des épisodes plus importants que souhaités à l'origine
- Tentatives répétées pour réduire, contrôler ou abandonner le comportement
- Temps important consacré à préparer les épisodes, à les entreprendre ou à s'en remettre
- L'engagement dans le comportement est tel que la personne ne peut plus accomplir des gestes élémentaires (se laver, se nourrir) et le conduit vers un désinvestissement social, professionnel et familial
- Poursuite du comportement malgré l'aggravation des problèmes sociaux et en dépit de la connaissance des conséquences négatives
- Tolérance marquée, c'est-à-dire besoin d'augmenter l'intensité ou la fréquence pour obtenir l'effet désiré, ou diminution de l'effet procuré par un comportement de même intensité
- Agitation, irritabilité et surtout angoisse si le passage à l'acte addictif est différé, empêché

On constate que pour ces gens-là, les conséquences néfastes peuvent être les suivantes :

- Appauvrissement de la vie sociale
- Manque de sommeil
- Trouble du comportement (saute d'humeur...)
- Mauvaise hygiène de vie
- Perte du sens des réalités
- Atteinte à la santé physique (maux de tête, yeux secs, douleurs au dos, manque d'activité physique, dépression, suicide)

Ces liens contiennent quelques témoignages qui permettent de comprendre l'importance de la gestion des stimuli addictifs pour les joueurs. En effet, il n'est pas néfaste en soit de jouer, mais il est absolument nécessaire de garder le contrôle.

<http://www.jeuxvideo.com/dossiers/00018270/l-addiction-aux-jeux-video-temoignages-de-joueurs-006.htm>[2]

<https://www.vice.com/fr/article/ppvzd7/dependance-jeux-video-843>[3]

## **1.2. But du projet**

Le projet consiste à développer un produit permettant aux utilisateurs de reprendre le contrôle lorsqu'ils sont soumis à des stimuli addictifs quand ils utilisent des jeux vidéos ou des réseaux sociaux.

L'enfant ou l'adolescent est encore en construction et il n'a pas encore tous les filtres lui permettant de comprendre les enjeux du temps passé sur un écran. C'est pourquoi cette application permettra de mettre un cadre où le joueur sera confronté à une responsabilisation de ses choix. A chaque âge le cadre doit être différent. Il sera donc plus restrictif pour des enfants que des adolescents.

Ce travail est réalisé dans le cadre du projet de semestre 6 en informatique à la haute école d'ingénieurs et d'architectes de Fribourg. Il est réalisé par M.Nicolas Fuchs et M.Grégory Ducrey pour la mandante Dr. Claire Korkmaz et il est supervisé par Mme Sandy Ingram ainsi que Mme Houda Chabbi.

## **1.3. Document**

Ce rapport fait partie intégrante du projet. Il contient des informations concernant les étapes de réalisation du produit final. La création de ce document se fait tout au long du projet. Il doit permettre au client de suivre les démarches de travail et si le projet devait être repris, le rapport donne toute les clés au développeur. Le document contient des informations sur les phases suivantes :

- Analyse
- Conception
- Implémentation
- Validation

Le rapport permet aux évaluateurs de vérifier la qualité et la rigueur du travail effectué.

## 2. Analyse des besoins et des technologies

### 2.1. Introduction

Nous ne connaissons pas énormément le sujet de la prévention des addictions aux jeux vidéos et aux réseaux sociaux, c'est pourquoi il est important d'étudier la base de ce sujet avant de réfléchir aux développements d'un produit pertinent.

Le client souhaite une application qui permet de monitorer et redonner le contrôle à l'utilisateur afin que ce dernier fasse des choix et en prenne l'entière responsabilité. Dans le chapitre 1.1 traitant du contexte du projet, des témoignages sont recueillis et ils attestent des dangers de ces addictions. En observant les critères d'addictions décrits, nous pouvons constater que le danger découle du temps passé sur les jeux ou les réseaux sociaux. C'est pourquoi nous avons en partie orienté nos recherches sur des moyens pour gérer le temps.

Nos recherches doivent nous aider à créer un produit répondant aux besoins de manière pertinente. Dans cette analyse nous avons effectué les travaux suivants :

- Exploration du sujet → se renseigner sur le contexte, des statistiques
- Etat de l'art → Explorer les produits de prévention présents sur le marché
- Test de produits existants → Tester certains de ces produits
- Tests technologiques → Tester certaines méthodes qui pourraient nous aider à implémenter des fonctionnalités (Récupération des processus, récupération des URL...)
- Idéation → Sur la base du travail accompli, faire un brainstorming de ce qui serait pertinent d'implémenter pour ce produit
- Cahier des charges → après ces 5 semaines d'analyse, établir un cahier des charges final décrivant le produit que nous allons réaliser en ayant une meilleure connaissance du sujet

## 2.2. Idéation

Pour commencer ce projet, nous nous sommes dans un premier temps renseignés sur le sujet des addictions (voir chapitre « 1 introduction ») afin d'avoir une meilleure vue d'ensemble. Une fois fait, nous avons effectué un brainstorming [4] des idées de fonctionnalités qui pourraient être intéressantes lors du développement de l'application. Ce brainstorming est effectué avant les tests d'applications existantes et de tests technologiques. Il a pour but de laisser aller notre créativité sans être limité par des recherches déjà effectuées.

Voici les résultats du brainstorming, nous avons classé les idées par catégories :

### Concept de l'application

- Contrôle des sites visités
- Gestion du temps
- Architecture client/serveur et paramétrisation de l'application native depuis la page web du serveur
- Système de récompense
- Contrôle parental
- Encouragement à la mobilité physique
- Blocage de certains processus en fonction d'une localisation GPS
- Validation par l'utilisateur

### Fonctionnalités concrètes

- Monitoring du temps
  - Paramétrer en fonction de l'âge
- Module statistique —> pour l'utilisateur et/ou contrôleur
  - Temps passé sur une semaine, et sur quel sujet
- Alerte visuelles et sonores ET Avertissement sur les dangers
  - Forcer des pauses
  - Vidéo de sensibilisation
  - Blocage du processus durant la vidéo
  - L'utilisateur est prévenu de l'interruption
- Filtrer les sites visités (avec une liste, ou avec un thème)
- Vérification de l'USB (Si le jeux est lancé par disque)

### Moyens techniques

- Récupération des processus
- Chronométrage du temps d'activité des processus
- Interruption de processus
- Récupération d'adresse IP et de localisation

### Plateforme

- Application mobile multiplateforme
- Natif mac os
- Natif PC
- Natif Linux
- Jeux vidéos WEB

### Moyens de prévention

- Liste des sites/processus sensibles a permettre/interdire
- Jeux



### 2.3. Etat de l'art


Nous avons exploré plusieurs applications existantes permettant de mieux gérer le temps passé sur l'ordinateur ou le smartphone. Le but de cette observation est de se rendre compte de ce qui existe sur le marché et réfléchir à des fonctionnalités que nous pourrions apporter en plus. Pour pouvoir analyser correctement, nous avons examiné les aspects suivants :

- Le type de blocage
- Orienté web/natif/mobile → Si l'application travaille sur les navigateurs, sur les mobiles ou en natif sur l'ordinateur. Il est possible qu'une application native monitore des navigateurs.
- La plateforme sur laquelle le logiciel fonctionne
- Base volontaire ou restrictions → est-ce que l'application a tendance à interdire ou à sensibiliser
- Public cible → quel type de personne est susceptible d'utiliser le logiciel ?
- Paramétrer en fonction de l'âge → Est-ce possible de changer le monitoring ou les restrictions en fonction de l'âge de l'utilisateur ?
- Redonne le contrôle → est-ce que l'utilisateur peut décider ce qu'il fait ?
- Difficulté de prise en main → critère subjectif
- Administration depuis la même machine
- Fournit des statistiques d'utilisation
- Avis des utilisateurs


Ainsi, sur la base de ces observations nous pouvons comparer différentes applications et nous rendre compte de l'utiliter des fonctionnalités pour notre client. Nous ne souhaitons pas plagier mais nous inspirer de ce qui existe déjà et créer un produit adapté aux besoins de nos clients.

Les applications testées proviennent des sites [5] à [12] de la bibliographie.


## 2.3.1. Self control

 <b>Self control [6]</b>	
<b>Utilisation</b>	
L'utilisateur choisit les sites webs ou les services mail à bloquer. Il choisit également la durée. Une fois fait il active la restriction et tant que le temps n'est pas écoulé c'est impossible d'accéder à ce qui est bloqué. (même en cas de suppression de l'application où de redémarrage).	
<b>Critère</b>	<b>Observation</b>
Blocage	Bloque les sites web (black list ou white list). Bloc les mails. Ne bloque pas des applications
Orienté web/natif/mobile	Application native orientée web
Plateforme	Fonctionne sur Mac OS
Base volontaire ou restrictions	L'utilisateur décide lui même ce qu'il autorise ou non
Public cible	Personnes ayant la maturité de savoir ce qui est bon pour lui ou non. L'administration depuis une autre machine n'est pas possible
Paramétrer en fonction de l'âge	Il est possible d'adater la liste des sites à bannir.
Redonne le contrôle	Redonne le contrôle au bout d'un certain temps
Difficulté de prise en main	Très facile à utiliser
Administration depuis la même machine	Il faut activer le logiciel sur la session à bloquer
Fournit des statistiques d'utilisation	Non, il ne fournit pas de statistiques
Avis des utilisateurs	Bon logiciel
Payant	Le logiciel est gratuit et opensource
Opensource	Oui
Offline	Est inutile en hors ligne puisqu'il monitore uniquement ce qui est en ligne
<b>Remarque complémentaire</b>	
L'outil est vraiment très simple. Il serait certainement utile pour se couper des distractions durant une journée de travail par exemple. L'utilisateur doit avoir une certaine maturité. <b>On a accès au code de l'application</b>	


