





AP

9- Développement du simulateur Windows (en C#)

1- Rappel de l'architecture applicative globale et objectif de cette étape

La page suivante présente l'architecture générale de l'application **TraceGPS**.

Cette étape porte sur le **développement et le test du simulateur de parcours sous Windows** .

Ce simulateur permettra à un **utilisateur authentifié** de simuler un parcours par clonage :

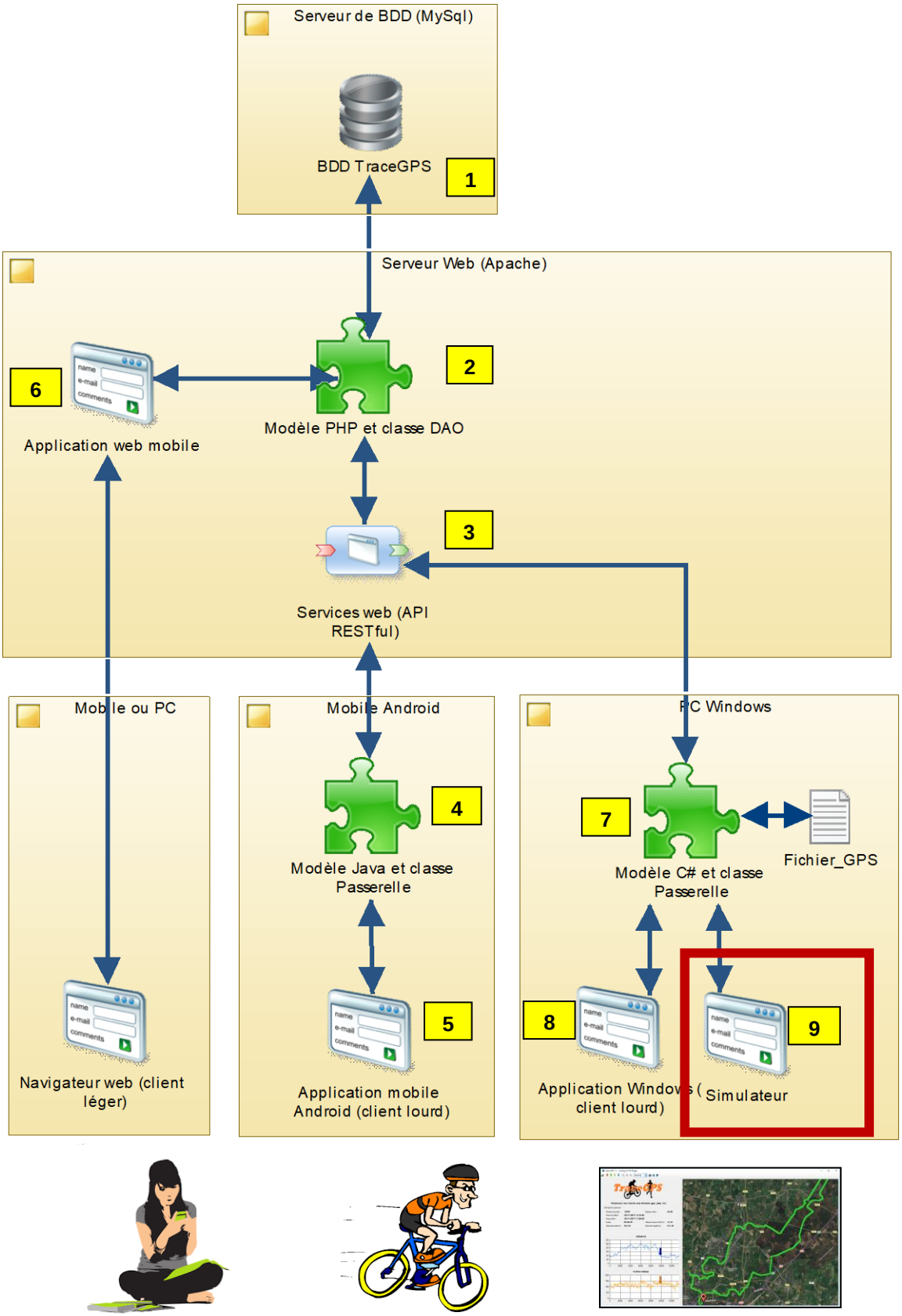
- d'un de ses propres parcours
- d'un parcours pour lequel il possède l'autorisation de consultation
- d'un parcours enregistré dans un fichier GPS (formats **GPX**, **PWX** et **TCX**)

La simulation consiste à créer en temps réel une trace en envoyant régulièrement (toutes les 15 ou 30 secondes par exemple) les points d'une trace existante. On utilise toutes les données du point existant (latitude, longitude, altitude, rythme cardio), à l'exception de l'heure de passage qui est remplacée par l'heure système, afin de simuler un parcours se déroulant à l'instant présent.

