

Définition et caractérisation du modèle de traces utilisateur pour l'application d'exploration urbaine MOBILES

Madjid Sadallah, Marie Lefevre

▶ **To cite this version:**

Madjid Sadallah, Marie Lefevre. Définition et caractérisation du modèle de traces utilisateur pour l'application d'exploration urbaine MOBILES. LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/École Centrale de Lyon. 2023. ⟨hal-05120659⟩

HAL Id: hal-05120659

https://hal.science/hal-05120659v1

Submitted on 19 Jun 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons CC BY-NC-SA 4.0 - Attribution - Non-commercial use - ShareAlike - International License

Définition et caractérisation du modèle de traces utilisateur pour l'application d'exploration urbaine MOBILES

Madjid Sadallah & Marie Lefevre

Mars 2023

Ce rapport présente la spécification détaillée du modèle de traces développé pour l'application MOBILES, destinée à accompagner les étudiants internationaux dans l'exploration urbaine. S'inscrivant dans le cadre de la spécification KTBS (Knowledge Trace Base System), ce modèle définit de manière formelle la structure des enregistrements d'interactions utilisateurs. Il catalogue les différents types d'observations correspondant aux actions clés entreprises au sein de l'application, ainsi que les attributs pertinents pour chaque observation, allant des coordonnées spatiales aux informations personnelles. L'objectif principal de ce modèle est d'assurer la génération de traces d'activités cohérentes, normalisées et riches sémantiquement. In fine, il vise à fournir une fondation robuste pour la traçabilité des comportements utilisateurs, facilitant leur analyse approfondie et la gestion des données générées par l'application MOBILES.

1. Introduction

L'application MOBILES, conçue pour accompagner les étudiants internationaux dans leur exploration urbaine, leur offre des outils pour annoter leurs découvertes, partager leurs expériences et documenter leurs parcours. Afin de comprendre en profondeur la manière dont ces utilisateurs s'approprient la ville et interagissent avec les fonctionnalités de l'application, la capture systématique de leurs activités s'avère indispensable. En effet, dans le domaine des systèmes interactifs et plus particulièrement des applications mobiles qui médiatisent des expériences complexes comme l'exploration spatiale et sociale, la collecte de traces d'interaction détaillées est un prérequis fondamental. Ces traces permettent non seulement d'analyser les usages et d'identifier des schémas comportementaux significatifs, mais aussi d'évaluer l'ergonomie de l'application, de personnaliser l'expérience utilisateur et, à terme, d'informer les cycles de conception et d'amélioration continue. C'est une pratique courante dans le développement logiciel et la recherche en Interactions Homme-Machine (IHM) que de s'appuyer sur des données d'usage pour fonder les décisions.

Face à ce besoin crucial, l'objectif principal de ce travail est la définition et la spécification d'un *modèle de traces* formel et dédié à l'application MOBILES. L'élaboration d'un tel modèle est une étape critique avant toute collecte de données. Sans un schéma directeur clair, les données collectées risqueraient d'être hétérogènes, incomplètes, ou difficiles à interpréter, rendant toute analyse ultérieure fastidieuse, voire impossible. Ce modèle vise donc à établir une structure rigoureuse et standardisée pour tous les enregistrements d'interaction qui seront générés par le moteur de traces de l'application. Il doit garantir une représentation exhaustive et précise des actions clés entreprises par les utilisateurs – des interactions les plus simples, comme la connexion, aux plus complexes, comme la création d'une annotation géolocalisée enrichie. En fournissant cette base structurée, le modèle a pour finalité d'assurer la génération de traces d'usage qui soient non seulement cohérentes et normalisées, mais aussi sémantiquement riches et directement exploitables pour des analyses approfondies. La normalisation, par exemple, est essentielle pour comparer les comportements entre différents utilisateurs ou sur différentes

périodes, tandis que la richesse sémantique assure que les traces contiennent suffisamment d'informations pour répondre à des questions de recherche ou d'analyse pertinentes.

Pour encadrer cette démarche de modélisation, le présent modèle de traces s'inscrit dans le cadre de la spécification *KTBS (Knowledge Trace Base System)*[1]. KTBS est un système de gestion de bases de traces qui propose un ensemble de concepts et de principes pour la modélisation, la collecte, le stockage et le partage de traces d'activités. Choisir un cadre comme KTBS permet de bénéficier d'une approche éprouvée et de concepts bien définis, facilitant l'interopérabilité potentielle avec d'autres outils ou systèmes d'analyse compatibles. Les notions fondamentales de KTBS, telles que l'« *Obsel* » et l'« *AttributeType* », sont donc centrales dans la conception du modèle de trace MOBILES. Un « Obsel » peut être compris comme l'unité élémentaire d'observation, l'enregistrement atomique d'une action ou d'un événement significatif survenu à un instant T (par exemple, "l'utilisateur X a ajouté un post Y à la position Z"). L'« AttributeType », quant à lui, définit la nature et le type des différentes propriétés ou informations associées à un Obsel (par exemple, pour un Obsel "addPost", les AttributeTypes pourraient inclure "userId", "position", "message", "timestamp").

Il est donc important de clarifier les termes employés :

- Une « *trace* », dans le contexte de ce rapport et plus généralement en analyse d'interactions, désigne un enregistrement séquentiel et chronologique des Obsels relatifs aux activités d'un ou plusieurs utilisateurs au sein de l'application MOBILES. Une trace est donc une *instance* de l'activité.
- Le « *modèle de traces* » est la spécification formelle – le schéma directeur, le « *blueprint* » – qui définit la nature, la structure, et le contenu potentiel de ces Obsels et, par extension, des traces. C'est le modèle qui dicte quels types d'événements sont enregistrés et quelles informations sont collectées pour chacun d'eux. Ce document constitue la description détaillée de ce modèle de traces pour MOBILES, servant de référence pour l'implémentation du mécanisme de traçage au sein de l'application et pour l'interprétation future des données collectées.

Ce rapport est structuré comme suit : la section 2 détaillera la spécification complète du modèle de traces MOBILES, en commençant par ses principes fondateurs, sa structure générale, puis en présentant un catalogue exhaustif des Obsels définis et de leurs AttributeTypes spécifiques. La section 3 abordera la transition de cette spécification théorique vers la pratique, en expliquant comment le modèle est destiné à être instancié par le moteur de traces de l'application pour générer les traces effectives. La section 4 discutera ensuite de la valeur et des implications de ce modèle de traces, tant pour la qualité des données que pour leur potentiel d'exploitation. Enfin, la section 5 présentera les conclusions de ce travail de spécification.

