Mise en place d'une stratégie de diffusion et d'échange des données contenues dans l'outil Geotrek

Parc national des Cévennes

Idrissa Djepa Creutz

M2 Géographies Numériques

septembre 2021

Encadrant·es :

Amandine Sahl (amandine.sahl@cevennes-parcnational.fr)

Kisito Cendrier (kisito.cendrier@cevennes-parcnational.fr)

Juliette Wettstein (juliette.wettstein@cevennes-parcnational.fr)

Tuteur universitaire :

Thierry Joliveau (thierry.joliveau@univ-st-etienne.fr)

GÉO NUM

Parc national des Cévennes

Université de Lyon

Université Jean Monnet Saint-Étienne

Université Lumière Lyon 2

ENS de Lyon

Remerciements

Je tiens à remercier Amandine, Kisito et Juliette pour avoir accueilli ma demande de stage et m'avoir si bien encadré, pour les biscuits au chocolat et les points quotidiens autour du pingouin. Merci à tou·tes les agent·es du parc pour leur accueil si bienveillant, le partage de leurs connaissances et leurs sourires.

Merci à Camille Monchicourt et Cendrine Hoarau du Parc national des Écrins (et du master géomatique!) pour les nombreux échanges.

Merci à Thierry Joliveau pour son suivi du stage et ses conseils éclairants, qui a été là pour répondre à mes questions. Merci à lui, Claire Cunty, Hélène Mathian et Luc Merchez pour m'avoir permis cette énième (ultime?) réorientation universitaire. Des études sur le genre à la géomatique la passerelle n'était ni limpide, ni des plus courtes, mais si je ne regrette aucune des années d'études qui m'ont mené ici, c'est bien grâce à ces équipes successives d'enseignant·es passionné·es que j'ai eu la chance d'avoir ! Je remercie également Sabine Loudcher et les enseignant·es du master Humanités Numériques de Lyon, qui m'ont initié à la programmation et propulsé à leur insu vers la géomatique.

Merci à mes camarades de master de m'avoir si bien accueilli, c'était bien trop court, mais nos chemins se recroiseront, le monde de la géomatique n'est pas si grand !

Merci aussi à ma famille qui ne s'étonne même plus chaque année de me voir plonger dans un nouveau master, et qui a toujours soutenu sans réserve mes choix d'études.

Merci à Léo, Blandine, Paul, Estelle, Manon, Maxence, Tifenn, Alicia...pour tous ces moments estivaux.

Bien sûr je pense à Lise, Manon et Céline sans qui ce séjour au pied du Causse n'aurait pu être aussi inoubliable : à nos soirées jeux sans jeux, à nos dégustations de pélardons, nos pauses déjeuner à la rivière, aux plaskis, carbonara et mousses au chocolat...à la Principauté de la Grézotière !

Lise, à tes fous rires de bonheur qui t'empêchent de dormir et tes questions quotidiennes. Manon, à ta passion de la nature et ta capacité à nous supporter. Céline, à notre potager et surtout...*What a life, what a night. What a beautiful, beautiful ride !*

Merci à vous de continuer à me faire confiance pour vous guider en randonnée, vous n'avez pas fini d'en voir des vertes et des pas mûres…

Enfin merci à la Lozère d'être Lozère, aux Cévennes d'être Cévennes, et au Causse d'être Méjean.

Résumé

Le Parc national des Cévennes est engagé dans plusieurs processus d'échange de données touristiques et de gestion. Le logiciel libre Geotrek est utilisé pour stocker, gérer et valoriser ces données au quotidien, par exemple via le site web Destination Cévennes. La diffusion de Geotrek chez les partenaires du Parc national comme les conseils départementaux de la Lozère et du Gard ouvrent des discussions sur les flux de données à mettre en place entre les instances du logiciel. Non prévu pour la communication entre instances, Geotrek manque de fonctionnalités pour le travail entre partenaires territoriaux. La création, à l'initiative du Parc national des Écrins, d'un schéma de données sur les itinéraires de randonnée est l'occasion pour le Parc national des Cévennes de commencer à ouvrir ses données, comme le veut la loi.

Mon stage a ainsi permis de publier une première version de ce schéma de données, de créer un prototype d'agrégateur de données de Geotrek et d'explorer la question de la fusion de réseaux linéaires.

Mots-clefs : schéma de données, randonnées, Parc national des Cévennes, parcs nationaux, Lozère, Gard

Abstract

The Cévennes national Park is active in several processes of exchange of tourist and management data. It uses the free software Geotrek to store, manage and enhance these data on a daily basis, for instance via the website Destination Cévennes. The distribution of Geotrek among the National Park's partners, such as the departmental councils of Lozère and Gard, has opened up discussions on the data flows to be set up between the software's instances. Geotrek is not designed for communication between instances, and lacks functionalities for work between territorial partners. The creation, at the initiative of the Écrins national Park, of a data schema of hiking routes is an opportunity for the Cévennes national Park to start opening its data, as required by law.

My internship allowed me to publish a first version of this data schema, to create a prototype of Geotrek data aggregator and to explore the issue of merging linear networks.