## 2.3.2. Contrôle parental Mac os

	
<b>Contrôle parental Mac os [7]</b>	
<b>Utilisation</b>	
L'utilisateur de la session administrateur donne des droits aux utilisateurs des autres sessions en activant le contrôle parental.	
Critère	Observation
Blocage	Assignment de plages horaires et de durée d'utilisation de l'ordinateur. Black-List/white-List) de sites web et limitation des sites web pour adultes. Blocage d'applications et de stores
Orienté web/natif/mobile	Application native
Plateforme	Fonctionne sur Mac OS
Base volontaire ou restrictions	Ne fonctionne que sur la base de restrictions. L'utilisateur de la session contrôlée ne peut pas changer quoi que ce soit aux restrictions, il subit des interdictions.
Public cible	Les enfants particulièrement. Ce serait plutôt sur un ordinateur familial où il y a plusieurs sessions que le contrôle parental serait pertinent
Paramétrer en fonction de l'âge	Tout est adaptable en fonction de ce que l'administrateur souhaite pour l'utilisateur
Redonne le contrôle	Ne donne aucun contrôle. L'application interdit mais n'apprend pas à gérer
Difficulté de prise en main	Simple d'utilisation, mais beaucoup d'options
Administration depuis la même machine	Les restrictions sont élaborées depuis la session administrateur de la machine
Fournit des statistiques d'utilisation	Il conserve les historiques de consultation de site webs
Avis des utilisateurs	Assez bon
Payant	Le logiciel est compris dans le prix de l'OS
Opensource	Non
Offline	Le contrôle parental fonctionne avec ou sans connexion internet
<b>Remarque complémentaire</b>	
Ce logiciel est très solide et permet de bien gérer d'un point de vue administrateur. Mais il ne permet pas à l'utilisateur de prendre conscience du temps qu'il passe sur telle ou telle application. L'idéal serait qu'il sorte des statistiques d'utilisations et qu'il permette des alertes sans créer de blocage	


## 2.3.3. Rescue Time

 RescueTime [8]	
Utilisation	
L'utilisateur installe l'application sur son appareil. Ensuite il paramètre la surveillance sur la page web. L'application répertorie tout le temps passé sur chaque application par l'utilisateur. Elle permet de fixer des objectifs et en fonction de ceux-ci, elle affiche des alertes. L'application ne bloque rien	
Critère	Observation
Blocage	Ne bloque rien, fournit des statistiques
Orienté web/natif/mobile	Application native dont les statistiques sont disponibles sur une page web
Plateforme	Fonctionne sur Mac os, Linux, Windows et Android. Une version iOS est en cours de développement
Base volontaire ou restrictions	Base volontaire. Le "dashboard" ne bloque rien, il permet de fixer des objectifs donne des alertes. Ex: Objectif, <2h sur facebook. Si le temps est atteint, une alerte approche
Public cible	Tout public
Paramétrer en fonction de l'âge	Oui, le parent pourrait fixer des objectifs pour l'enfant, et ce dernier ne verrait que les alertes.
Redonne le contrôle	Il redonne le contrôle grâce aux alertes, cela met l'utilisateur devant un choix.
Difficulté de prise en main	Il y a quelques options à décider, et il faut indiquer pour certain sites web s'ils sont sources de distraction ou non
Administration depuis la même machine	L'application est installée sur une machine, et on peut voir les statistiques sur une page web.
Fournit des statistiques d'utilisation	Donne des statistiques à la seconde près sur le temps passé sur les pages web, les applications.
Avis des utilisateurs	Très bon logiciel
Payant	Parties gratuites et payantes
Open source	Non
Offline	Offline, l'utilisation n'est pas monitorée. Sauf en partie payante
Remarque complémentaire	
<p>Cette application correspond beaucoup à ce que l'on souhaite faire : Redonner le contrôle.</p> <p><b>Une API est disponible</b> pour travailler avec les données récupérées par ce dashboard. Nous allons explorer les possibilités dans le chapitre « 2.5 tests technologiques ». La partie payante de l'application permet l'utilisation d'un bracelet paramétrable pour envoyer des décharges électriques en cas d'utilisation nocive d'application (à gérer en fonction des objectifs) [9]. Le nom de l'app est « Pavlok ». Plusieurs autres plugin utilisent les données.</p>	


## 2.3.4. Moment

	
<b>Moment [10]</b>	
<b>Utilisation</b>	
L'utilisateur télécharge l'application sur son produit iOS. Il l'ouvre et il autorise les accès par l'app aux photos ainsi qu'à la localisation. Une fois fait, il doit encore activer le traçage des apps utilisées.	
<b>Critère</b>	<b>Observation</b>
Blocage	Ne bloque rien, monitore
Orienté web/natif/mobile	Application native pour iOS. Monitore les applications, non pas les sites web.
Plateforme	iOS
Base volontaire ou restrictions	Base volontaire. Ce n'est que du monitoring. Pas de blocage.
Public cible	Toute personne utilisant un produit avec iOS
Paramétrer en fonction de l'âge	Non
Redonne le contrôle	Oui, car il peut donner des alertes
Difficulté de prise en main	Très simple
Administration depuis la même machine	Les statistiques apparaissent sur le mobile
Fournit des statistiques d'utilisation	Oui
Avis des utilisateurs	Bonne application (4/5 sur l'apple store)
Payant	Gratuit
Opensource	Non
Offline	L'utilisation est la même offline
<b>Remarque complémentaire</b>	
L'application affiche le temps passé par app. Et elle donne un fil rouge pour la journée. Cependant elle n'est pas capable de filtrer le contenu web. Par contre, étant donné que les réseaux sociaux ont une app en général, c'est tout de même intéressant si les gens l'utilisent. Le produit est très semblable à Rescue Time, sauf qu'il est créé pour iOS. Une version Android est en cours de développement. L'application compte le nombre de fois que l'on déverrouille le smartphone.	

## 2.3.5. Contrôle parental Swisscom

 <b>swisscom [11]</b>	
Fonctionnement	
<p>L'utilisateur choisit les appareils sur lesquels il veut activer le contrôle parental. Celui-ci consiste dans cette version standard à limiter les heures de navigation sur internet individuellement. Un créneau horaire peut être défini selon les jours de la semaine et du weekend ainsi que le temps total. Ce contrôle est paramétrable sur la swisscom TV box. Les émissions contenant des limites d'âge peuvent également être bloquées par un PIN que seules les personnes autorisées connaissent.</p>	
Critère	Observation
Blocage	Bloque la navigation sur internet et certains contenus
Orienté web/natif/mobile	Orienté web
Plateforme	Fonctionne pour tout ordinateur, tablette et smartphone pouvant se connecter à internet
Base volontaire ou restrictions	Ne fonctionne que sur la base de restrictions. L'utilisateur du réseau contrôlé ne peut pas changer quoi que ce soit aux restrictions, il subit des interdictions.
Public cible	Les enfants particulièrement et les adolescents
Paramétrer en fonction de l'âge	Navigation internet -> Pas directement. Si l'enfant utilise toujours les mêmes appareils, alors le contrôle est indirectement associé à l'âge. Blocage de contenu lié à l'âge -> Oui
Redonne le contrôle	Ne donne aucun contrôle. Le service interdit mais n'apprend pas à gérer
Difficulté de prise en main	Très facile à utiliser
Administration depuis la même machine	La gateway est paramétrée via une interface web donc administrable depuis n'importe quelle machine
Fournit des statistiques d'utilisation	Non, il ne fournit pas de statistiques
Avis des utilisateurs	Aucun avis disponible
Payant	La fonctionnalité est comprise dans l'abonnement swisscom
Opensource	Non
Offline	Est inutile en hors ligne puisqu'il monitore uniquement ce qui est en ligne
Remarque complémentaire	
<p>Ce contrôle parental est facile d'utilisation. Par contre, il ne permet pas de filtrer spécifiquement le contenu (mis à part le blocage du contenu limité par l'âge). Internet security permet un contrôle beaucoup plus approfondi, c'est le prochain service analysé.</p>	

## 2.3.6. Internet Security Swisscom

 <b>swisscom [12]</b>	
Fonctionnement	
<p>L'utilisateur installe cette application sur la machine que l'enfant ou l'adolescent utilise depuis le compte administrateur. L'application définit le contenu non-accessible par catégories et par sites spécifiques. Elle permet aussi de définir les horaires de navigation sur internet et d'accès à la machine même.</p>	
Critère	Observation
Blocage	Bloque l'accès à l'ordinateur, les heures de navigation sur internet et certains contenus
Orienté web/natif/mobile	Application native
Plateforme	Fonctionne sur ordinateur, tablette et smartphone (fonctionnalités différentes)
Base volontaire ou restrictions	Ne fonctionne que sur la base de restrictions. L'utilisateur du réseau contrôlé ne peut pas changer quoi que ce soit aux restrictions, il subit des interdictions.
Public cible	Les enfants particulièrement et les adolescents
Paramétrer en fonction de l'âge	Non
Redonne le contrôle	Ne donne aucun contrôle. L'application interdit mais n'apprend pas à gérer
Difficulté de prise en main	Facile à utiliser
Administration depuis la même machine	Les restrictions sont élaborées depuis la session administrateur de la machine
Fournit des statistiques d'utilisation	Non, il ne fournit pas de statistiques
Avis des utilisateurs	Bonne application (4.4/5 sur google play)
Payant	La fonctionnalité est gratuite pendant 6 mois et ensuite payante
Opensource	Non
Offline	Utile car restreint également l'accès à la machine et pas seulement internet.
Remarque complémentaire	
<p>Cette application est également très utile pour la protection des données et contre les sites webs dangereux ainsi que les virus.</p>	

### 2.3.7. Synthèse

Durant ces recherches nous avons pu constater que les applications existantes permettent de réaliser déjà un bon monitoring ou contrôle par rapport à ce que fait l'utilisateur. Nous trouvons des produits qui interdisent l'accès à des ressources, d'autres qui informent l'utilisateur de son utilisation.

La plupart de ces produits sont gratuits, voir même opensource. L'application « Rescue Time » nous semble être celle qui est la plus complète. Elle permet dans la version gratuite de :

- Sortir des statistiques d'utilisation des applications desktop ou encore des pages web visitées (rapports journaliers) sur une page web
- Créer des alertes
- Classer les occupations par niveaux de productivité (mail = productif, Spotify != distraction), selon l'appréciation de l'utilisateur

La version payante [13] permet de :

- Monitorer le temps Offline
- Bloquer des sites web pendant une durée déterminée (comme self control)
- Avoir des rapports d'utilisations plus détaillés
- Créer des alertes personnalisées

Le tout pour 9\$ par mois ou 72\$ par année.