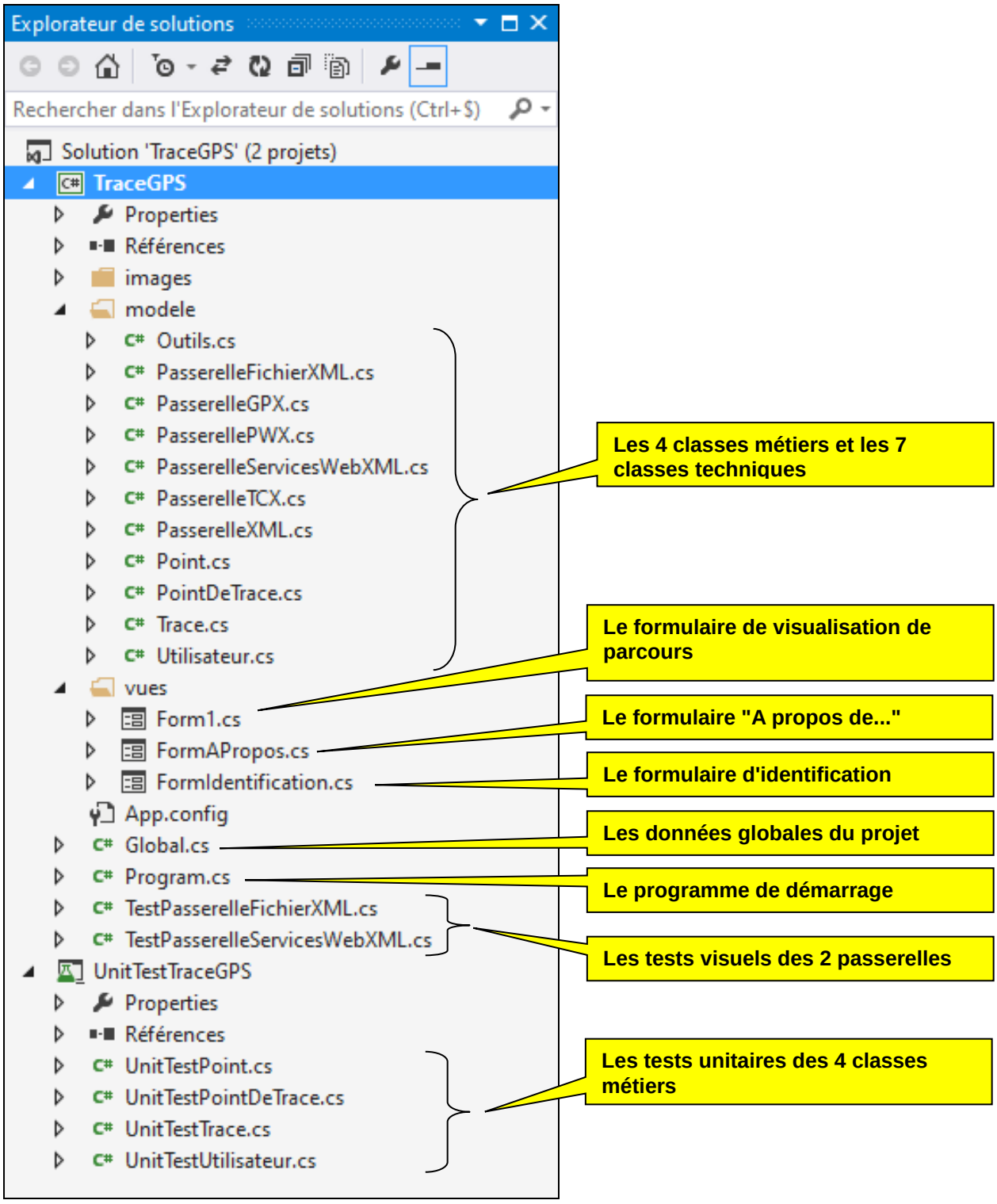
Ce simulateur a pour objectif de fournir un outil de test de l'application Windows, et notamment de la fonction d'affichage d'un parcours en cours d'exécution (quand on n'a pas la possibilité d'envoyer quelqu'un sur le terrain).