Keywords: data standard, hiking routes, Cévennes national Park, National parks, Lozère, Gard

Choix d'écriture

J'ai choisi d'utiliser dans ce rapport de stage une forme d'écriture appelée « inclusive » ou encore « non-discriminante ». Partant du constat que la langue française considère dans un certain nombre de cas le masculin comme neutre[1], que le langage participe à façonner nos représentations du monde[2], et que certaines règles et usages de grammaire ou d'orthographe ont une origine fondée non sur des raisons linguistiques mais sexistes[3], j'ai adopté des usages que je m'efforce de respecter afin de rétablir une forme d'égalité dans les discours. Ces usages répondent à plusieurs principes :

- l'utilisation des noms féminins lorsqu'ils qualifient une femme (ex : professeure, coureuse, entraîneuse) ;
- la préférence donnée à l'accord de proximité des adjectifs plutôt qu'au masculin neutre (ex : les ajouts et les modifications sont enregistrées) ;
- ne pas utiliser « Homme » ou ses déclinaisons pour qualifier le genre humain ou un ensemble de personnes de genres différents ;
- l'usage du point médian entre le suffixe et le reste du mot lorsque le suffixe marquant le féminin constitue un ajout par rapport au suffixe marquant le masculin, et que le retrait du point médian donnerait le mot féminin (ex : les agent·es du Parc national). Un seul point médian est utilisé, et pas deux comme on peut le voir régulièrement, pour éviter de surcharger la lecture ;
- l'usage de la barre oblique lorsque le suffixe marquant le masculin est trop différent du suffixe marquant le féminin et que le point médian induirait une fausse continuité dans la lecture (ex : les administrateurs/trices).

1 Académie Française (2014). La féminisation des noms de métiers, fonctions, grades ou titres - Mise au point de l'Académie française. Repéré à http://www.academie-francaise.fr/actualites/la-feminisation-des-noms-de-metiers-fonctions-grades-ou-titres-mise-au-point-de-lacademie Consulté le 03/09/2021

2 Tornay, S. (1973). Langage et perception. La dénomination des couleurs chez les Nyangatom du Sud-Ouest éthiopien. *Homme*, *13*(4), 66-94. https://doi.org/10.3406/hom.1973.367381

3 Académie Française (2019). La féminisation des noms de métier et de fonction. Repéré à http://www.academie-francaise.fr/sites/academie-francaise.fr/files/rapport_feminisation_noms_de_metier_et_de_fonction.pdf Consulté le 03/09/2021

Table des matières

1. Introduction

Des onze parcs nationaux français, celui des Cévennes est spécifique à bien des égards. Plus grand parc de France métropolitaine (937 km carrés), il est également un des plus visités (800 000 personnes par an) et un des seuls dont la zone cœur est habitée et exploitée de manière permanente (agriculture, élevage, chasse)[4].

Une des missions du Parc national étant de promouvoir son territoire et d'accompagner la découverte de celui-ci, une plateforme dédiée a été lancée en 2016 : Destination Cévennes[5]. Elle permet aux touristes de trouver des idées de randonnées, des hébergements où loger, des prestataires organisant des activités de pleine nature, etc.

Derrière cette plateforme il y a Geotrek[6], un logiciel qui permet, en plus de cette valorisation touristique, d'assurer la gestion d'un réseau de chemins (foncier, entretien, signalétique, autorisations de passage…) via une application web accessible aux agent·es.

Mon stage s'est concentré sur les problématiques de partage et d'échange des données de Geotrek.

Tout d'abord, l'obligation réglementaire d'ouverture des données pour les établissements publics nous a amené à publier les itinéraires de randonnée du Parc national sous licence ouverte. Nous avons profité de la constitution d'un groupe de travail sur un schéma de données des itinéraires de randonnées pour avancer sur ce point.

Ensuite, alors que Geotrek est adopté par un nombre grandissant d'organisations à l'échelle nationale (collectivités locales, parcs naturels régionaux, agences de tourisme…), la question de l'interaction entre plusieurs Geotrek mitoyens, voire couvrant un même territoire, se pose avec plus d'acuité que jamais. L'installation de Geotrek est par nature décentralisée, le code source étant disponible sur internet, et chaque organisation est libre de le faire évoluer selon ses besoins. Cette force cache aussi une faiblesse, qui apparaît maintenant que de nombreuses organisations utilisatrices souhaitent faire communiquer leur Geotrek, car les fonctionnalités permettant à plusieurs instances d'interagir sont quasiment inexistantes.

Les conseils départementaux de la Lozère et du Gard (les deux principaux départements dans lesquels se trouve le Parc national) faisant partie de ces utilisateurs récents, il devient crucial pour le Parc national de mettre en place un partage des outils et des méthodes d'échange de données avec ces deux importants partenaires. Deux principaux aspects sont à creuser : d'abord la mise en commun des données de gestion et des données touristiques (itinéraires, signalétique, foncier…), puis la modification du réseau linéaire de référence en intégrant des nouveaux tronçons.