Nous trouvons que cette application est puissante complète et efficace. Nous allons nous inspirer des fonctions suivantes pour notre projet :

- Récupération du temps que l'utilisateur passe sur chacune de ses activités
- Création de graphiques d'utilisation
- Création d'objectifs et d'alertes personnalisées

Voici un tableau semblable à ceux créés pour les technologiques mais il est rempli ce qui se caractérise l'application SAS :

SAS	
Fonctionnement	
Le superviseur envoie une invitation de monitoring à l'utilisateur. Ce dernier accepte l'invitation. Le logiciel récupère les informations des activités de l'utilisateur. Le superviseur et l'utilisateur peuvent consulter ces informations. Le superviseur peut mettre un temps limite d'utilisation, et l'utilisateur verra une barre passant du vert au rouge évoluer en fonction du temps.	
Critère	Observation
Blocage	Ne bloque pas
Orienté web/natif/mobile	Fonctionne comme application desktop
Plateforme	Windows, Mac OS
Base volontaire ou restrictions	Base volontaire
Public cible	Enfant et adolescents
Paramétrer en fonction de l'âge	Oui, possibilité de créer des objectifs adaptés à l'âge (barre d'utilisation)
Redonne le contrôle	Oui
Administration depuis la même machine	Non, c'est possible depuis une autre machine
Fournit des statistiques d'utilisation	Oui
Offline	Non



## 2.4. Tests technologiques

Cette section présente les différents tests technologiques menés lors de la phase d'analyse afin de déterminer les fonctionnalités techniquement faisables. La première partie concerne les processus tournant sur la machine de l'utilisateur du système. La seconde partie se penche sur la récupération du trafic internet. Deux petits programmes écrits en langage java ont été développés afin d'effectuer ces tests technologiques. Trois aspects différents des programmes sont mentionnés dans ce contexte : la compatibilité des devices et systèmes d'exploitation, les informations récupérées par le programme et l'utilisation d'éventuelles bibliothèques.

### 2.4.1. Analyse des processus

Le premier test implémente la récupération d'une liste des processus tournant sur un ordinateur portable. Concrètement pour le projet, les noms d'exécutables ainsi que leur chemin sur la machine sont analysés pour connaître la nature des applications qui tournent sur la machine (jeux, réseaux sociaux,...).

#### Compatibilité des devices et systèmes d'exploitation

Le programme a été testé et fonctionne sur les systèmes d'exploitation suivants :

- Windows 10
- Mac OSX
- Kali Linux Rolling

Une liste des processus peut également être récupérée sur des appareils mobiles, notamment avec le langage C#. Ce langage est utilisé par Xamarin pour le développement d'application cross-plateform. Sur Android, on peut aussi utiliser le langage java. En ce qui concerne IOS, l'accès aux informations système semble plus complexe car à partir de la version IOS 9, l'accès aux données des applications est très restreint.

En estimant que la récupération de la liste des processus sur IOS prendrait trop de ressources par rapport à la durée du projet et qu'un appareil android peut aussi obtenir la liste des processus avec du java ou du c# (très similaire à java), il n'a pas été jugé nécessaire d'implémenter des tests technologiques sur un téléphone mobile.

#### Versions/Utilisation de librairie

Deux façons de récupérer les processus ont été implémentées.

La première solution consiste à lancer une commande système depuis le programme java. Cependant, cette solution n'est pas générique. Il faut donc créer deux versions différentes : Une version pour windows et une autre pour les systèmes UNIX (Mac OSX et Linux). Le programme de test étant petit, la gestion des OS (Operating System) est gérée au travers de conditions dans le code. Dans le cas où cette option est choisie dans le cadre de ce projet, le code doit être séparé en plusieurs fichiers, un fichier relatif à un système d'exploitation.

La deuxième solution utilise une librairie java nommé *oshi* qui fournit aux développeurs des informations sur le système d'exploitation. Cette solution est moins rapide mais offre des options de tri intéressantes de même que d'autres données importantes liées aux processus détectés, telles que le processID ou le chemin de l'exécutable par exemple. La figure ci-dessous affiche les deux résultats obtenus.

*****		*****
First way	Première solution	Second way
services.exe		Process name : System Idle Process
svchost.exe		Path : unknown
MySQLNotifier.exe		ProcessID : 0
Telegram.exe		ParentProcessID : 0
acrotray.exe		
chrome.exe		Process name : javaw.exe
...		Path : C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe
explorer.exe		ProcessID : 1152
SearchUI.exe		ParentProcessID : 10912
chrome.exe		...
eclipse.exe		Process name : eclipse.exe
javaw.exe		Path : C:\Users\Nicolas\eclipse\java-mars\eclipse\eclipse.exe
tasklist.exe		ProcessID : 10912
		ParentProcessID : 1884
Total number of processes : 192		Total number of processes : 187

Résultats de la récupération des processus []

La figure ci-dessous montre le contenu du fichier ListProcesses.java contenant les deux solutions du test.

```
package test;

import oshi.SystemInfo;
import oshi.software.os.OSProcess;
import oshi.software.os.OperatingSystem;

import java.io.*;
import java.util.regex.*;

public class ListProcesses {

    public static void main(String[] args) {
        try {

            String username = System.getProperty("user.name");
            System.out.println("Username detected : " + username);

            SystemInfo si = new SystemInfo();
            OperatingSystem os = (OperatingSystem) si.getOperatingSystem();

            Process p = null;
            String operatingSystem = os.getFamily();
            System.out.println(operatingSystem + " OS detected !\n");
            System.out.println("*****");
            System.out.println("First way\n");

            if (operatingSystem.equals("Windows")) {
                p = Runtime.getRuntime().exec("tasklist.exe");
            } else if (operatingSystem.equals("MacOS") || operatingSystem.equals("Kali GNU/Linux")) {
                p = Runtime.getRuntime().exec("ps -eo comm"); //another command is "top" on mac (live update)
            }

            BufferedReader input = new BufferedReader(new InputStreamReader(p.getInputStream()));
            String line = ""; int firstCounter = 0;

            if (operatingSystem.equals("Windows")) {
                Pattern pattern = Pattern.compile(".*.exe");
                while((line = input.readLine()) != null) {
                    Matcher matcher = pattern.matcher(line);
                    if (matcher.find()) System.out.println(matcher.group(0));
                    firstCounter++;
                }
            } else if (operatingSystem.equals("MacOS") || operatingSystem.equals("Kali GNU/Linux")) {
                while((line = input.readLine()) != null) {
                    System.out.println(line);
                    firstCounter++;
                }
            }

            System.out.println("\n Total number of processes : " + firstCounter);
            System.out.println("*****");
            System.out.println("Second way\n");

            OSProcess[] processes = os.getProcesses(0, OperatingSystem.ProcessSort.CPU);
            int secondCounter = 0;
            PrintWriter pwn = new PrintWriter(new FileWriter(new File("Processes_Names_list.txt")));
            PrintWriter pwp = new PrintWriter(new FileWriter(new File("Processes_Pathes_list.txt")));
            for (int i = 0; i < processes.length; i++) {
                System.out.println("Process name : " + processes[i].getName());
                System.out.println("Path : " + processes[i].getPath());
                System.out.println("ProcessID : " + processes[i].getProcessID());
                System.out.println("ParentProcessID : " + processes[i].getParentProcessID());
                System.out.println();
                pwn.write(processes[i].getName());
                pwp.write(processes[i].getPath());
                secondCounter++;
            }
            pwn.close();
            pwp.close();
            System.out.println("\n Total number of processes : " + secondCounter);

        } catch (Exception e) {
            System.out.println("No command line opened");
            e.printStackTrace();
        }
    }
}
```

Première solution

Deuxième solution

Code de la récupération des processus []

### Informations récupérées

Comme cité plus haut dans le sous-chapitre précédent, la quantité et la forme des informations récupérées sont dépendantes de l'utilisation d'une librairie et du système d'exploitation. Les informations récupérables sont les suivantes : Name, Path, ProcessID et parentProcessID. Comme on peut le voir dans le code de la première solution de la figure 2, Windows 10 requiert un parsing pour ne récupérer que le nom de l'exécutable et non le chemin complet.

### 2.4.2. Analyse des sites web visités

Le deuxième test implémente la récupération d'une liste des sites web sur un ordinateur portable. Concrètement pour le projet, les sites web avec leurs métadonnées et leur temps de visite sont analysés pour distinguer les catégories des sites visités (jeux en ligne, réseaux sociaux,...).

#### *Compatibilité des devices et systèmes d'exploitation*

Le programme (première solution) a été testé et fonctionne sur le système d'exploitation Windows 10. Les autres systèmes d'exploitation (Linux et Mac) et devices n'ont pas été testés car cette solution a vite été considérée comme inadaptée pour ce projet. La deuxième solution utilise un navigateur internet donc elle est générique au niveau des systèmes d'exploitation des ordinateurs portables. Elle a également été testée sur windows 10. Pour ce qui est des téléphones mobiles, cette solution ne fonctionne que sur le navigateur Firefox sur un appareil Android. Estimant que le trafic web ne valant pas la peine d'être analysé sur des smartphones, aucun test technologique n'a été fait dans ce sens.

#### *Versions/Utilisation de librairie*

Deux façons de récupérer les sites web ont été implémentées.

La première solution utilise la librairie libpcap (utilisée par l'outil Wireshark) enrobée dans un wrapper Java. Il en existe plusieurs sortes plus ou moins efficaces et faciles d'intégration dans un projet. La première librairie testée se nomme pcap4j. Elle n'est malheureusement pas utile pour ce projet car les protocoles que l'on désire capturer sont HTTP et HTTPS de la couche 7 du modèle OSI. Ces protocoles ne sont pas supportés par cette librairie. La seconde librairie testée se nomme jnetpcap. L'inconvénient est qu'elle est capable de capturer les protocoles HTTP et HTTPS, un protocole sécurisé et chiffré. Cela signifie que le protocole HTTPS est capturable mais non-exploitable. C'est pour cette raison principalement que cette solution n'a pas été envisagée pour la suite du projet. De plus, la manière de capturer le trafic web a été remise en cause. Lors de la capture de paquets sur le réseau, on obtient uniquement les entêtes HTTP mais aucune informations concernant les onglets sur lesquels l'utilisateur est actif ainsi que le temps passé sur ceux-ci individuellement. Grâce aux applications Rescue Time et le contrôle parental de Swisscom, on a pu découvrir une autre façon énormément plus efficace pour contrôler les sites sur lesquels navigue l'utilisateur. Cette alternative est la deuxième solution décrite en dessous.