Cette étape consiste à adapter l'application TraceGPS modifiée lors de l'étape précédente.

Vous allez compléter la solution TraceGPS de l'étape précédente (**développement et test de l'application Windows)**.


Serveur de BDD (MySql)
BDD TraceGPS
1
Serveur Web (Apache)
2
Modèle PHP et classe DAO
6
Application web mobile
3
Services web (API RESTful)
Mobile ou PC
Navigateur web (client léger)
Mobile Android
4
Modèle Java et classe Passerelle
5
Application mobile Android (client lourd)
PC Windows
7
Modèle C# et classe Passerelle
Fichier_GPS
8
Application Windows (client lourd)
9
Simulateur

2- La structure de l'application



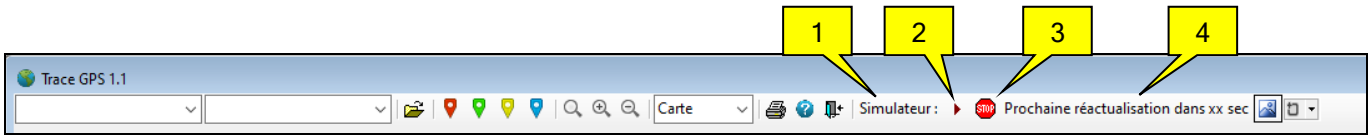
3- Les modifications demandées

3-1 Modifications graphiques du formulaire et de la barre d'outils

Dans le bas du formulaire, ajoutez un contrôle **Timer** qui servira à envoyer périodiquement la position. Il sera désactivé au départ, et sa fréquence sera réglée à 1 seconde (1000 ms).

Dans la barre d'outils, ajouter un séparateur et 4 contrôles :

1. un **label** annonçant que le simulateur est actif
2. un **bouton** pour démarrer la simulation (l'icône est fournie)
3. un **bouton** pour arrêter la simulation (l'icône est fournie)
4. un **label** annonçant la durée (en secondes) avant le prochain envoi d'une géolocalisation



Ces 4 contrôles ne seront visibles qu'en cas d'activation du mode simulateur.

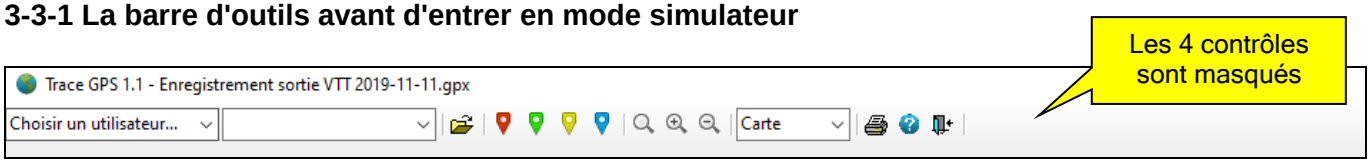
3-2 Activation du mode simulateur

Le mode simulateur s'obtiendra par un double-clic sur le logo de l'application et ne sera possible que si les 2 conditions suivantes sont réunies :

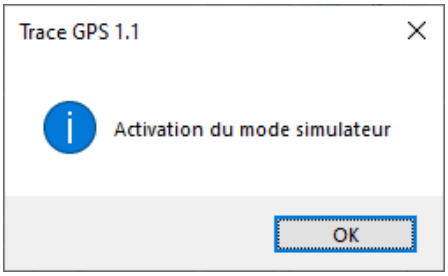
- l'accès à l'application s'est fait avec une connexion authentifiée (l'utilisateur autentifié sera automatiquement le propriétaire de la trace créée par simulation)
- une **trace terminée** a été chargée et est visible sur la carte.