2. Spécification du modèle de traces MOBILES

La conception du modèle de traces pour l'application MOBILES a été guidée par la nécessité de capturer fidèlement et de manière structurée la diversité des interactions des utilisateurs. Ce modèle, comme introduit précédemment, s'appuie sur les concepts de KTBS et vise à fournir une représentation normalisée et sémantiquement riche des activités au sein de l'application. Cette section détaille les principes fondateurs de ce modèle, sa structure générale, et présente un catalogue exhaustif des types d'observations (Obsels) définis, ainsi que les attributs (AttributeTypes) qui les caractérisent.

[1] Settouti LS, Prié Y, Cram D, Champin PA and Mille A (2009) A trace-based framework for supporting digital object memories. In: Workshops Proceedings of the 5th International Conference on Intelligent Environments. IOS Press, pp. 39–44.

2.1. Principes fondateurs du modèle

Le modèle de traces MOBILES repose sur plusieurs principes directeurs essentiels pour atteindre ses objectifs de traçabilité et d'analyse :

1. **Adéquation aux besoins spécifiques de MOBILES :** Le modèle a été conçu sur mesure pour refléter les fonctionnalités et les flux d'interaction propres à l'application MOBILES. Il ne s'agit pas d'un modèle générique, mais d'une spécification tenant compte du contexte d'exploration urbaine par des étudiants internationaux, de l'annotation d'expériences et de la documentation de parcours.
2. **Granularité des observations (Obsels) :** Chaque action utilisateur considérée comme significative pour l'analyse a été modélisée comme un Obsel distinct. Le niveau de granularité a été choisi pour permettre une reconstruction fine des sessions utilisateur tout en évitant une surabondance de données peu informatives. Les Obsels couvrent un large spectre d'activités, allant de la gestion de compte utilisateur à la création de contenu et à la navigation.
3. **Richesse sémantique des attributs (AttributeTypes) :** Pour chaque type d'Obsel, un ensemble précis d'AttributeTypes a été défini. Ces attributs visent à capturer toutes les informations contextuelles pertinentes associées à l'action. Comme l'indique le document, cela va "des coordonnées spatiales aux informations personnelles des utilisateurs", assurant ainsi une compréhension approfondie de chaque événement tracé.
4. **Cohérence et normalisation :** Un des objectifs primordiaux est d'assurer, *via* ce modèle, que toutes les traces générées soient cohérentes dans leur structure et leur format. Cette normalisation est cruciale pour simplifier le traitement et l'analyse des données, permettant des agrégations, des comparaisons et l'application d'algorithmes d'analyse de manière standardisée.
5. **Conformité KTBS :** L'adhésion aux principes de KTBS facilite non seulement la conception du modèle, mais ouvre également la voie à l'utilisation potentielle d'outils et de plateformes compatibles avec KTBS pour le stockage, la visualisation ou l'analyse des traces.

2.2. Structure générale des traces basée sur le modèle

Toute trace générée par l'application MOBILES sera une séquence chronologique d'Obsels, chacun étant une instanciation d'un des types définis dans ce modèle. Conformément aux conventions (et comme visible sur le diagramme *via* la mention "Par défaut"), chaque Obsel, quel que soit son type spécifique, partagera un ensemble d'attributs fondamentaux :

- **ID :** Un identifiant unique pour chaque instance d'Obsel, permettant de le référencer de manière non ambiguë.
- **Timestamp de début :** L'horodatage précis marquant le commencement de l'action ou de l'événement représenté par l'Obsel.
- **Date et heure de début :** Représentation lisible par l'homme du timestamp de début.
- **Timestamp de fin :** L'horodatage précis marquant la fin de l'action ou de l'événement. Certaines actions peuvent être quasi instantanées, auquel cas le timestamp de fin peut être très proche ou identique à celui de début.
- **Date et heure de fin :** Représentation lisible par l'homme du timestamp de fin.

- **Sujet (Subject) :** L'entité principale concernée par l'Obsel. Dans la majorité des cas pour MOBILES, il s'agira de l'identifiant de l'utilisateur (userId) qui a initié l'action.

Ces attributs par défaut assurent une base commune à toutes les observations, facilitant le tri chronologique, l'identification des acteurs et la délimitation temporelle des événements.

2.3. Catalogue détaillé des Obsels et de leurs AttributeTypes

Le cœur du modèle de traces MOBILES réside dans la définition précise des différents types d'Obsels et des AttributeTypes qui leur sont associés. Le diagramme et le tableau en annexe détaillent cette structure. Nous allons les parcourir de manière thématique, en explicitant la signification de chaque Obsel et de ses attributs.

2.3.1. Modélisation des interactions de gestion des utilisateurs

Ces Obsels tracent les actions fondamentales liées au cycle de vie du compte utilisateur et à la gestion de son profil.

- **createUser / updateUser :**
    - **Rôle :** Enregistre la création d'un nouveau compte utilisateur ou la mise à jour des informations d'un profil existant.
    - **AttributeTypes spécifiques :**
        - arrivalDate (Texte) : La date d'arrivée de l'utilisateur dans la ville. Pertinent pour comprendre le contexte de l'étudiant international.
        - gender (Texte) : Le genre de l'utilisateur.
        - nationality (Texte) : La nationalité de l'utilisateur.
        - formation (Texte) : Le domaine ou niveau d'études de l'utilisateur (sa formation).
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de la création/mise à jour du profil*. Cela peut être moins pertinent pour le profil lui-même que pour l'action de le modifier.
        - lastname (Texte) : Le nom de famille de l'utilisateur.
        - firstname (Texte) : Le prénom de l'utilisateur.
        - birthdate (Date) : La date de naissance de l'utilisateur.
        - address (Texte) : L'adresse de l'utilisateur.
        - city (Texte) : La ville de résidence de l'utilisateur.
        - postalcode (Texte) : Le code postal de l'utilisateur.
        - username (Texte) : Le nom d'utilisateur choisi pour se connecter à l'application (probablement aussi l'identifiant userId).
        - institution (Texte, visible sur le diagramme pour createUser et updateUser) : L'établissement d'enseignement de l'utilisateur.

        - diploma (Texte, visible sur le diagramme pour createUser et updateUser) : Le type de diplôme préparé par l'utilisateur.