4 https://www.cevennes-parcnational.fr/fr/le-parc-national-des-cevennes/un-territoire-reconnu
5 https://destination.cevennes-parcnational.fr
6 https://geotrek.fr/

2. Le Parc national des Cévennes

Le territoire du Parc national est entièrement situé en moyenne montagne, ce qui le distingue de ses homologues de haute montagne (Pyrénées, Écrins, Vanoise, Mercantour), maritimes (Port-Cros, Calanques), tropicaux (Guyane, Guadeloupe, La Réunion) ainsi que de plaine pour le tout nouveau Parc de Forêts. Cela explique la spécificité de sa zone cœur habitée.

Géré par un établissement public à caractère administratif depuis sa création en 1970, ce territoire est divisé en deux zones à statut différent[7] :

- la zone cœur (937 km²) : fixée par décret ministériel, c'est la zone sur laquelle le parc a un pouvoir réglementaire et de police. Elle est à cheval sur les départements de la Lozère et du Gard.

- l'aire d'adhésion (2035 km²) : elle correspond au territoire de toutes les communes ayant adhéré à la charte du parc, celles-ci devant être comprises dans l'aire optimale d'adhésion. En plus des deux départements précédemment cités, l'Ardèche y est représentée par trois communes.

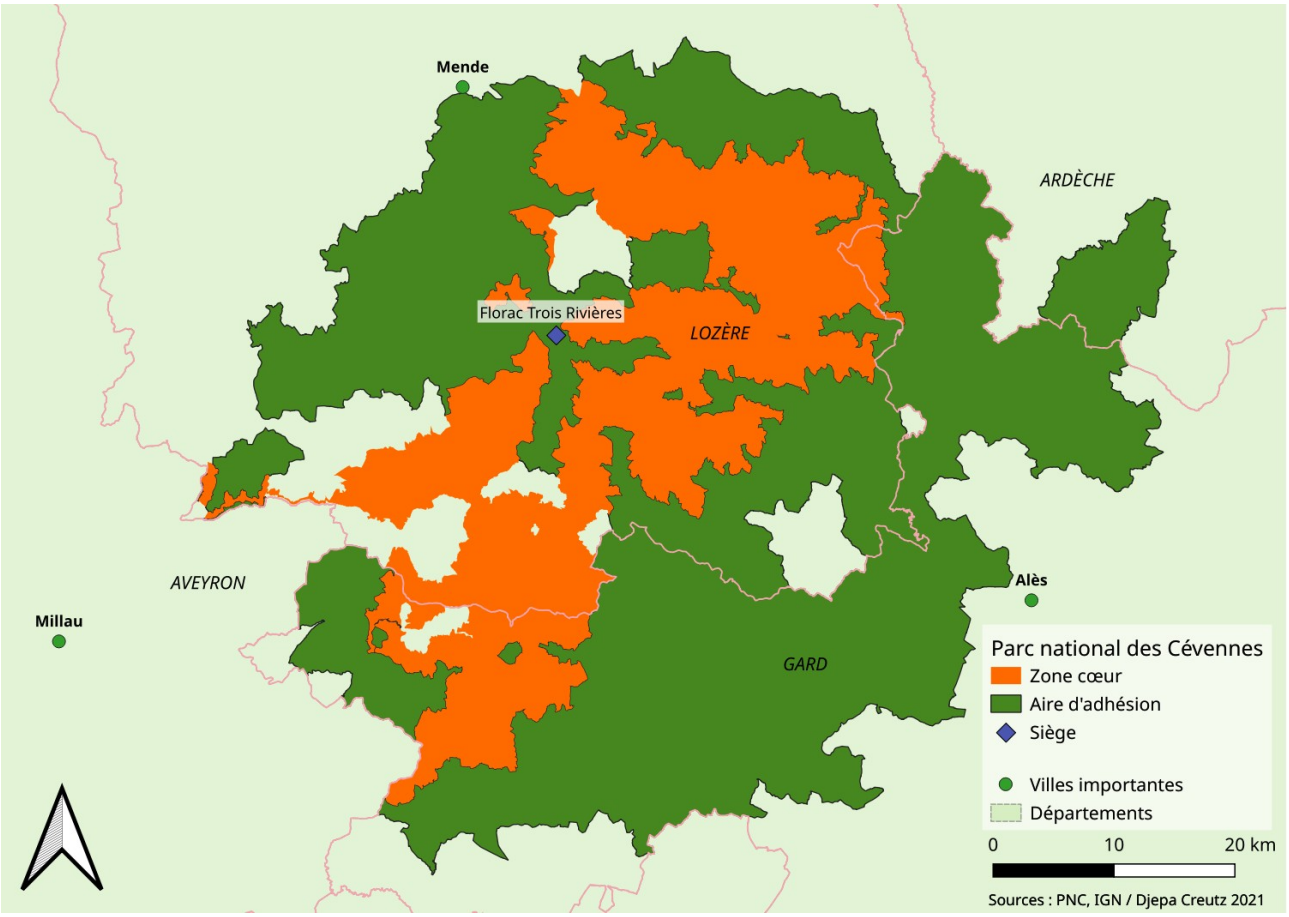


Figure 1: Parc national des Cévennes

7 https://www.cevennes-parcnational.fr/fr/le-parc-national-des-cevennes/un-territoire-reconnu

La charte est un projet de territoire d'une durée de quinze ans[8]. Sa signature par un conseil municipal implique trois engagements minimaux prévus par la loi (compatibilité des documents d'urbanisme avec la charte, réglementation de la circulation des véhicules à moteur, interdiction de la publicité), des engagements collectifs, et des engagements individuels. L'adhésion est libre : une commune peut avoir la majeure partie de son territoire en zone cœur et ne pas faire partie de l'aire d'adhésion, ou inversement être éloignée de la zone cœur et adhérer à la charte.