3-3 Les différents états de la barre d'outils

3-3-1 La barre d'outils avant d'entrer en mode simulateur



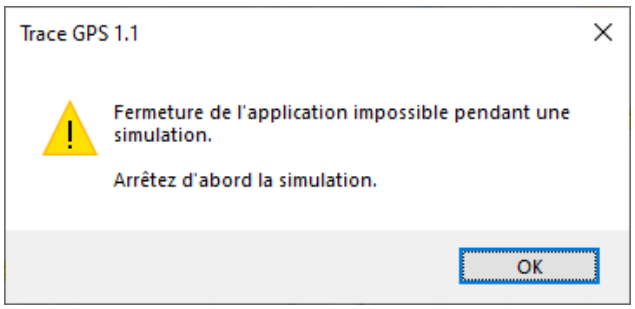
Activation du mode simulateur par un double-clic sur le logo (l'activation est confirmée par l'affichage d'un message) :



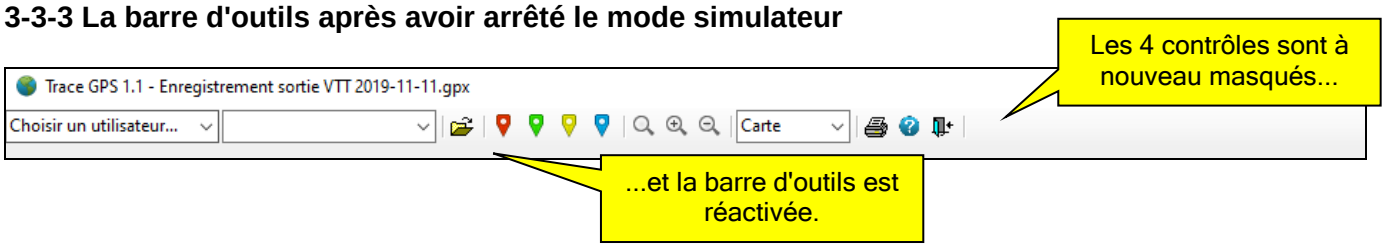
3-3-2 La barre d'outils pendant la simulation (après avoir cliqué sur le bouton Démarrer)

Pendant la simulation, la fermeture de l'application doit être impossible.

Ceci se gère avec l'événement **FormClosing** du formulaire, et un message en cas de tentative de fermeture :



3-3-3 La barre d'outils après avoir arrêté le mode simulateur



3-4 Gestion du clic sur le bouton Démarrer

Créer la trace avec la méthode **PasserelleServiceWebXML.demarrerEnregistrementParcours**.

- En cas de problème, afficher un message.
- En cas d'absence de problème :
  - mémoriser l'heure de début du parcours
  - afficher le label d'affichage de la durée avant le prochain envoi (1 seconde avant le premier envoi, 15 ou 30 secondes entre les envois suivants, en fonction du paramètre **Global.FREQUENCE_ENVOI**)
  - désactiver tous les contrôles de la barre d'outils (sauf le bouton permettant d'arrêter la simulation)
  - activer le Timer qui gère l'envoi périodique (fréquence : 1 seconde)

Testez cette fonctionnalité et vérifiez avec **PhpMyAdmin** qu'une trace a bien été créée (trace non terminée avec 0 points pour l'instant) dans la base de données.

3-5 Gestion du clic sur le bouton Arrêter

Terminer la trace avec la méthode **PasserelleServiceWebXML.arreterEnregistrementParcours**.

- désactiver le Timer qui gère l'envoi périodique
- masquer les 4 contrôles graphiques de la fonction simulateur
- réactiver tous les contrôles de la barre d'outils (sauf le bouton permettant d'arrêter la simulation)

Testez cette fonctionnalité et vérifiez avec **PhpMyAdmin** que la trace a bien été terminée dans la base de données.