Cette deuxième méthode fonctionne grâce à une extension Google Chrome, composée de fichiers HTML, CSS et JS. Cette extension est également développable sur d'autres navigateurs comme Firefox par exemple. L'extension a accès directement aux onglets du navigateur ainsi qu'aux URLs entrées. Le temps aussi peut être analysé. A chaque changement/ouverture/fermeture d'onglet, l'extension récupère le temps passé sur l'onglet précédent et envoie ces informations à un serveur HTTP local (dans ce test, codé en Java).

```

package test;

import java.util.ArrayList;
import java.util.List;
import org.jnetpcap.Pcap;
import org.jnetpcap.PcapIf;
import org.jnetpcap.packet.PcapPacket;
import org.jnetpcap.packet.PcapPacketHandler;
import org.jnetpcap.protocol.tcpip.Http;
import org.jnetpcap.protocol.tcpip.Http.Request;
import org.jnetpcap.protocol.tcpip.Tcp;

public class SniffNetwork {

    public static void main (String[] args) {
        List<PcapIf> alldevs = new ArrayList<PcapIf>(); // List of all capture interfaces
        StringBuilder errbuff = new StringBuilder(); // Used for error messages

        int statusDevs = Pcap.findAllDevs(alldevs, errbuff);
        if (statusDevs == Pcap.NOT_OK || alldevs.isEmpty()) {
            System.err.println("Network devices list issue");
            return;
        }

        for (PcapIf device : alldevs) {
            System.out.println("Name : " + device.getName());
            System.out.println("Description : " + device.getDescription() + "\n");
        }

        PcapIf device = alldevs.get(2);

        int snaplen = 64 * 1024; // Capture all packets, no truncation
        int flags = Pcap.MODE_PROMISCUOUS; // Capture all packets
        int timeout = 10 * 1000; // 10 seconds in milliseconds

        Pcap pcap = Pcap.openLive(device.getName(), snaplen, flags, timeout, errbuff); // Open the device for sniffing
        if (pcap == null) {
            System.err.println("Opening sniffing device issue : " + errbuff);
            return;
        }

        PcapPacketHandler<String> packethandler = new PcapPacketHandler<String>() {
            private final Tcp tcp = new Tcp();
            private final Http http = new Http();
            @Override
            public void nextPacket(PcapPacket packet, String user) {
                if (!packet.hasHeader(tcp) || !packet.hasHeader(http) || http.isResponse()) return;
                String host = http.fieldValue(Request.HOST);
                if (host != null) System.out.println("Host : " + host);
            }
        };

        pcap.loop(-1, packethandler, "jNetPcap rocks!");
        pcap.close();
    }
}

```

Première solution de sniffing web []

```

package test;

import java.io.*;
import com.sun.net.httpserver.HttpExchange;
import com.sun.net.httpserver.HttpHandler;

public class SniffHandler implements HttpHandler {

    @Override
    public void handle(HttpExchange he) throws IOException {
        InputStreamReader isr = new InputStreamReader(he.getRequestBody(), "utf-8");
        BufferedReader br = new BufferedReader(isr);
        String line = br.readLine();
        System.out.println(line);

        String response = "OK";
        he.sendResponseHeaders(200, response.length());
        OutputStream os = he.getResponseBody();
        os.write(response.toString().getBytes());
        os.close();
    }
}

```

Deuxième solution de sniffing web - SniffHandler.java []

```

package test;

import java.io.IOException;
import java.net.InetSocketAddress;
import com.sun.net.httpserver.HttpServer;

public class ExtensionServer {

    public static void main(String[] args) {
        int port = 19119;
        HttpServer server;
        try {
            server = HttpServer.create(new InetSocketAddress(port), 0);
            System.out.println("server started at " + port);
            server.createContext("/sniff", new SniffHandler());
            server.setExecutor(null);
            server.start();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

Deuxième solution de sniffing web - ExtensionServer.java []

```

var port = 19119;
var previousURL = "";
var startTime = Date.now() / 1000;
var endTime = Date.now() / 1000;

chrome.tabs.onUpdated.addListener(function(tabId, changeInfo, tab) {
    endTime = Date.now() / 1000;
    if (typeof tab !== undefined && changeInfo.url !== undefined) {
        if (previousURL !== "" && previousURL !== "chrome://newtab/") {
            sendTimedURL();
        }
        previousURL = changeInfo.url;
        startTime = endTime;
    }
});

chrome.tabs.onActivated.addListener(function(activeInfo) {
    endTime = Date.now() / 1000;
    chrome.tabs.get(activeInfo.tabId, function(tab) {
        if (previousURL !== "" && previousURL !== "chrome://newtab/") {
            sendTimedURL();
        }
        previousURL = tab.url;
        startTime = endTime;
    });
});

function sendTimedURL() {
    var http = new XMLHttpRequest();
    var description = getMetaContentByName('description');
    var TimedUrl = previousURL + " " + (endTime - startTime) + " " + description;
    http.open("POST", "http://localhost:" + port + "/sniff", false);
    http.setRequestHeader("Content-type", "text/plain; charset=utf-8");
    http.setRequestHeader("Content-length", TimedUrl.length);
    http.setRequestHeader("Connection", "close");
    http.send(TimedUrl);
}

function getMetaContentByName(mn) {
    returnContent = "";
    var m = document.getElementsByTagName('meta');

    var metadata = "Metadata\n";
    alert(document.querySelector('meta[property="description"]').getAttribute('content'));
    for (var i = 0; i < m.length; i++) {
        try {
            alert("Content : " + m[i].getAttribute("content"));
            if (m[i].name.toLowerCase() == mn.toLowerCase()) {
                returnContent = m[i].content;
                break;
            }
        } catch (err) {
            continue;
        }
    }
    return returnContent.trim();
}

```

Deuxième solution de sniffing web – sniff.js []

### Informations récupérées

Avec cette extension, on peut récupérer les URLs recherchées par l'utilisateur avec le temps passé sur chaque onglet comme le montre la figure ci-dessous. Les métadonnées sont également récupérables mais n'apparaissent pas dans la figure.