Dirigé par un Conseil d'Administration[9] nommé par le ministère en charge de l'écologie, le Parc national est géré par une équipe de près d'une centaine d'agent·es réparti·es dans trois grands services[10]:

- Le service Développement durable (SDD) accompagne les personnes dans leurs projets d'activité (agriculture, sylviculture, gestion de l'eau, chasse...) ou d'aménagement (architecture, travaux...)
- Le service Connaissance et Veille du territoire (SCVT) a pour mission la connaissance, la surveillance et la protection des patrimoines naturel et culturel.
- Le service Accueil et Sensibilisation (SAS) travaille à destination des habitant·es, des touristes et des scolaires pour les sensibiliser aux richesses du Parc national

Le pôle Système d'Information (SI) au sein duquel je travaillais, composé d'Amandine Sahl, administratrice de données, Kisito Cendrier, chargé de mission SIG, Frédéric Fidon, administrateur réseau et Joël Clément, développeur informatique, bien que transversal, est rattaché au SCVT. En effet, la majorité des données produites et exploitées par le parc sont d'ordre naturaliste.

Les activités du pôle SI tournent autour de quelques points principaux :

- gestion des données naturalistes : l'application GeoNature[11] est utilisée par beaucoup d'agent·es pour recenser leurs observations naturalistes et produire des analyses à partir de celles-ci.

- support aux agent·es : formations à l'utilisation de QGIS, à la création de cartes ou à l'utilisation d'applications comme GeoNature, Geotrek ou des applications de terrain.

- édition cartographique : production de cartes pour les besoins de communication du Parc national et de ses partenaires, des rapports scientifiques, etc.

- gestion des données touristiques : surtout via Geotrek

8 http://www.cevennes-parcnational.fr/fr/le-parc-national-des-cevennes/la-charte
9 http://www.cevennes-parcnational.fr/fr/le-parc-national-des-cevennes/letablissement-public/la-gouvernance
10 http://www.cevennes-parcnational.fr/fr/le-parc-national-des-cevennes/letablissement-public/missions-et-organisation
11 https://geonature.fr/

- maintien et développement des applications existantes : mise à jour, prise en charge des problèmes, participation au développement de GeoNature, Geotrek et TaxHub[12] (application de gestion des taxons)…