- **Login :**
    - **Rôle :** Trace l'événement de connexion d'un utilisateur à l'application.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'identifiant de l'utilisateur qui se connecte. (Note : Le diagramme mentionne username sous Login, ce qui est logique pour l'action de login, mais userId est plus générique pour identifier l'utilisateur tracé).
        - position (Coordonnées) : La position géographique de l'utilisateur au moment de la connexion.
- **Logout :**
    - **Rôle :** Trace l'événement de déconnexion d'un utilisateur.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'identifiant de l'utilisateur qui se déconnecte.
        - position (Coordonnées) : La position géographique de l'utilisateur au moment de la déconnexion.
- **seeMyProfile :**
    - **Rôle :** Enregistre l'action d'un utilisateur consultant son propre profil.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'identifiant de l'utilisateur qui consulte son profil.
        - position (Coordonnées) : La position géographique de l'utilisateur au moment de la consultation.

2.3.2. Modélisation des interactions de gestion des annotations (posts)

Les "posts" sont vraisemblablement les annotations ou les expériences que les utilisateurs créent et partagent.

- **addPost / editPost :**
    - **Rôle :** Trace la création d'une nouvelle annotation ou la modification d'une annotation existante.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui crée/modifie l'annotation.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de l'action* (création/modification).
        - tourId (Identifiant, visible sur le diagramme pour addPost, editPost) : L'identifiant du parcours (tour) auquel cette annotation pourrait être rattachée.
        - tags (Liste de Textes) : Une liste de mots-clés associés à l'annotation.

        - coordinates (Coordonnées) : Les coordonnées géographiques *du lieu de l'annotation elle-même* (qui peuvent différer de la position de l'utilisateur).
        - comment (Texte) : Le contenu textuel principal de l'annotation (le message). Le diagramme liste aussi message pour addPost, ce qui semble redondant avec comment mais pourrait indiquer des champs distincts.
        - placeType (Texte) : Le type de lieu concerné par l'annotation (ex: restaurant, parc).
        - postId (Identifiant, visible sur le diagramme pour editPost, implicite pour addPost) : L'identifiant unique de l'annotation.
        - imageId (Identifiant ou Chemin) : L'identifiant ou la référence d'une image associée à l'annotation.
        - shared (Booléen ou Options) : Indique les options de partage de l'annotation (ex: public, amis, privé).
        - emoticon (Texte ou Code) : Une émoticône associée à l'annotation.
        - icons (Liste de Textes ou Codes) : Une liste d'icônes associées à l'annotation.
        - timing (Texte ou Options) : Information sur le moment de l'expérience (ex: "aujourd'hui", "hier", "matin").
        - dateDisplay (Date/Heure, visible sur le tableau page 3 pour addPost/editPost, date sur le diagramme pour addPost et editPost) : La date de création (ou affichée) de l'annotation.
        - dateModif (Date/Heure, visible sur le tableau page 3 pour addPost/editPost) : La date de la dernière modification de l'annotation.
        - mask (Texte ou Options, visible sur le diagramme pour addPost, editPost): Potentiellement un filtre de visibilité ou un type spécifique de post.
        - colors (Liste de Textes/Codes, visible sur le diagramme pour addPost) : Des couleurs associées à l'annotation.
        - postType (Texte, visible sur le diagramme pour addPost, editPost) : Catégorisation plus fine du post.
- **movePost :**
    - **Rôle :** Trace le déplacement géographique d'une annotation existante.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui déplace l'annotation.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de l'action de déplacement*.
        - tourId (Identifiant) : L'identifiant du parcours auquel l'annotation appartient.
        - postId (Identifiant) : L'identifiant de l'annotation déplacée.

        - coordinates (Coordonnées) : Les *nouvelles* coordonnées géographiques de l'annotation.
        - prev_coordinates (Coordonnées) : Les *anciennes* coordonnées géographiques de l'annotation avant le déplacement.

- **deletePost :**
    - **Rôle :** Trace la suppression d'une annotation existante.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui supprime l'annotation.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de la suppression*.
        - tourId (Identifiant) : L'identifiant du parcours auquel l'annotation appartenait.
        - postId (Identifiant) : L'identifiant de l'annotation supprimée.
        - postType (Texte, visible sur le diagramme) : Le type du post supprimé.
- **openPost :**
    - **Rôle :** Trace l'action d'un utilisateur ouvrant ou consultant une annotation existante.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui ouvre l'annotation.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de l'ouverture*.
        - tourId (Identifiant) : L'identifiant du parcours auquel l'annotation appartient.
        - postId (Identifiant) : L'identifiant de l'annotation ouverte.
        - Le diagramme mentionne aussi post, emoticon, colors, postType, coordinates, imageId, date, tags, postType (répété), comment pour openPost. Cela suggère que lors de l'ouverture, des informations clés du post sont également tracées, peut-être pour éviter des jointures complexes lors de l'analyse de séquences de consultation.

2.3.3. Modélisation des interactions de gestion des parcours (Tours)

Les "tours" représentent des parcours ou des "balades" créés par les utilisateurs.

- **startTour / editTour :**
    - **Rôle :** Trace le début de la création d'un nouveau parcours ou la modification d'un parcours existant.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui démarre/modifie le parcours.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de l'action*.

        - tourId (Identifiant, implicite pour startTour, explicite pour editTour) : L'identifiant unique du parcours.
        - shared (Booléen ou Options) : Les options de partage du parcours.
        - date (Date/Heure, visible sur le diagramme) : Date de création/modification.
        - mask (Texte ou Options, visible sur le diagramme) : Similaire à celui des posts, un filtre ou un type.
        - tourName (Texte) : Le nom donné au parcours par l'utilisateur.
        - tourBody (Données de Tracé/Polyligne) : Les données géographiques constituant le tracé du parcours (série de coordonnées).
        - tourMode (Texte : "automatique" ou "manuel") : Indique si le tracé du parcours est généré automatiquement (ex: par GPS) ou dessiné manuellement.
        - dateDisplay (Date/Heure) : Le mode d'affichage de la date du parcours.
        - dateModif (Date/Heure) : La date de la dernière modification du parcours.
- **moveTour :**
    - **Rôle :** Trace la modification du tracé géographique d'un parcours existant.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui modifie le tracé du parcours.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de la modification*.
        - tourId (Identifiant) : L'identifiant du parcours modifié.
        - tourBody (Données de Tracé/Polyligne) : Le *nouveau* tracé du parcours.
        - Le diagramme mentionne aussi tour pour moveTour sous l'utilisateur, ce qui est probablement tourId.
- **deleteTour :**
    - **Rôle :** Trace la suppression d'un parcours existant.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui supprime le parcours.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de la suppression*.
        - tourId (Identifiant) : L'identifiant du parcours supprimé.
        - autoDelete (Booléen) : Indique si le parcours a été supprimé automatiquement (par exemple, par manque de données ou d'activité).