```
server started at 19119
chrome://extensions/?id=gnkebgfnodnhdhghmbgldmpkgdgbjaed 16.388000011444092
https://www.google.ch/ 2.064000129699707
chrome://extensions/?id=gnkebgfnodnhdhghmbgldmpkgdgbjaed 22.98099994659424
https://www.youtube.com/watch?v=Fi4YmLiC18E 2.875999927520752
```

Résultat de la récupération des URLs et leur temps de navigation []

## 2.5. Conclusion

Cette partie d'analyse nous a permis de mieux nous rendre compte de la portée du projet et de la manière d'aborder les fonctionnalités que nous pourrions implémenter. D'un point de vue humain, en lisant des témoignages et des informations sur le sujet des addictions aux réseaux sociaux et aux jeux vidéos, nous nous sommes rendu compte que la gestion du temps passé est un point central pour les générations en cours et à venir. Cela nous motive pour la phase d'implémentation.

Techniquement, il existe déjà beaucoup de produits existants avec des fonctionnalités performantes. Parmi ces logiciels figure notamment Rescute Time. Nous estimons qu'il est efficace. Nous avons observé sa manière de monitorer l'activité web des utilisateurs : nous pensons qu'il utilise les extensions des navigateurs afin de connaître les URL. Cela permet de ne pas avoir de problème de décodage en utilisant les couches réseaux. Nous nous en sommes inspirés pour créer la récupération de l'activité web de l'utilisateur.

Cette application monitorerait toutes les activités effectuées sur l'ordinateur. De notre côté, il a été convenu avec les clientes que nous nous concentrerions uniquement sur les thèmes des jeux vidéos et des réseaux sociaux (web et desktop). C'est pourquoi nous allons récupérer uniquement les informations relatives aux thèmes.

Lors de l'analyse technologique, nous avons constaté que les extensions Google Chrome ne sont pas installables sur les mobiles et que les extensions Firefox sont disponibles pour Android. Nous allons nous concentrer sur l'activité des ordinateurs, plus particulièrement des OS mac et Windows et nous utiliserons les extensions chromes pour monitorer les activités web de l'utilisateur.

La valeur ajoutée de notre produit par rapport aux produits existants est le fait que le projet est plus personnalisé par rapport aux souhaits des clientes. En effet, nous ciblons le monitoring sur un thème précis et nous ne nous occupons pas du reste. Il est possible que cela améliore les performances du logiciel.

Suite à ces recherches nous pouvons donc fixer le cadre du projet : nous allons mettre en place une application qui :

- Monitore uniquement l'activité sur les réseaux sociaux et jeux vidéos sur l'ordinateur de l'utilisateur (nous n'allons pas implémenter la surveillance sur les smartphones)
- Sort un fichier de log contenant les informations de l'activité de l'utilisateur
- Permet de consulter ces informations :
  - Sous forme de graphiques
  - Sur une application desktop
  - Sur le web

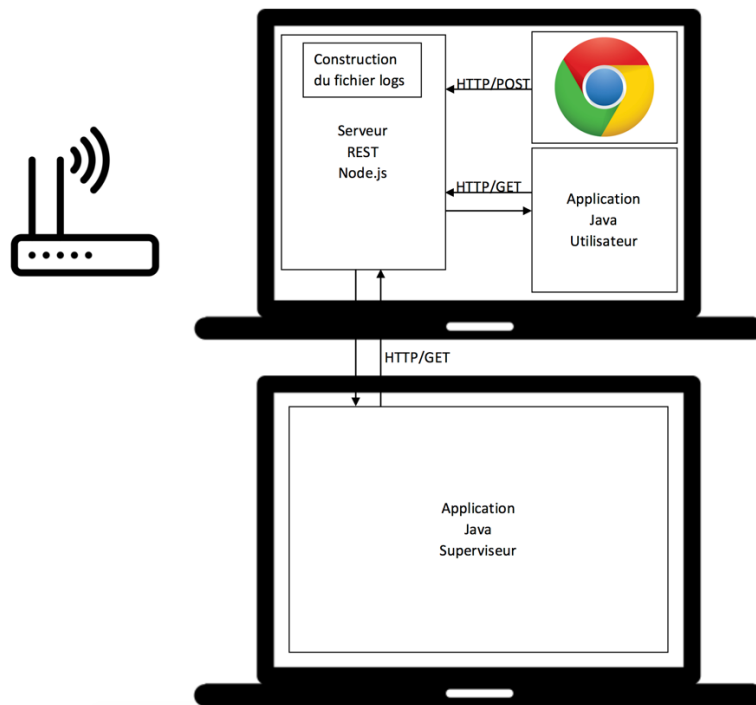
Nous avons ainsi le « scope » que nous allons donner à notre produit final.

### 3. Conception

#### 3.1. Introduction

La partie d'analyse de ce projet nous a permis de fixer le cadre du projet. Dans ce chapitre, c'est la manière de mettre en œuvre ce projet qui est présentée. Un diagramme de cas d'utilisations décrit les actions que le logiciel est capable d'effectuer. Pour chacun de ces cas d'utilisations il y a une fiche descriptive qui le caractérise, ainsi qu'un diagramme de séquence et les besoins en IHM. Il est possible que le fonctionnement final du logiciel diffère de ce qui est prévu à la conception, car en cette phase le développeur exprime comment il souhaite construire programme, mais parfois la phase d'implémentation révèle que ce qui est prévu doit être réalisé autrement.

Le schéma ci-dessous représente l'architecture de l'application:



Dans un même réseau se trouvent plusieurs machines :

- La machine de l'utilisateur à monitorer
- La machine du superviseur

Sur la machine utilisateur tourne un serveur local en NodeJS. Celui-ci récupère les informations sur l'activité utilisateur envoyées par le browser ainsi que par l'OS. Avec ces informations il crée un fichier de log. L'application java utilisateur récupère ce fichier de log et affiche des statistiques d'utilisation sur la machine monitorée.

Sur la machine du superviseur, il y a une application Java (qui est intégrée dans une application web) où le superviseur peut connaître les statistiques d'utilisation pour les machines qu'il monitoré.

Les informations entre les machines et les composants communiquent par des requêtes HTTP/GET et HTTP/POST.

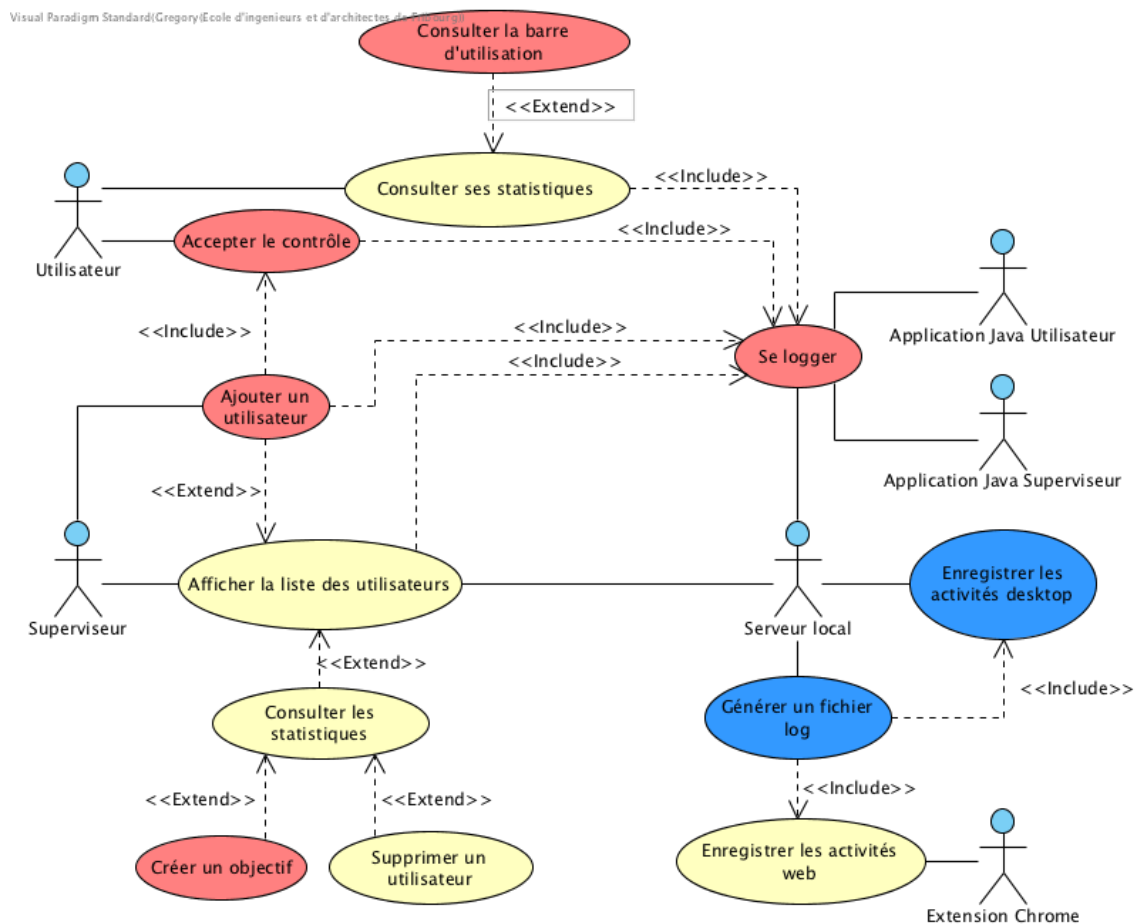