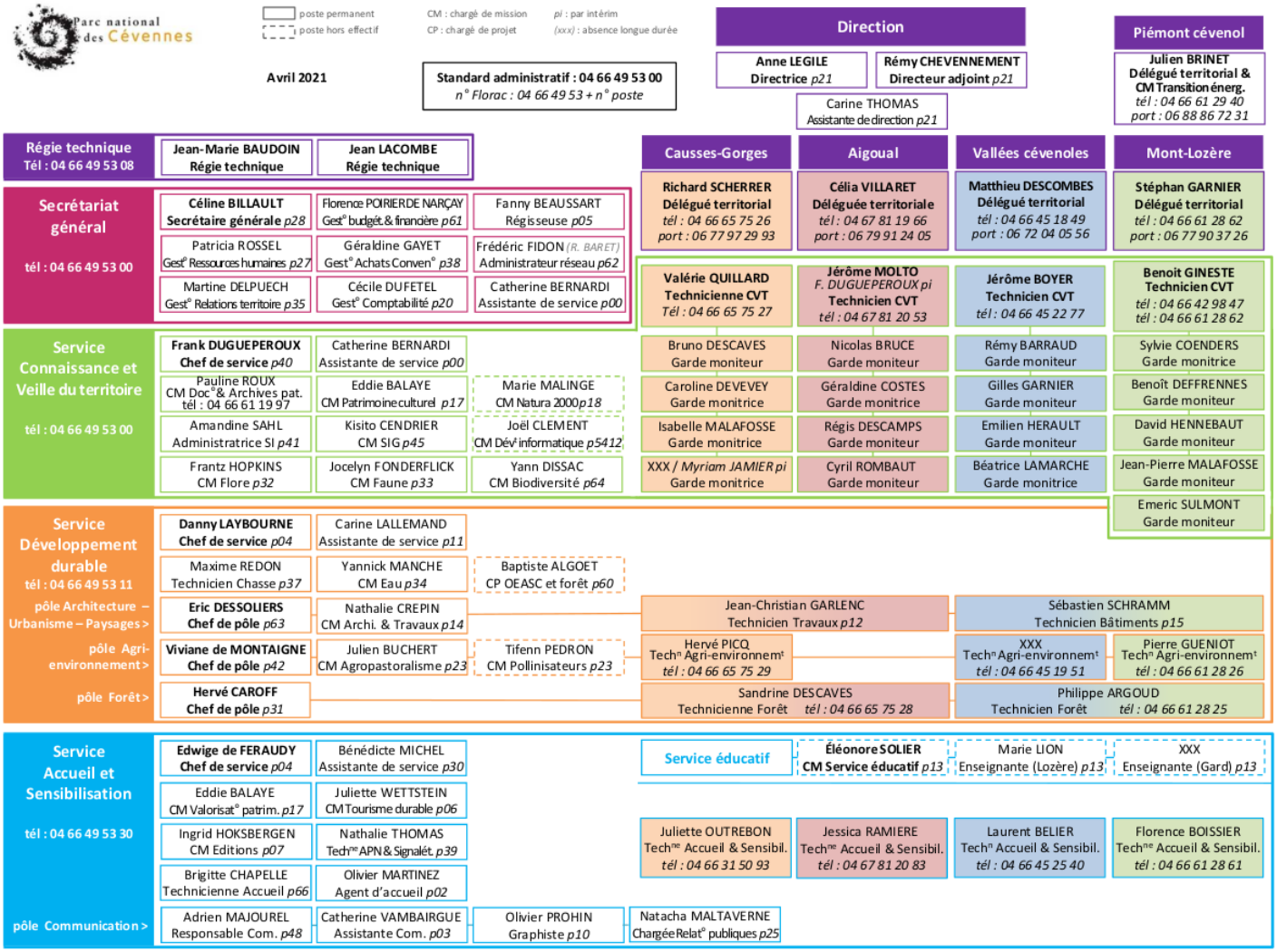


Figure 2: Organigramme 2021 du Parc national des Cévennes

Mon stage était tutoré par trois personnes :

- Amandine Sahl, qui a assuré la majorité de l'accompagnement en programmation ;

- Kisito Cendrier, qui a, entre autres, apporté son expérience de gestion du réseau linéaire de Geotrek ;

- Juliette Wettstein, chargée de mission tourisme durable, qui a assuré le lien avec les besoins métiers du SAS autour de Geotrek.

L'alliance de ces compétences et expertises diverses m'a permis d'appréhender les différentes missions dans leur globalité et de les mener plus efficacement.

12 https://github.com/PnX-SI/TaxHub

3. Geotrek

Geotrek est un logiciel libre développé en 2012 par l'entreprise Makina Corpus[13] à l'initiative de deux Parcs nationaux français (Écrins et Mercantour) et d'un italien (*Alpi Marittime*). Son code source est disponible sur GitHub[14] sous licence BSD-2 Clause[15], qui autorise l'usage commercial, la redistribution, la modification et l'usage privé.

Il est constitué de deux applications distinctes qui fonctionnent ensemble (plus une application mobile) : Geotrek-admin et Geotrek-rando.

3.1 Geotrek-admin

C'est l'application métier, accessible aux agent·es et partenaires du Parc national, qui permet d'assurer la saisie et la gestion des données via une interface web. Les données sont gérées dans une base de données PostgreSQL/Postgis.