2.3.4. Modélisation des Interactions avec l'Interface et Fonctionnalités Spécifiques

Ces Obsels couvrent diverses interactions avec les fonctionnalités de l'application.

- **Filter :**
    - **Rôle :** Trace l'application de filtres par l'utilisateur pour visualiser des annotations.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui applique le filtre.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de l'application du filtre*.
        - emoticon (Texte) : Filtre basé sur une émoticône.
        - mode (Texte, visible sur le diagramme): Le mode de filtrage.
        - nationality (Texte) : Filtre basé sur la nationalité.
        - tour (Identifiant, visible sur le diagramme) : Filtre pour afficher les posts d'un tour spécifique.
        - institution (Texte) : Filtre basé sur l'institution.
        - diploma (Texte) : Filtre basé sur le diplôme.
        - tags (Liste de Textes) : Filtre basé sur des tags.
        - placeType (Texte, visible sur le tableau page 3 sous "Filter") : Filtre basé sur le type de lieu.
        - icons (Liste de Textes, visible sur le tableau page 3 sous "Filter") : Filtre basé sur des icônes.
- **seePostsMode :**
    - **Rôle :** Trace le changement du mode d'affichage des annotations par l'utilisateur.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui change le mode d'affichage.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment du changement*.
        - seeMode (Texte) : Le mode de visualisation des annotations choisi (ex: liste, carte).
        - mode (Texte) : Le mode de fonctionnement de l'application choisi par l'utilisateur (peut être lié à seeMode ou plus général).
        - layer (Texte, visible sur le diagramme sous seePostsMode, pourrait être une confusion avec selectLayer ou un type de vue spécifique).
- **selectLayer :**
    - **Rôle :** Trace la sélection d'une couche de données géographiques spécifique par l'utilisateur (ex: fond de carte, points d'intérêt).

- **AttributeTypes spécifiques :**
    - userId (Identifiant) : L'utilisateur qui sélectionne la couche.
    - position (Coordonnées) : La position géographique de l'utilisateur *au moment de la sélection*.
    - layer (Texte) : Le nom ou l'identifiant de la couche de données géographiques sélectionnée.

- **startGPSTracking :**
    - **Rôle :** Trace le moment où l'utilisateur active le suivi GPS au sein de l'application.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur qui active le GPS.
        - position (Coordonnées) : La position géographique de l'utilisateur *au moment de l'activation du GPS*.

- **goOffline :**
    - **Rôle :** Trace le passage de l'application en mode hors ligne.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur concerné.
        - position (Coordonnées) : La position géographique de l'utilisateur au moment du passage hors ligne.
        - keepon (Booléen) : Indique si l'utilisateur souhaite conserver ses données localement en mode hors ligne.

- **synchrOnline :**
    - **Rôle :** Trace la synchronisation des données de l'utilisateur avec le serveur lorsque l'application repasse en ligne ou lors d'une synchronisation manuelle.
    - **AttributeTypes spécifiques :**
        - userId (Identifiant) : L'utilisateur concerné.
        - position (Coordonnées) : La position géographique de l'utilisateur au moment de la synchronisation.
        - synchronize (Texte ou Options) : Le mode de synchronisation choisi par l'utilisateur (ex: "tout synchroniser", "synchroniser les nouveaux éléments").
        - Le diagramme mentionne user, position, synchronize sous synchrOnline.

2.3.5. Modélisation des Interactions de Réactions

- **Messages sur les annotations :**

- addMessage, editMessage, deleteMessage, openMessage.
- **Rôle :** Ces Obsels suggèrent un système de commentaires ou de messages distincts attachés aux annotations (posts). Ils suivraient une logique similaire à addPost, editPost, etc., mais appliqués à un niveau de granularité plus fin (le message/commentaire lui-même plutôt que le post entier).
- **AttributeTypes probables (par inférence) :** userId, postId (l'annotation parente), messageId (identifiant du message spécifique), content (texte du message), timestamp.

- **Réaction sur les annotations et les tours :**
  - addEmoji (sur annotation), replaceEmoji (sur annotation), removeEmoji (sur annotation).
  - addEmoji (sur tour), replaceEmoji (sur tour), removeEmoji (sur tour).
  - **Rôle :** Tracent les réactions émotionnelles (*via* emojis) des utilisateurs sur les annotations ou les parcours.
  - **AttributeTypes probables :** userId, targetId (soit postId soit tourId), emojiCode (le type d'emoji), timestamp. replaceEmoji pourrait avoir oldEmojiCode et newEmojiCode.
- **Recherche de lieu :**
  - showAddressSearch (l'utilisateur ouvre l'interface de recherche).
  - addressSearch (l'utilisateur soumet une requête de recherche).
  - selectSearchResult (l'utilisateur sélectionne un résultat de recherche).
  - **Rôle :** Tracent le processus de recherche d'adresses ou de lieux.
  - **AttributeTypes probables :**
    - showAddressSearch: userId, timestamp.
    - addressSearch: userId, searchQuery (le texte recherché), timestamp.
    - selectSearchResult: userId, searchQuery, selectedResultId (ou selectedResultCoordinates, selectedResultName), resultRank (position dans la liste), timestamp.
- **Favoris :**
  - showFavorites (l'utilisateur affiche sa liste de favoris).
  - selectFavorite (tour/post) (l'utilisateur sélectionne un favori spécifique pour le visualiser).
  - addFavorite (tour/post) (l'utilisateur ajoute un tour ou un post à ses favoris).
  - removeFavorite (tour/post) (l'utilisateur retire un tour ou un post de ses favoris).
  - **Rôle :** Tracent la gestion et l'utilisation des favoris (pour les tours et les posts).
  - **AttributeTypes probables :**
    - showFavorites: userId, timestamp.

        - selectFavorite: userId, favoriteType ("tour" ou "post"), favoriteId (tourId ou postId), timestamp.
        - addFavorite: userId, favoriteType, favoriteId, timestamp.
        - removeFavorite: userId, favoriteType, favoriteId, timestamp.

- **Notifications :**
    - showNotifications (l'utilisateur affiche la liste des notifications).
    - selectNotification (l'utilisateur interagit avec une notification spécifique).
    - hideReadNotifications (l'utilisateur masque les notifications lues).
    - **Rôle :** Tracent l'interaction de l'utilisateur avec le système de notification de l'application.
    - **AttributeTypes probables :**
        - showNotifications: userId, timestamp.
        - selectNotification: userId, notificationId, notificationType, timestamp.
        - hideReadNotifications: userId, timestamp.