3-6 Gestion de l'envoi périodique de la position

Le Timer génère l'événement **Tick** chaque seconde. Cet événement déclenchera le traitement suivant :

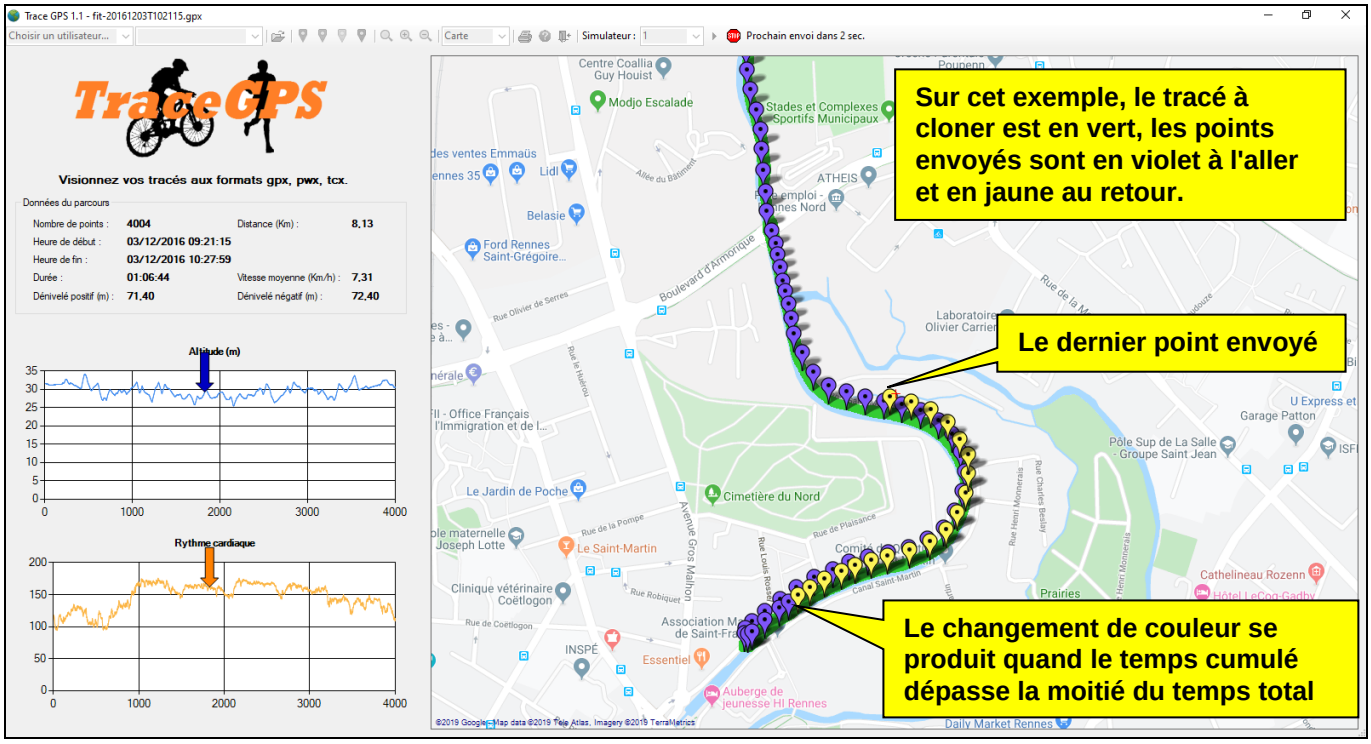
- décrémenter le nombre de secondes restant avant le prochain envoi
- si le nombre de secondes restant avant le prochain envoi est égal à 0 :
    - calculer la durée (en secondes) écoulée depuis le début de la simulation (exemple : 60 secondes)
    - rechercher dans la trace à clôner le premier point dont le temps cumulé (en secondes) est supérieur ou égal au temps écoulé depuis le début de la simulation (exemple : le premier point dont le temps cumulé est supérieur ou égal à 60 secondes)
    - si cette recherche aboutit au dernier point de la trace, terminer la trace avec la méthode **PasserelleServiceWebXML.arreterEnregistrementParcours** et effectuer le même traitement que dans le cas du clic sur le bouton **Arrêter**
    - si cette recherche trouve un point du parcours, récupèrer ce point, lui modifier son heure de passage en lui attribuant l'heure système, et l'envoyer avec la méthode **PasserelleServiceWebXML.envoyerPosition**
        - si l'envoi retourne un message d'erreur, afficher ce message
        - si l'envoi se passe bien, ajouter un marqueur au tracé actuellement affiché (**GMarkerGoogleType.purple_dot** en première moitié du parcours, **GMarkerGoogleType.yellow_dot** en seconde moitié du parcours) et faire en sorte que le survol de ce marqueur par la souris ne déclenche pas le traitement prévu pour les points de la trace à clôner
    - recharger le nombre de secondes restant avant le prochain envoi à partir de la constante **Global.FREQUENCE_ENVOI** placée dans le fichier **Global.cs**
- afficher le nombre de secondes restant avant le prochain envoi dans la barre d'outils

Testez cette fonctionnalité et vérifiez avec **PhpMyAdmin** que les points sont bien ajoutés à la trace démarrée dans la base de données.