### 3.2. Cas d'utilisation

Le schéma ci-dessous représente le diagramme de cas d'utilisation de l'application. Les cas d'utilisations ont été établis sur la base des recherches et des discussions effectuées dans les premières semaines du projet. Il exprime une large palette des fonctionnalités que l'utilisateur pourra effectuer

Les différents cas sont implémentés dans les prototypes suivants :

1. En bleu, le prototype 1
2. En jaune, le prototype 2
3. En rouge, le prototype 3

Note : Les cas implémentés dans le prototype 1 le sont également dans le prototype 2 et 3. Ceux implémentés dans le 2 le sont aussi dans le 3.





### 3.2.1. Se logger

#### Fiche descriptive

**Description** : Un utilisateur ou un superviseur souhaite se logger à l'application afin de consulter les statistiques.

**Acteurs** : Serveur local, Utilisateur, Superviseur, Applications java

**Description des enchaînements** :

**Pré conditions** : -

**Scénario nominal**

1. La personne ouvre l'application java
2. L'application java vérifie si la personne est loggée
3. L'application java affiche une page de login
4. La personne entre son login et son mot de passe
5. L'application java vérifie et valide les données de connexion
6. Fin du scénario nominal

**Scénario alternatif**

A1 : La personne est déjà loggée

- Démarre au point 3 du scénario nominal
- 1. Reprend au point 6 du scénario nominal

A2 : Les données de connexion sont incorrectes

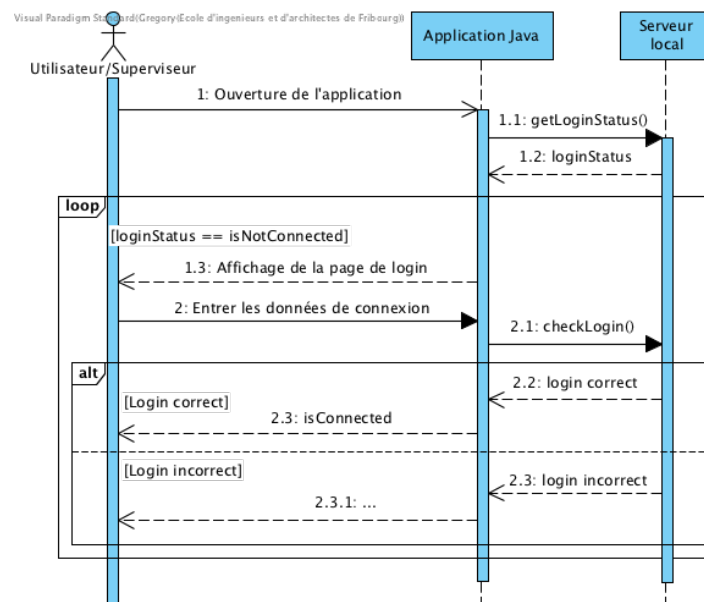
- Démarre au point 6 du scénario nominal
- 1. L'application affiche un message d'erreur
- 2. Reprend au point 4 du scénario nominal

**Post conditions** : -

**Besoin d'IHM** :

**Remarques (optionnel)** :

#### Diagramme de séquences



### 3.2.2. Enregistrer les activités desktop et web

#### Fiche descriptive

**Description :** Le serveur local / extension chrome récupère les activités desktop et web de l'utilisateur.

**Acteurs :** Serveur local, Utilisateur, Extension Chrome et Application Java Utilisateur

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

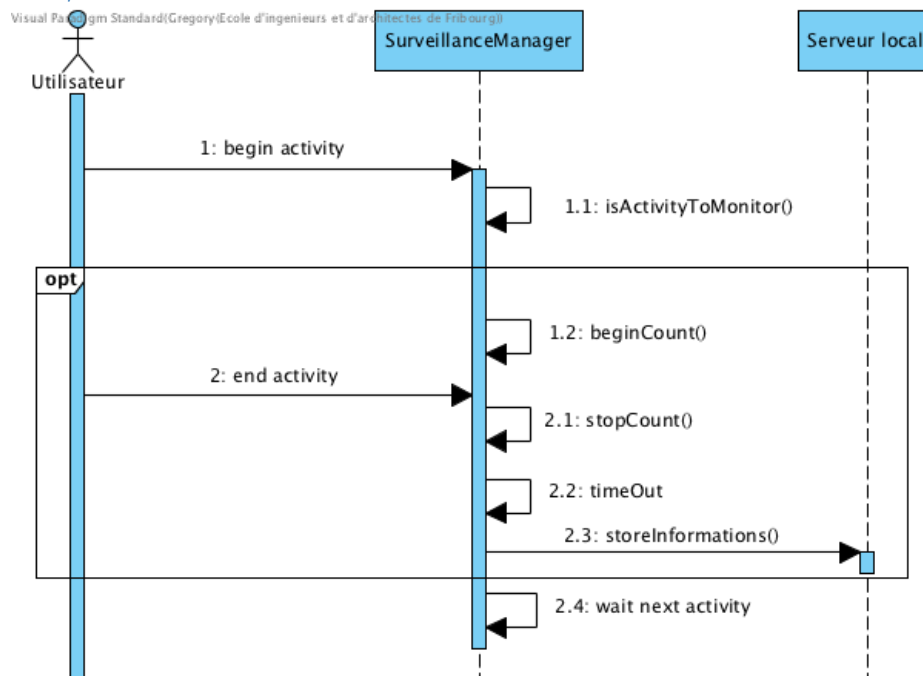
1. L'utilisateur commence une activité sur l'ordinateur
2. La surveillance détecte s'il s'agit d'un jeu ou d'un réseau social (desktop ou page web)
  - a. Si c'est le cas, un compteur démarre et se termine à la fin de l'activité
3. La surveillance est en attente jusqu'à la prochaine activité, auquel cas elle retourne au cas 1 du scénario nominal
4. La surveillance envoie les activités enregistrées au serveur local après un timeout
5. Reprend au point 1 du scénario nominal

**Post conditions :** -

**Remarques (optionnel) :**

La surveillance (extension chrome / serveur local) est constamment en train de vérifier les activités de l'utilisateur. Il est possible que le serveur local ne détecte pas tous les processus comme étant des jeux vidéos.

#### Diagramme de séquences



### 3.2.3. Générer un fichier de log

#### Fiche descriptive

**Description :** Les activités ont été récupérées par les cas <<UC : enregistrer les activités desktop>> et <<UC : enregistrer les activités web>>. Un fichier de log est généré à partir de ces informations.

**Acteurs :** Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

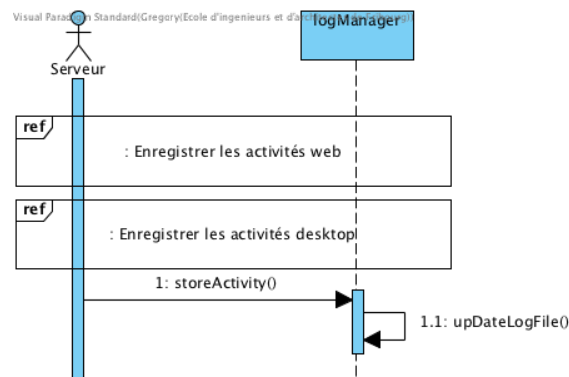
1. <<UC : Enregistrer les activités web>>
2. <<UC : Enregistrer les activités desktop>>
3. Le serveur local stocke les activités reçues dans un fichier log mis à jour régulièrement
4. Reprend au point 1 du scénario nominal

**Post conditions :** -

**Besoin d'IHM :**

**Remarques (optionnel) :**

#### Diagramme de séquences



### 3.2.4. Afficher la liste des utilisateurs

#### Fiche descriptive

**Description :** Le superviseur souhaite administrer ses utilisateurs et consulter les statistiques des informations enregistrées. Il veut aussi pouvoir supprimer ou ajouter des utilisateurs à surveiller et éventuellement leurs créer des objectifs.

**Acteurs :** Serveur local, Superviseur, Application Java Superviseur

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. <<UC : se logger>>
2. L'application Java Superviseur affiche la liste des utilisateurs
3. Le superviseur consulte la liste des utilisateurs monitorés
4. Le superviseur quitte l'application
5. Fin du scénario nominal

**Scénario alternatif**

A1 : Consulter les statistiques d'un utilisateur monitoré

- Démarre au point 2 du scénario nominal
  1. <<UC : Consulter les statistiques>>
  2. Reprend au point 2 du scénario nominal

A2 : Ajouter un utilisateur

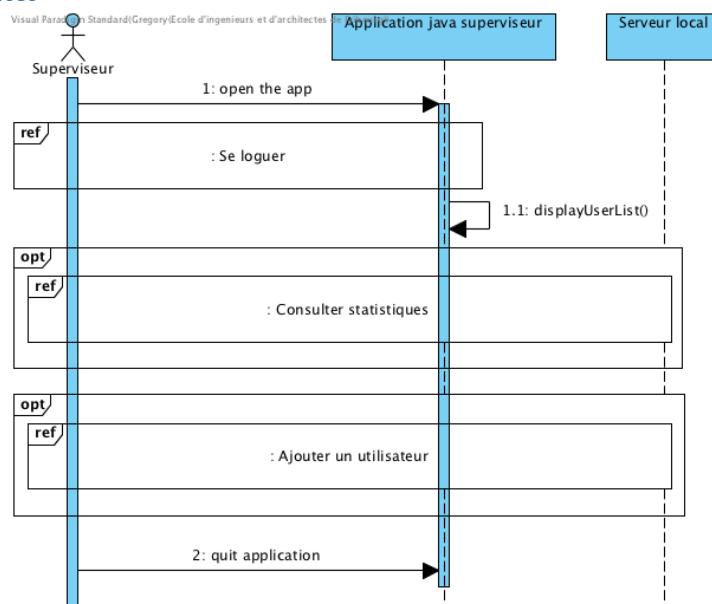
- Démarre au point 2 du scénario nominal
  1. <<UC : Ajouter un utilisateur>>
  2. Reprend au point 2 du scénario nominal

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences



### 3.2.5. Consulter les statistiques

#### Fiche descriptive

**Description** : Le superviseur souhaite consulter les statistiques des informations enregistrées des utilisateurs monitorés.

**Acteurs** : Superviseur, Application Java Superviseur, Serveur local

**Description des enchaînements** :

**Pré conditions** : -

**Scénario nominal**

1. Le superviseur sélectionne un utilisateur dans la liste
2. L'application Java Superviseur récupère les informations de l'utilisateur sur son serveur local
3. L'application Java Superviseur affiche les statistiques de l'utilisateur sélectionné
4. Le superviseur consulte les informations
5. Le superviseur revient à la liste des utilisateurs
6. Fin du scénario nominal

**Scénario alternatif**

A1 : Créer un objectif pour un utilisateur monitoré

- Démarre au point 4 du scénario nominal
- 1. <<UC : Créer un objectif>>
- 2. Reprend au point 4 du scénario nominal

A2 : Supprimer un utilisateur monitoré

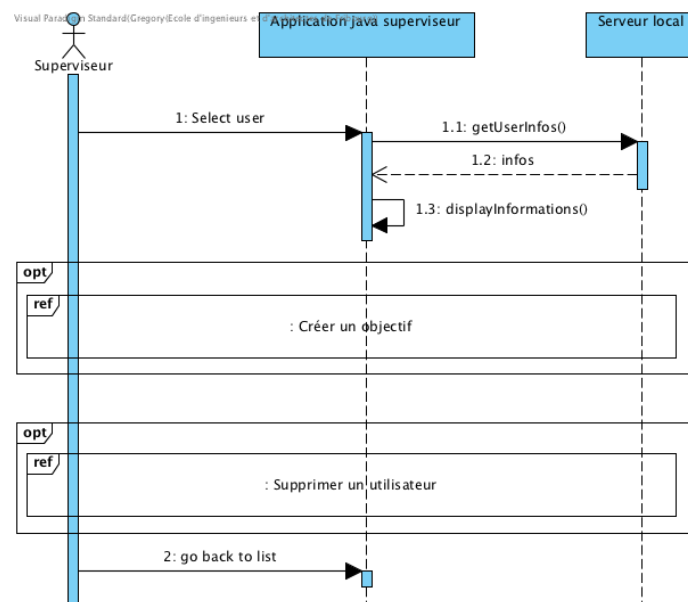
- Démarre au point 4 du scénario nominal
- 1. <<UC : Supprimer un utilisateur>>
- 2. Reprend au point 4 du scénario nominal

**Post conditions** : -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences



### 3.2.6. Créer un objectif

#### Fiche descriptive

**Description :** Le superviseur souhaite créer un objectif pour un utilisateur monitoré.

**Acteurs :** Superviseur, Application Java Superviseur, Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. Le superviseur clique sur le bouton "Créer un objectif"
2. Le système affiche le formulaire de création d'objectif
3. Le superviseur remplit le formulaire et clique sur le bouton "Sauvegarder"
4. Le serveur local de l'utilisateur monitoré enregistre l'objectif

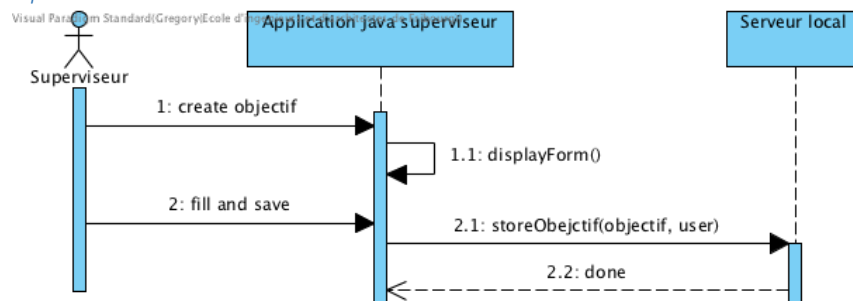
**Scénario alternatif**

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences



### 3.2.7. Ajouter un utilisateur

#### Fiche descriptive

**Description :** Le superviseur souhaite ajouter un utilisateur à monitorer.

**Acteurs :** Superviseur, Application Java Superviseur, Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. Le superviseur clique sur le bouton "Ajouter un utilisateur"
2. L'application Java Superviseur affiche le formulaire pour ajouter un utilisateur
3. Le superviseur remplit les champs puis valide
4. Le serveur local de l'utilisateur à monitorer enregistre les informations
5. <<UC : Accepter le contrôle>>
6. Fin du scénario nominal

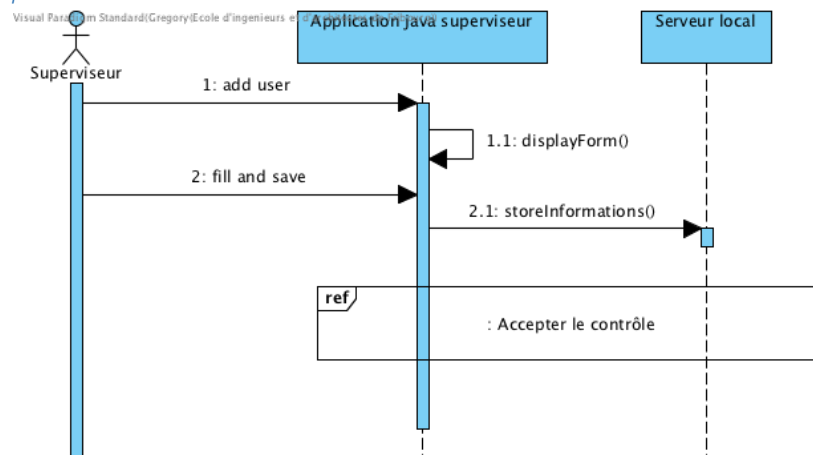
**Scénario alternatif**

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences



### 3.2.8. Supprimer un utilisateur

#### Fiche descriptive

**Description :** Le superviseur souhaite ne plus monitorer un utilisateur.

**Acteurs :** Superviseur, Application Java Superviseur, Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. Le superviseur clique sur le bouton "Supprimer un utilisateur"
2. L'application Java Superviseur affiche une demande confirmation de suppression
3. Le superviseur clique sur le bouton "Supprimer"
4. Le serveur local de l'utilisateur à ne plus monitorer supprime le droit
5. Fin du scénario nominal

**Scénario alternatif**

A1 : La suppression est annulée

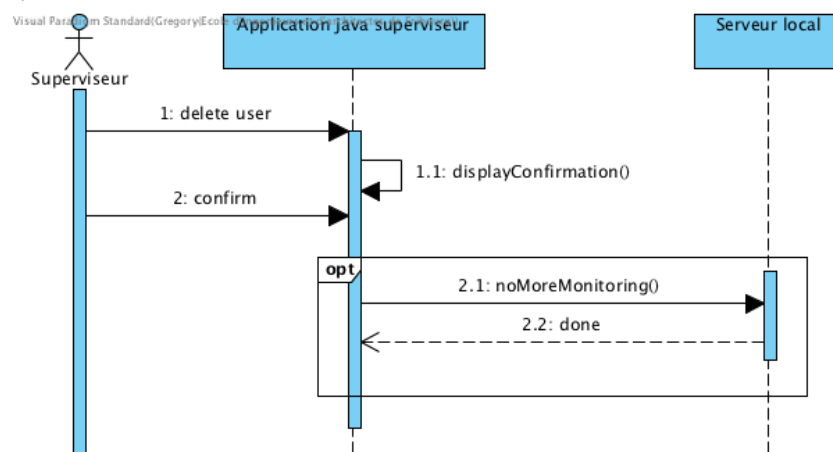
- Démarre au point 3 du scénario nominal
1. Le superviseur clique sur le bouton "Annuler"
  2. Fin du scénario alternatif

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences





### 3.2.9. Accepter le contrôle

#### Fiche descriptive

**Description :** Un utilisateur autorise le contrôle de ses statistiques à un superviseur

**Acteurs :** Utilisateur, Application Java Utilisateur, Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. <<UC : Se logger>>
2. L'utilisateur clique sur l'onglet "Demandes de monitoring"
3. L'application Java Utilisateur récupère les demandes de monitoring sur le serveur local
4. L'application Java Utilisateur affiche la liste des demandes
5. L'utilisateur clique sur le bouton "Accepter" à côté d'une demande
6. L'application Java Utilisateur met à jour la liste des demandes
7. L'application java envoie une confirmatin/infirmation au serveur local
8. Fin du scénario nominal

**Scénario alternatif**

A1 : L'utilisateur refuse la demande de monitoring

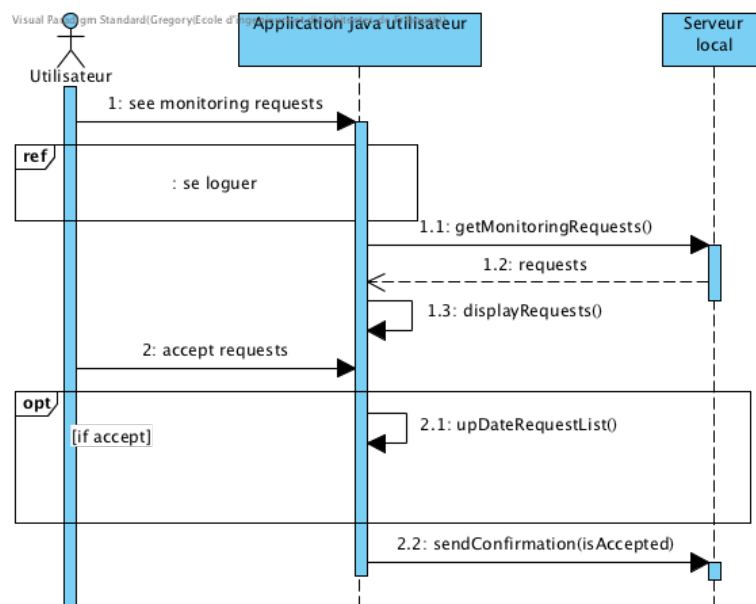
- Démarre au point 4 du scénario nominal
- 1. L'utilisateur clique sur le bouton "Refuser" à côté d'une demande
- 2. Fin du scénario alternatif

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences



### 3.2.10. Consulter la barre d'utilisation

#### *Fiche descriptive*

**Description :** Un utilisateur souhaite consulter sa barre d'utilisation.

**Acteurs :** Utilisateur, Application Java Utilisateur, Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. L'utilisateur clique sur l'icône de l'application sur le bureau
2. L'application Java Utilisateur récupère les données sur le serveur local
3. L'application Java Utilisateur affiche les données récupérées
4. Fin du scénario nominal