- **Autres :**
    - refresh.
    - **Rôle :** Trace une action de rafraîchissement de l'affichage ou des données par l'utilisateur.
    - **AttributeTypes probables :** userId, context (quelle partie de l'application est rafraîchie), timestamp.

Cette description détaillée des Obsels et de leurs AttributeTypes constitue le cœur de la spécification du modèle de traces MOBILES. Elle fournit un dictionnaire de données précis pour toutes les interactions qui seront enregistrées, assurant que les traces collectées soient riches, structurées et prêtes pour l'analyse.

3. De la Spécification du modèle à la génération des traces

La spécification détaillée du modèle de traces, telle que présentée dans la section précédente, ne constitue pas seulement un exercice théorique. Elle est le fondement sur lequel repose la collecte effective des données d'interaction au sein de l'application MOBILES. Cette section décrit comment ce modèle est concrètement mis en œuvre et instancié par le moteur de traces intégré à l'application, aboutissant à la génération de traces d'usage structurées et conformes.

3.1. Implémentation du modèle au sein du moteur de traces de MOBILES

L'application MOBILES, développée en C# avec le moteur de jeu Unity, intègre un *moteur de traces dédié*. Ce composant logiciel a été spécifiquement conçu pour implémenter le modèle de traces décrit. Son rôle principal est d'intercepter les actions pertinentes de l'utilisateur et les événements système au sein de l'application, de les formater conformément aux définitions d'Obsels et d'AttributeTypes du modèle, puis de les enregistrer.

L'implémentation de ce moteur de traces implique typiquement :

1. **Instrumentation du code applicatif :** Des points de collecte de données (souvent appelés "sondes" ou "hooks") sont insérés aux endroits stratégiques du code source de MOBILES. Par exemple, lorsqu'un utilisateur clique sur le bouton "Ajouter un Post", une fonction du moteur de traces est appelée.
2. **Mapping des actions aux Obsels :** Le moteur de traces contient une logique qui mappe chaque action utilisateur interceptée ou événement système au type d'Obsel correspondant défini dans le modèle. Par exemple, l'action de validation du formulaire de création de post déclenchera la création d'un Obsel de type addPost.
3. **Collecte des valeurs des attributs :** Au moment où une action est interceptée, le moteur de traces collecte les valeurs des différents AttributeTypes associés à l'Obsel concerné. Cela implique d'accéder à l'état actuel de l'application, aux données saisies par l'utilisateur, à la position GPS, à l'identifiant utilisateur, etc. Par exemple, pour un Obsel addPost, le moteur récupérera le texte du message, les coordonnées, les tags, l'identifiant de l'utilisateur, etc.
4. **Gestion des attributs par défaut :** Le moteur de traces est également responsable de la génération automatique des attributs par défaut pour chaque Obsel, tels que l'ID unique de l'Obsel et les timestamps de début et de fin de l'événement. Le timestamping précis est crucial pour l'analyse séquentielle et temporelle des activités.
5. **Formatage et enregistrement :** Une fois toutes les informations collectées et structurées selon le modèle, l'Obsel est formaté et enregistré. L'enregistrement peut se faire localement sur l'appareil de l'utilisateur (surtout en mode goOffline) avant une synchronisation ultérieure (synchrOnline) vers une base de données centrale, ou directement vers un serveur si une connexion est disponible. La conformité avec la spécification KTBS peut également influencer le format de sortie ou la manière dont les traces sont ensuite stockées dans une base de traces KTBS.

Ce moteur de traces, en opérant continuellement en arrière-plan pendant l'utilisation de MOBILES, transforme les interactions volatiles des utilisateurs en données persistantes et structurées.

3.2. Caractéristiques des traces générées

L'instanciation rigoureuse du modèle de traces par le moteur intégré à MOBILES confère plusieurs caractéristiques essentielles aux traces d'usage effectivement générées :

1. **Conformité au modèle :** Chaque enregistrement de trace (chaque Obsel) est garanti conforme à la structure définie dans le modèle. Cela signifie que les types d'Obsels sont respectés, les attributs attendus sont présents (sauf s'ils sont optionnels et non renseignés), et les types de données des attributs sont corrects. Cette conformité est la clé de voûte pour toute exploitation ultérieure des données.
2. **Structure uniforme :** Grâce au modèle, toutes les traces partagent une structure uniforme, quel que soit l'utilisateur, la session, ou la version de l'application (tant que le modèle de trace lui-même n'est pas modifié). Cette uniformité simplifie considérablement le développement d'outils d'analyse, de scripts de traitement et de pipelines de données, car les analystes et les développeurs savent exactement à quel format de données s'attendre.
3. **Richesse sémantique contextualisée :** Les traces ne sont pas de simples logs techniques ; elles sont enrichies sémantiquement par les AttributeTypes spécifiques qui capturent le contexte de chaque action.

Par exemple, savoir qu'un addPost a été effectué à une position spécifique, avec certains tags et un message particulier, offre une vision beaucoup plus riche que de savoir simplement qu'un "post a été ajouté".

4. **Traçabilité des parcours utilisateurs :** La nature séquentielle et horodatée des Obsels permet de reconstituer les parcours des utilisateurs au sein de l'application, de comprendre les séquences d'actions menant à certains résultats, d'identifier les points de friction ou les fonctionnalités les plus utilisées.
5. **Préparation pour l'analyse :** Les traces générées sont, de par leur structure et leur conformité, directement prêtes (ou nécessitent un prétraitement minimal) pour diverses formes d'analyses :
    - **Analyses statistiques :** Fréquence d'utilisation des fonctionnalités, temps passé sur certaines tâches, etc.
    - **Analyses séquentielles :** Identification de motifs d'usage, modélisation de processus.
    - **Analyses géospatiales :** Étude des déplacements, des zones d'activité privilégiées, des relations entre annotations et lieux.
    - **Analyses comportementales :** Segmentation des utilisateurs en fonction de leurs profils d'activité, compréhension des stratégies d'exploration.

En résumé, le moteur de traces de MOBILES, en instanciant fidèlement le modèle de trace spécifié, agit comme un pont essentiel entre les interactions dynamiques des utilisateurs et la création d'un corpus de données structurées, fiables et hautement exploitables pour la recherche et l'amélioration de l'application.

4. Valeur et implications du modèle de traces défini

La définition et l'implémentation rigoureuses du modèle de traces pour l'application MOBILES ne sont pas une fin en soi, mais un moyen stratégique pour atteindre des objectifs d'analyse, de compréhension et d'amélioration. Ce modèle, en structurant la collecte des données d'interaction, apporte une valeur considérable au projet MOBILES et a des implications significatives pour l'exploitation future des traces générées.