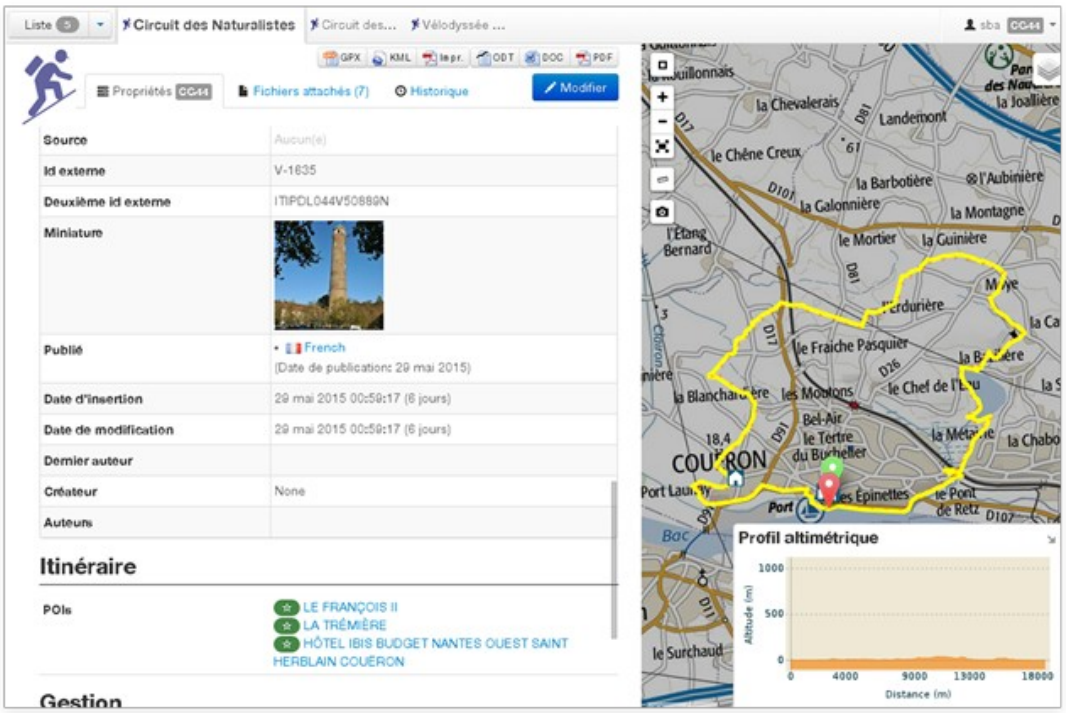


Figure 3: Interface web de Geotrek-admin

13 https://makina-corpus.com/
14 https://github.com/geotrekce
15 https://opensource.org/licenses/BSD-2-Clause

3.1.1 Segmentation dynamique

Geotrek repose sur le principe de la segmentation dynamique, ou système de référencement linéaire. Cette méthode de gestion de réseaux linéaires, à considérer comme un ensemble de lignes qui s'entrecroisent, permet de mettre en relation des tronçons dont la géométrie est renseignée et des objets qui se plaquent sur ceux-ci, sans avoir à stocker la géométrie de ces derniers.

Deux notions sont à la base de ce concept : les nœuds et les tronçons (ou segments). Un nœud est une intersection entre plusieurs tronçons. Chaque intersection est un nœud. Les deux extrémités d'un tronçon sont forcément deux nœuds.

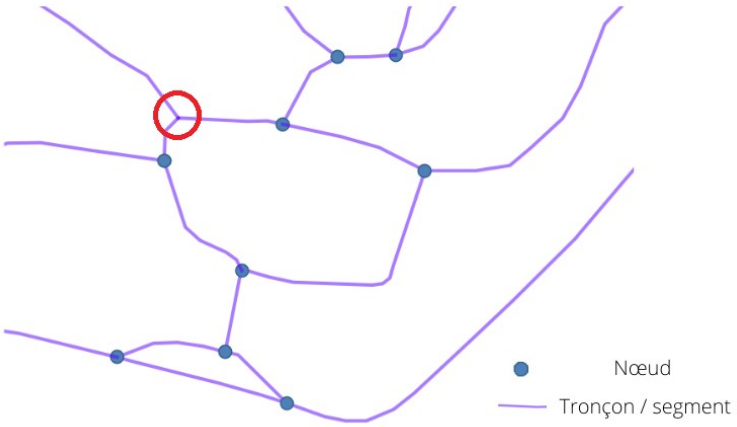
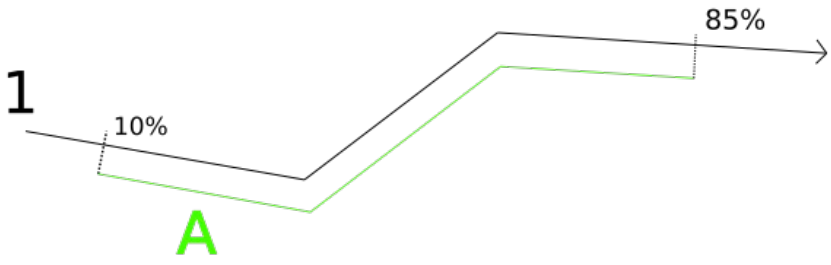


Figure 4: Réseau linéaire

On voit dans cet exemple (fig. 4) que le réseau est presque conforme aux principes de la segmentation dynamique. Toutes les intersections sont en effet des nœuds, à l'exception de celle entourée en rouge. Une fois cette intersection transformée en nœud, ce réseau linéaire sera cohérent topologiquement et prêt à être utilisé dans un contexte de segmentation dynamique.

Chaque objet que l'on souhaite ajouter à la base sera lié à un ou des tronçons par une table (en l'occurrence *core_pathaggregation*) qui recense à partir de, et jusqu'à quel point d'un tronçon l'objet se trouve (fig. 5). Ainsi, sa géométrie est calculée à partir de celle des tronçons, et mise à jour chaque fois que ceux-ci le sont.



id_objet	id_troncon	depart	arrivee
A	1	0,1	0,85

Figure 5: Principe de la segmentation dynamique (1)

Un des principaux avantages est qu'on peut associer autant d'objets que nécessaire sur un même tronçon sans avoir à stocker de géométrie supplémentaire, et tous ces objets auront une géométrie strictement similaire sur la partie de tronçon qu'ils ont en commun. Ainsi, si on veut représenter quatre objets sur le même tronçon 1, voici à quoi cela ressemble (fig. 6) :

id_objet	id_troncon	depart	arrivee
A	1	0,1	0,85
B	1	0,05	1
C	1	1	0,93
D	1	0	1

Figure 6: Principe de la segmentation dynamique (2)

Le point de départ peut être supérieur au point d'arrivée si l'objet emprunte le tronçon dans le sens inverse de celui-ci. Dans Geotrek, une colonne *order* est ajoutée, qui permet d'indiquer dans quel ordre l'objet emprunte les différents tronçons sur lesquels il est plaqué. C'est une information indispensable pour recréer une géométrie correcte, en fusionnant les géométries des différents tronçons dans le bon ordre.