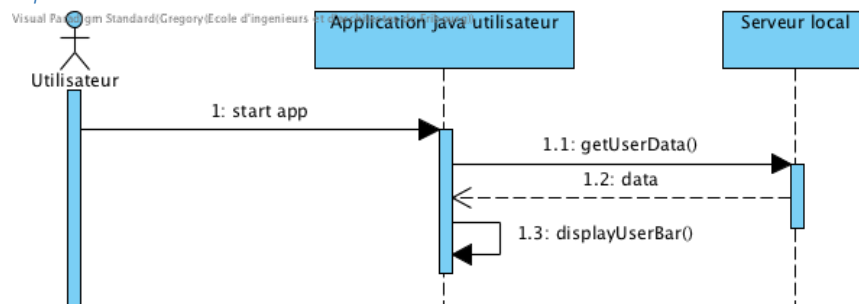
**Scénario alternatif**

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### *Diagramme de séquences*



### 3.2.11. Consulter ses statistiques

#### Fiche descriptive

**Description :** Un utilisateur souhaite consulter ses statistiques.

**Acteurs :** Utilisateur, Application Java Utilisateur, Serveur local

**Description des enchaînements :**

**Pré conditions :** -

**Scénario nominal**

1. <<UC : Se logger>>
2. L'application Java Superviseur récupère les informations de l'utilisateur sur son serveur local
3. L'application Java Superviseur affiche les statistiques récupérées
4. L'utilisateur consulte ses informations

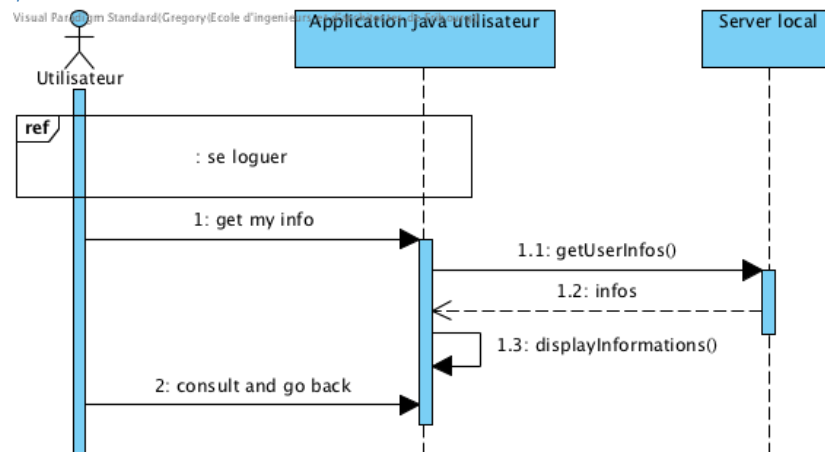
**Scénario alternatif**

**Post conditions :** -

**Besoin d'IHM:**

**Remarques (optionnel) :**

#### Diagramme de séquences



### **3.3. Structure du fichier Json**

### 3.4. Conclusion

Cette phase de conception oblige le développeur à penser concrètement à la manière dont il souhaite développer l'application. A ce stade la manière présumée de réaliser le logiciel est donc connue et il ne devrait y avoir qu'à appliquer ce qui a été réfléchi pour créer le produit final. Plus concrètement, cette phase de conception a permis :

1. D'établir les uses-case de l'application et pour chacun d'eux :
  - Les fiches descriptives
  - Les besoins IHM
  - Les enchaînements des actions
  - Les intervenants (système et humain)
2. De créer la structure du fichier JSON qui contiendra les informations sur les activités de l'utilisateur

Avec ces informations, il est possible de réaliser le logiciel en ayant une vision concrète du produit final tout en étant souple sur les besoins techniques inhérents à la phase d'implémentation.

## 4. Implémentation

### 4.1. Introduction

La phase de conception a permis de décider concrètement comment l'on souhaite que l'application fonctionne. La phase d'implémentation est celle qui réalise le logiciel de manière pratique. Voici un rappel de la portée du travail effectué par le logiciel de monitoring:

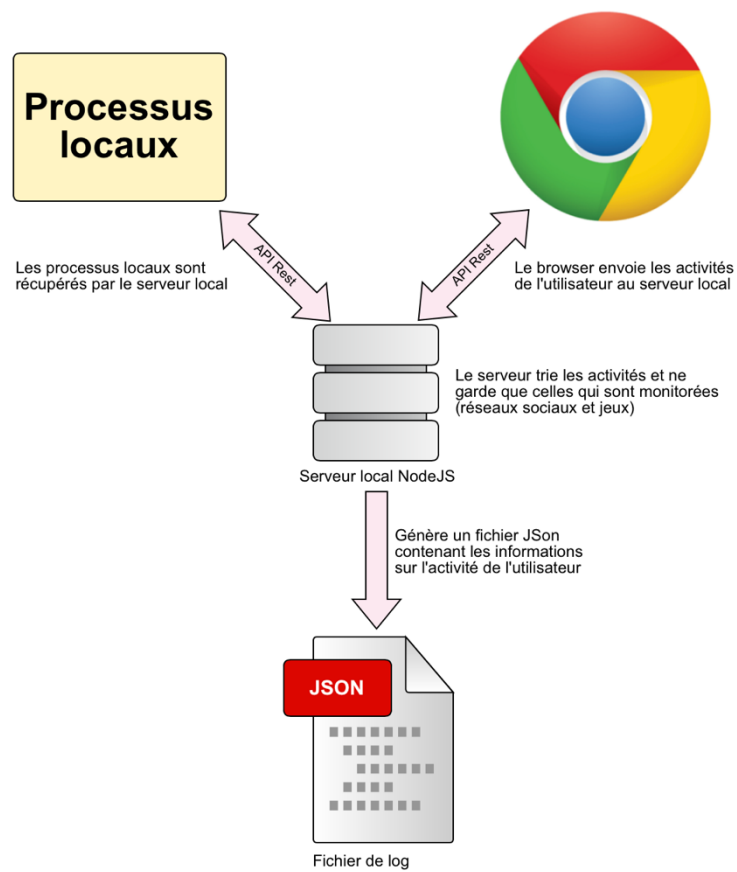
- Il monitore les activités web et desktop de l'utilisateur
- Les activités filtrées sont celles qui concernent les jeux vidéos et les réseaux sociaux
- Les plateformes systèmes supportant le monitoring sont Mac OS et Windows
- Le browser permettant la surveillance des activités web est Google Chrome
- Le logiciel récupère l'activité de machines se trouvant sur le même réseau

L'implémentation est découpée en 3 parties. Il y a 3 prototypes qui sont construits successivement en réutilisant le prototype précédent :

1. Récupération, filtrage et enregistrement dans un fichier de log des des activités web et desktop de l'utilisateur
2. Affichage des informations de log dans une application Java tournant sur la machine monitorée
3. Affichage des informations de log dans une application web afin que le superviseur puisse voir les informations de plusieurs machines

Comme expliqué précédemment, le lien entre ces prototypes est très fort car tous ont besoin du prototype précédent pour fonctionner (sauf le 1).

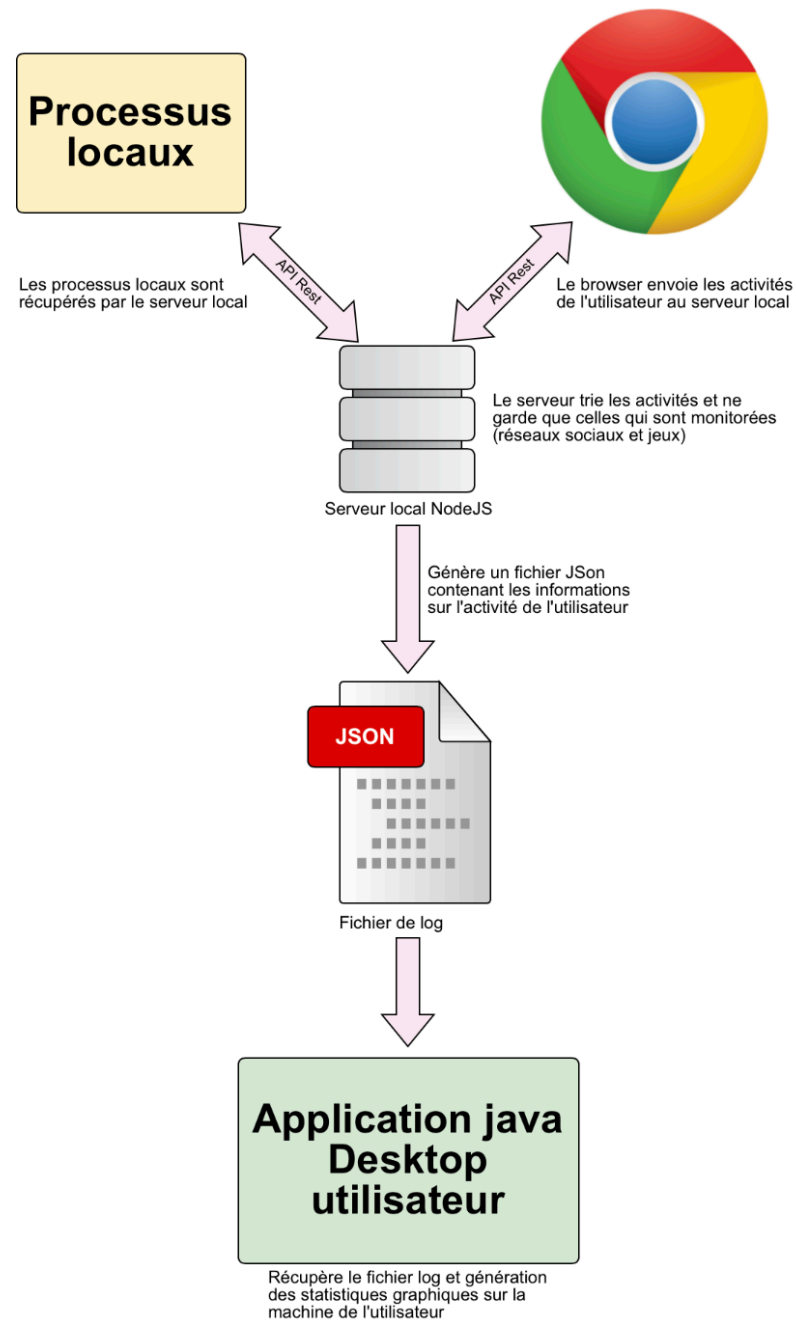
## 4.2. Prototype 1



Fichier log, serveur local

### 4.3. Prototype 2

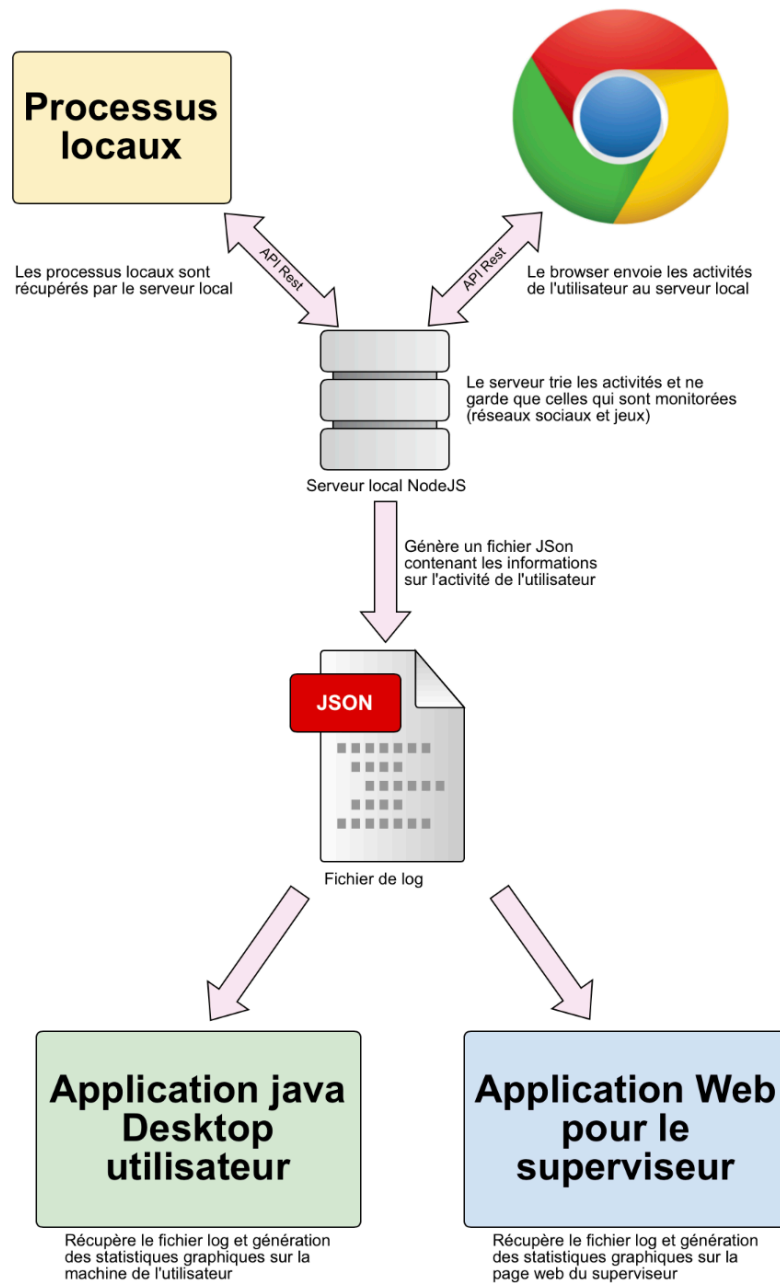
DadshBoard desktop (app java), serveur local





#### 4.4. Prototype 3

DashBoard web, serveur



#### **4.5. Conclusion**

## **5. Validation**

### **5.1. Introduction**

## 5.2. Conclusion

# 6. Conclusion

### 6.1. Conclusions de groupe

### 6.2. Conclusions personnelles

6.2.1. Grégory Ducrey

6.2.2. Nicolas Fuchs

# 7. Annexes

## 7.1. Déclaration d'honneur

## 7.2. Bibliographie

Ces sites ont été visités sur la période du 03.2018 et 05.2018.

- [1] [https://fr.wikipedia.org/wiki/Dépendance\\_au\\_jeu\\_vidéo](https://fr.wikipedia.org/wiki/Dépendance_au_jeu_vidéo)
- [2] <http://www.jeuxvideo.com/dossiers/00018270/l-addiction-aux-jeux-video-temoignages-de-joueurs-006.htm>
- [3] <https://www.vice.com/fr/article/ppvzd7/dependance-jeux-video-843>
- [4] <http://www.ot-lab.ch/?p=5605>
- [5] <http://humanetech.com/take-control/>
- [6] <https://selfcontrolapp.com>
- [7] [https://support.apple.com/kb/PH25799?viewlocale=fr\\_FR&locale=hu\\_HU](https://support.apple.com/kb/PH25799?viewlocale=fr_FR&locale=hu_HU)
- [8] <https://www.rescuetime.com/dashboard>
- [9] <https://www.rescuetime.com/anapi/setup/overview>
- [10] <https://itunes.apple.com/us/app/moment-screen-time-tracker/id771541926?mt=8>
- [11] <https://www.swisscom.ch/fr/clients-prives/aide/internet/kinder-und-jugendschutz.html>
- [12] <https://www.swisscom.ch/fr/business/pme/internet-reseaux-fixe-telephonie/internet/services-supplementaires/internet-security.html>
- [13] <https://www.rescuetime.com/rescuetime-pro>