4.1. Apports spécifiques du modèle de traces pour la collecte et la qualité des données

Le modèle de traces est le garant de la qualité et de la pertinence des données collectées. Ses apports spécifiques incluent :

1. **Standardisation et cohérence des traces :** C'est l'un des bénéfices les plus directs. En imposant une structure commune à tous les enregistrements (Obsels), le modèle élimine l'hétérogénéité qui pourrait provenir de différentes interprétations ou d'implémentations ad-hoc de la collecte. Chaque type d'action est tracé de la même manière, avec les mêmes attributs, assurant une base de données de traces uniforme. Cette standardisation est cruciale, notamment dans un contexte de développement itératif où différentes versions de l'application pourraient coexister ou se succéder.
2. **Exhaustivité et précision des informations capturées :** Grâce à la définition minutieuse des Obsels et de leurs AttributeTypes, le modèle vise à capturer un ensemble complet et précis d'informations pour chaque interaction jugée pertinente. Des aspects comme la position géographique de l'utilisateur lors d'une action, les tags associés à un post, le tourMode pour un parcours, ou même les prev_coordinates

lors d'un movePost, apportent une profondeur de détail qui serait difficile à obtenir sans une spécification claire en amont.

3. **Fiabilité des données :** Un modèle bien défini et correctement implémenté dans un moteur de traces réduit les risques d'erreurs, d'omissions ou d'incohérences dans les données collectées. Cela renforce la confiance dans les traces et, par conséquent, dans les analyses qui en découleront.
4. **Facilitation de la compréhension et de l'interprétation des traces :** Le modèle agit comme un dictionnaire de données partagé. Les chercheurs, les analystes ou les développeurs qui travailleront avec les traces disposent d'une référence claire sur la signification de chaque Obsel et de chaque attribut. Cela accélère la prise en main des données et minimise les risques de mauvaise interprétation. Le diagramme et les tableaux descriptifs du modèle sont des outils essentiels pour cette compréhension.

4.2. Potentiel d'exploitation des traces structurées par ce modèle

Les traces d'usage, collectées conformément au modèle spécifié, ouvrent un large champ de possibilités d'exploitation, allant bien au-delà d'un simple monitoring technique. La structure et la richesse sémantique des traces permettent notamment :

1. **Analyse comportementale approfondie :**
    - **Identification de profils utilisateurs :** En analysant les séquences d'actions (addPost, startTour, Filter, etc.) et la nature des contenus créés, il devient possible de segmenter les utilisateurs en différents profils (par exemple, "explorateurs intensifs", "annoteurs sociaux", "documentaristes solitaires").
    - **Compréhension des stratégies d'exploration et d'annotation :** Comment les étudiants découvrent-ils la ville ? Quelles fonctionnalités utilisent-ils pour documenter leurs expériences ? Y a-t-il des séquences d'actions typiques ?
    - **Étude de l'adoption des fonctionnalités :** Quelles sont les fonctionnalités les plus populaires, les moins utilisées, ou celles qui posent des difficultés (identifiables par des séquences d'actions répétitives ou des abandons) ?
2. **Amélioration de l'expérience utilisateur (UX) et de l'application :**
    - **Identification des points de friction :** Des analyses de séquences peuvent révéler des étapes où les utilisateurs hésitent, reviennent en arrière, ou abandonnent une tâche, signalant des problèmes d'ergonomie.
    - **Optimisation des parcours utilisateurs :** Comprendre comment les utilisateurs naviguent dans l'application permet d'optimiser les interfaces et les flux de travail.
    - **Personnalisation :** À terme, la connaissance fine des préférences et comportements individuels (déduite des traces) pourrait permettre de proposer des contenus, des suggestions ou des fonctionnalités personnalisées.
3. **Recherche sur l'expérience des étudiants internationaux :**
    - **Analyse de l'appropriation de l'espace urbain :** Les données de position de l'utilisateur et des coordonnées associées aux posts et aux tours peuvent cartographier les zones explorées, les

types de lieux fréquentés, et l'évolution de cette exploration dans le temps (grâce à arrivalDate et aux timestamps).

- **Étude des pratiques d'annotation et de partage :** Quels types d'expériences sont partagés (comment, emoticon, placeType) ? Avec qui (shared) ?
- **Impact de l'application sur l'intégration et la découverte :** Les traces peuvent fournir des indicateurs indirects de la manière dont MOBILES aide les étudiants dans leur processus d'adaptation.

4. **Maintenance et évolution de l'application :**
   - **Détection de bugs ou de comportements anormaux :** Des séquences d'Obsels inhabituelles ou des valeurs d'attributs aberrantes peuvent aider à identifier des problèmes techniques.
   - **Prise de décision basée sur les données :** Les informations issues de l'analyse des traces peuvent guider les choix pour les futures évolutions de MOBILES (quelles fonctionnalités développer, améliorer ou supprimer).

4.3. Limitations du modèle actuel et perspectives d'évolution

Bien que complet et spécifiquement adapté à MOBILES, tout modèle de traces est une abstraction et possède des limites inhérentes, ainsi que des pistes d'évolution possibles :

1. **Granularité des traces :** Le modèle actuel capture un ensemble défini d'actions. Il est possible que certaines interactions plus fines (par exemple, le temps passé à regarder une carte sans interagir, les défilements précis, les hésitations avant un clic) ne soient pas tracées si elles n'ont pas été jugées initialement pertinentes ou trop coûteuses à collecter. Une évaluation future pourrait identifier le besoin d'une granularité plus fine pour certaines analyses.
2. **Capture de l'intention ou de la satisfaction :** Le modèle se concentre sur les actions observables ("quoi", "où", "quand"). Il ne capture pas directement l'intention de l'utilisateur ("pourquoi") ni son niveau de satisfaction ou de frustration, sauf de manière très indirecte (par exemple, *via* l'analyse).
3. **Évolutivité du modèle :** L'application MOBILES est susceptible d'évoluer, avec l'ajout de nouvelles fonctionnalités ou la modification des existantes. Le modèle de traces devra alors être mis à jour en conséquence pour maintenir sa pertinence et son exhaustivité. Cela implique un processus de gestion des versions du modèle et potentiellement une rétrocompatibilité à considérer pour l'analyse des anciennes traces.
4. **Volume et gestion des données :** Un traçage détaillé peut générer un volume important de données, surtout avec un grand nombre d'utilisateurs actifs. Des stratégies efficaces de stockage, de traitement et d'archivage des traces sont nécessaires. Bien que cela ne soit pas une limitation du modèle lui-même, c'est une implication directe de son utilisation.
5. **Contextes d'usage non capturés :** Le modèle se focalise sur les interactions *au sein* de l'application. Des facteurs externes influençant l'usage (par exemple, l'utilisateur est-il seul ou en groupe, quel est son état émotionnel général, les conditions météorologiques) ne sont pas directement tracés, bien que certains pourraient être inférés ou collectés par d'autres moyens.