Cette segmentation dynamique nécessite donc un réseau de tronçons parfaitement topologique. En effet, si les extrémités de deux tronçons supposés mitoyens sont géographiquement distantes de quelques centimètres, cela a des conséquences sur les géométries de l'ensemble des objets qui emprunteraient ces deux tronçons. Ainsi, les erreurs de topologie suivantes (fig. 7), très courantes dans les réseaux linéaires (y compris ceux étant théoriquement topologiques comme OpenStreetMap), doivent être corrigées pour que la segmentation dynamique fonctionne correctement :

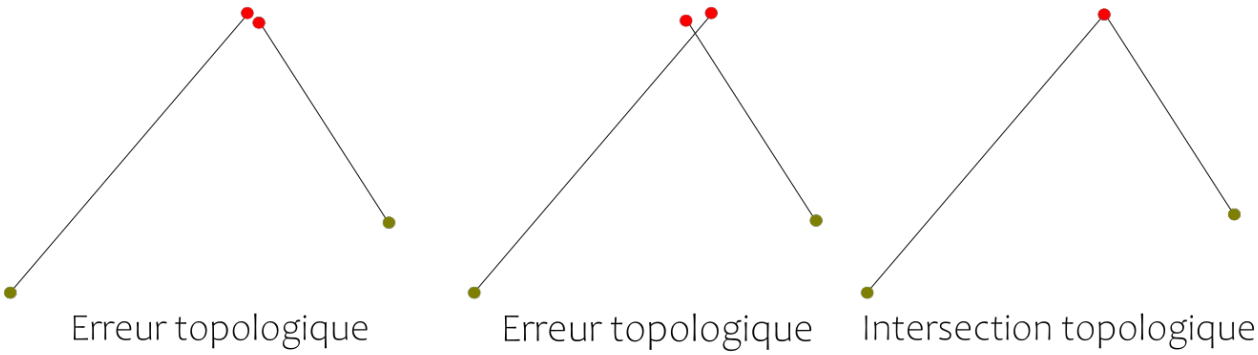


Figure 7: Erreurs topologiques et leur correction

3.1.2 Modèle de données

Le modèle relationnel de Geotrek est trop volumineux pour le présenter dans son ensemble ici, mais en voici une version très simplifiée :