Les pages suivantes montrent les écrans à obtenir avec le simulateur et avec la visualisation de parcours non terminé.

Exemple d'un aller-retour en kakak sur le canal d'Ille et Rance

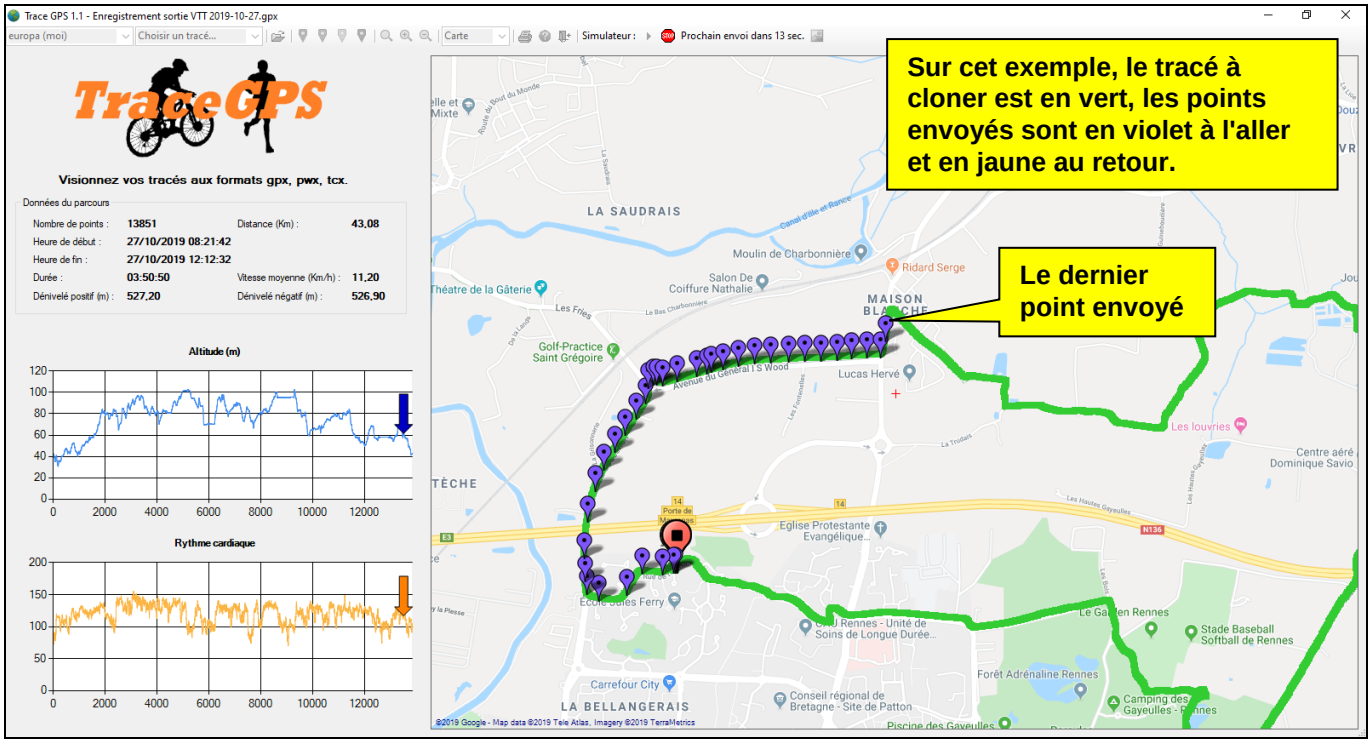
Ecran du simulateur :



On peut utiliser une autre instance de l'application pour visualiser le parcours en temps réel, ce qui donnerait avec la simulation précédente :

Exemple d'une boucle en VTT

Ecran du simulateur :



On peut utiliser une autre instance de l'application pour visualiser le parcours en temps réel, ce qui donnerait avec la simulation précédente :