En dépit de ces points, le modèle de traces actuel fournit une base solide et extrêmement précieuse. Les perspectives d'évolution visent à l'enrichir et à l'adapter continuellement pour maximiser sa valeur pour le projet MOBILES.

5. Conclusion

Ce rapport a présenté la spécification détaillée et la justification du modèle de traces conçu et implémenté pour l'application MOBILES. Ancré dans les principes du Knowledge Trace Base System (KTBS), ce modèle constitue le schéma directeur pour la collecte systématique et structurée des données d'interaction des étudiants internationaux utilisant l'application pour explorer leur environnement urbain, annoter leurs expériences et documenter leurs parcours.

Nous avons détaillé les principes fondateurs de ce modèle, sa structure générale incluant les attributs par défaut de chaque observation (Obsel), et fourni un catalogue exhaustif des différents types d'Obsels et de leurs attributs (AttributeTypes) spécifiques. De la gestion des comptes utilisateurs à la création et la manipulation d'annotations (posts) et de parcours (tours), en passant par les interactions avec les fonctionnalités de filtrage, de sélection de couches cartographiques ou de suivi GPS, chaque action clé a été soigneusement modélisée pour assurer une capture d'information riche et contextuelle. L'inclusion des "Nouveaux types" d'interactions, tels que les messages, les réactions émotionnelles, les recherches de lieux, la gestion des favoris et des notifications, démontre la volonté d'une couverture extensive des usages.

Il a été souligné que ce modèle de traces n'est pas un simple artefact théorique, mais qu'il est activement instancié par un moteur de traces dédié au sein de l'application MOBILES, développée en C# avec Unity. Ce mécanisme assure la génération de traces d'usage conformes, normalisées et prêtes pour l'analyse, transformant les interactions éphémères en un corpus de données structurées et fiables.

La valeur de ce modèle de traces réside dans sa capacité à garantir la qualité, la cohérence et l'exhaustivité des données collectées. Ces données constituent une ressource inestimable pour une multitude d'exploitations : de l'analyse comportementale approfondie des utilisateurs à l'amélioration continue de l'expérience utilisateur et des fonctionnalités de MOBILES, en passant par des recherches académiques sur l'appropriation de l'espace par les étudiants internationaux. Si des limitations et des perspectives d'évolution du modèle ont été identifiées, notamment en termes de granularité ou de capture d'intentions, le cadre actuel offre déjà une fondation robuste.

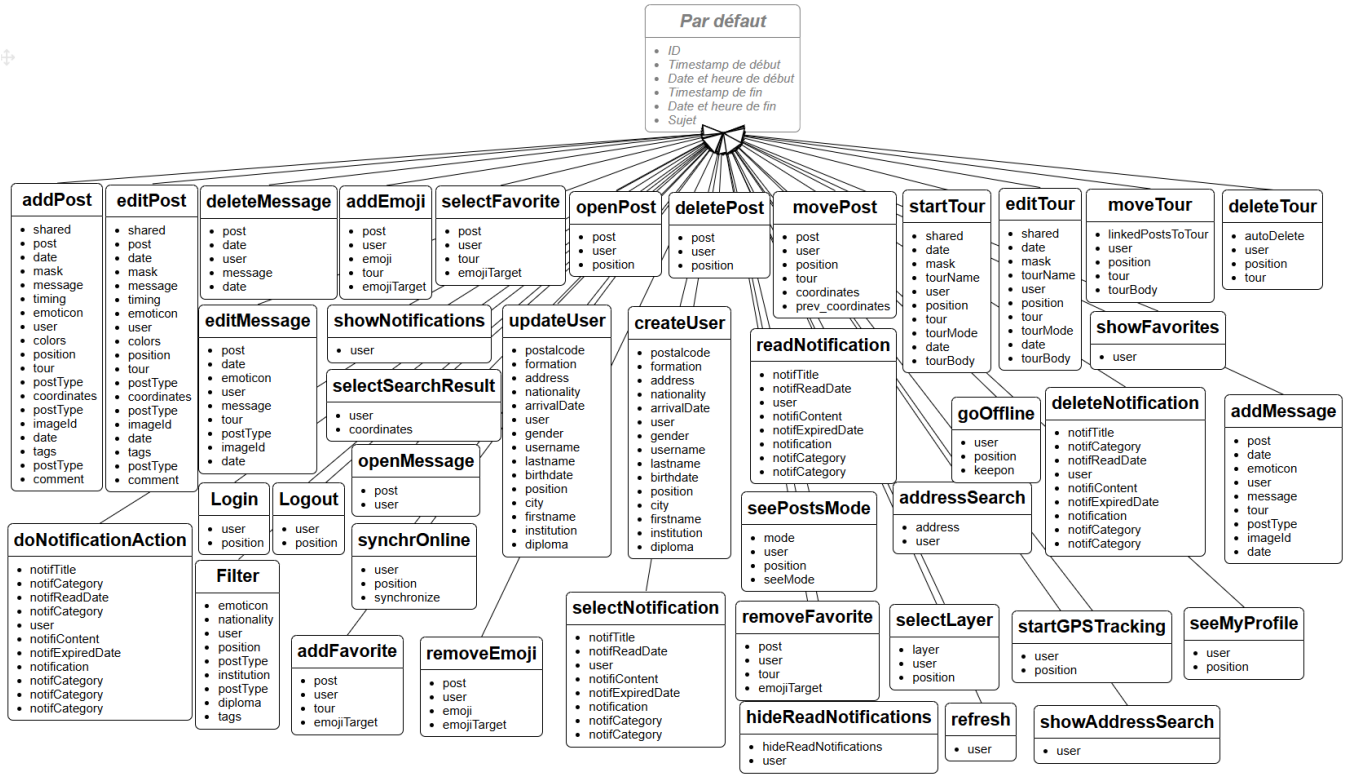
En conclusion, le modèle de traces de MOBILES est un composant fondamental de l'écosystème de l'application. Il est le prérequis essentiel pour comprendre finement les usages, pour évaluer l'impact de l'application et pour guider son évolution future de manière éclairée et fondée sur des données probantes. Les travaux futurs s'orienteront naturellement vers l'exploitation analytique de ce riche corpus de traces, afin de concrétiser pleinement le potentiel de compréhension et d'amélioration qu'il recèle pour le projet MOBILES et ses utilisateurs.

6. Annexes

Les annexes suivantes fournissent les détails visuels et tabulaires de la spécification du modèle de trace MOBILES, tels que référencés tout au long de ce rapport.

Annexe A : Diagramme du Modèle de Trace MOBILES

Cette annexe présente le diagramme complet illustrant les différents types d'Obsels (observations) et leurs relations, ainsi que les attributs par défaut et certains attributs spécifiques.



Annexe B : Tableaux des AttributeTypes par Type d'Obsel

Cette annexe contient les tableaux qui listent de manière exhaustive les AttributeTypes pour chaque type d'Obsel défini dans le modèle de traces, accompagnés de leur explication.

Type	Attributs	Explication
createUser / updateUser	arrivalDate	la date d'arrivée de l'utilisateur dans la ville.
	gender	le genre de l'utilisateur.
	nationality	la nationalité de l'utilisateur.
	formation	la formation de l'utilisateur.
	position	la position géographique de l'utilisateur.
	lastname	le nom de famille de l'utilisateur.
	firstname	le prénom de l'utilisateur.
	birthdate	la date de naissance de l'utilisateur.
	address	l'adresse de l'utilisateur.
	city	la ville de l'utilisateur.
	postalcode	le code postal de l'utilisateur.
	username	le nom d'utilisateur de l'utilisateur.
Login	userId	l'identifiant de l'utilisateur qui se connecte à l'application.
	position	la position géographique de l'utilisateur qui se connecte à l'application.
Logout	userId	l'identifiant de l'utilisateur qui se déconnecte de l'application.
	position	la position géographique de l'utilisateur qui se déconnecte de l'application.
seeMyProfile	userId	l'identifiant de l'utilisateur qui consulte son profil.

	position	la position géographique de l'utilisateur qui consulte son profil.
goOffline	userId	l'identifiant de l'utilisateur qui passe en mode hors ligne.
	position	la position géographique de l'utilisateur qui passe en mode hors ligne.
	keepon	si l'utilisateur souhaite garder ou non ses données en mode hors ligne.
synchrOnline	userId	l'identifiant de l'utilisateur qui synchronise ses données en ligne.
	position	la position géographique de l'utilisateur.
	synchronize	le mode de synchronisation choisi par l'utilisateur.
Filter	userId	l'identifiant de l'utilisateur qui filtre les annotations selon ses critères.
	position	la position géographique de l'utilisateur qui filtre les annotations selon ses critères.
	tags	liste de tags utilisés pour filtrer les annotations.
	placeType	le type de lieu utilisé pour filtrer les annotations.
	emoticon	l'émoticône utilisé pour filtrer les annotations.
	icons	les icônes utilisés pour filtrer les annotations.
	nationality	la nationalité utilisée pour filtrer les annotations.
	diploma	le diplôme utilisé pour filtrer les annotations.
	institution	l'institution utilisée pour filtrer les annotations.
seePostsMode	userId	l'identifiant de l'utilisateur qui choisit le mode d'affichage des annotations.
	position	la position géographique de l'utilisateur qui choisit le mode d'affichage des annotations.
	seeMode	le mode de visualisation des annotations choisi par l'utilisateur.
	mode	le mode de fonctionnement de l'application choisi par l'utilisateur.
selectLayer	userId	l'identifiant de l'utilisateur qui sélectionne une couche de données géographiques.
	position	la position géographique de l'utilisateur qui sélectionne une couche de données géographiques.
	layer	le nom de la couche de données géographiques sélectionnée par l'utilisateur.
startGPSTracking	userId	l'identifiant de l'utilisateur qui démarre le suivi GPS.
	position	la position géographique de l'utilisateur qui démarre le suivi GPS.
addPost / editPost	userId	l'identifiant de l'utilisateur qui ajoute/modifie une annotation.
	position	la position géographique de l'utilisateur qui ajoute/modifie une annotation.
	tourId	l'identifiant de la ballade auquel l'annotation appartient.
	tags	liste de tags de l'annotation.
	coordinates	les coordonnées du lieu de l'annotation.
	comment	le commentaire de l'annotation.
	placeType	le type de lieu de l'annotation.
	postId	l'identifiant de l'annotation.
	imageId	l'identifiant de l'image de l'annotation.
	shared	les options de partage de l'annotation.
	emoticon	l'émoticône de l'annotation.
	icons	liste d'icônes de l'annotation.
	timing	le type de l'expérience renseignée par l'annotation.
	dateDisplay	la date de création (affichage) associée.

	dateModif	la date de modification de l'annotation.
	message	le message de l'annotation.
movePost	userId	l'identifiant de l'utilisateur qui déplace une annotation existante.
	position	la position géographique de l'utilisateur qui déplace une annotation existante.
	tourId	l'identifiant de la ballade auquel l'annotation appartient.
	coordinates	les nouvelles coordonnées du lieu de l'annotation.
	postId	l'identifiant de l'annotation.
	prev_coordinates	les anciennes coordonnées du lieu de l'annotation.
deletePost	userId	l'identifiant de l'utilisateur qui supprime une annotation existante.
	position	la position géographique de l'utilisateur qui supprime une annotation existante.
	tourId	l'identifiant de la ballade auquel l'annotation appartient.
	postId	l'identifiant de l'annotation.
openPost	userId	l'identifiant de l'utilisateur qui ouvre une annotation existante.
	position	la position géographique de l'utilisateur qui ouvre une annotation existante.
	tourId	l'identifiant de la ballade auquel l'annotation appartient.
	postId	l'identifiant de l'annotation.
startTour / editTour	userId	l'identifiant de l'utilisateur qui démarre une ballade.
	position	la position géographique de l'utilisateur qui démarre une ballade.
	tourId	l'identifiant de la ballade.
	shared	les options de partage de la ballade.
	tourName	le nom de la ballade.
	tourBody	le tracé de la ballade.
	tourMode	le mode de la ballade (automatique ou manuel).
	dateDisplay	le mode d'affichage de la date de la ballade.
	dateModif	la date de modification de la ballade.
moveTour	userId	l'identifiant de l'utilisateur qui déplace une ballade existante.
	position	la position géographique de l'utilisateur qui déplace une ballade existante.
	tourId	l'identifiant de la ballade.
	tourBody	le nouveau tracé de la ballade.
deleteTour	userId	l'identifiant de l'utilisateur qui supprime une ballade existante.
	position	la position géographique de l'utilisateur qui supprime une ballade existante.
	tourId	l'identifiant de la ballade.
	autoDelete	si la ballade a été supprimée automatiquement par manque de données.