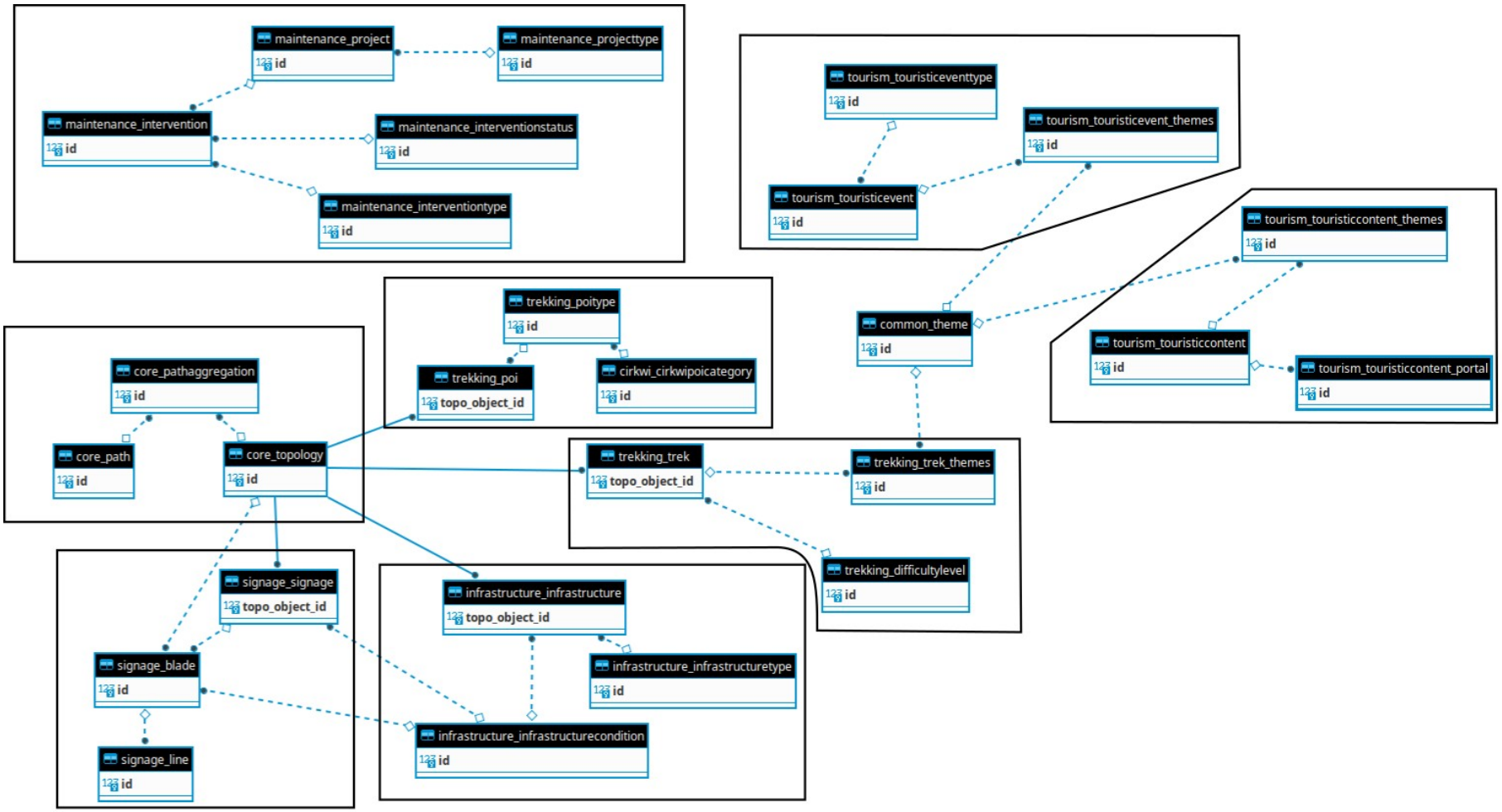


Figure 8: Modèle relationnel de la base de données Geotrek

Comme on le voit, la table *core_path* constitue la fondation de nombreuses tables et fonctionnalités de Geotrek. C'est le linéaire, constitué de tronçons auxquels de nombreux objets s'associent par la suite : les randonnées (*trekking_trek*), les points d'intérêt (*trekking_poi*) ou encore la signalétique (*signage_signage*).

3.2 Geotrek-rando

C'est le site web de valorisation touristique : il permet de montrer sur une carte et une liste tous les itinéraires de randonnée sélectionnés par le Parc national, mais aussi les hébergements, les prestataires touristiques, les commerçant·es… Un ensemble de filtres permet aux internautes de trouver plus facilement ce qui les intéresse. À chaque itinéraire de randonnée sont associés des points d'intérêt (POI), autres itinéraires ou hébergements géographiquement proches.

Figure 9: Interface web de Geotrek-rando

Les données de Geotrek-admin sont exportées via une simili API sous forme de fichiers GeoJSON, récupérés par Geotrek-rando afin de les afficher dans l'interface web.

3.3 Développement

Le développement de Geotrek est en théorie ouvert à tou·tes : le code source est disponible publiquement sur GitHub, les *issues* et les *pull requests* sont bienvenues. Dans les faits cependant, le logiciel est presque intégralement maintenu et amélioré par l'équipe de Makina Corpus. Deux fonctionnements cohabitent. Dans le cadre d'un maintien courant, Makina Corpus réalise les opérations de règlement de bugs, de mises à jour de sécurité, etc.

Par contre, pour ce qui est du développement de nouvelles fonctionnalités ou des mises à jour d'importance, ce sont les organisations utilisatrices du logiciel qui en sont à l'initiative via de simples prestations ou comme plus récemment via un groupement de commande. Cette dernière méthode a permis de réaliser une refonte et mise à jour technique en plus du développement de nouvelles fonctionnalités.

3.4 Geotrek au Parc national des Cévennes

Ma première mission en arrivant au Parc national a été de prendre en main Geotrek, ainsi que de dresser un état des lieux de son utilisation dans les services : qui s'en sert, pour quoi faire, quelles sont les fonctionnalités investies, quelles sont celles qui ne sont pas exploitées, etc. ?

Depuis son adoption en 2014, Geotrek est devenu la pierre angulaire de la stratégie numérique de valorisation touristique du Parc national. Le site Destination Cévennes est ainsi le seul endroit où le Parc national publie des informations touristiques de manière dynamique (ajout, suppression, mises à jour). Les pages dédiées au tourisme sur le site internet principal du parc (https://www.cevennes-parcnational.fr) comprennent quelques informations pérennes, par exemple le nombre de kilomètres de sentiers dans le parc ou les principaux itinéraires type GR, mais renvoient toutes *in fine* vers Destination Cévennes.

Le Parc national utilise Geotrek pour valoriser :

- les itinéraires de randonnée (à pied, à vélo, à cheval) ;
- les hébergements (maisons d'hôte, campings, gîtes...) ;
- les prestataires touristiques (canyoning, visite de grottes, musées…) ;
- les évènements organisés par le Parc (sorties naturalistes, animations pour les enfants…) ;
- les restaurants et fermes proposant de la vente directe.

Geotrek-admin permet de saisir les informations attributaires pour l'ensemble de ces données via des formulaires. Afin de ne pas trop alourdir l'application, seuls les tronçons utilisés par des itinéraires de randonnée sont activés dans Geotrek-admin. La majorité des tronçons présents dans *core_path* ne sont en fait utiles qu'au plan de circulation, et ne sont pas visibles dans l'application. La création de la géométrie d'un itinéraire de randonnée se fait dans Geotrek-admin par *snapping* sur les tronçons visibles.

En plus de cela, Geotrek-admin est utilisé pour gérer :

- le plan de circulation[16] : c'est ce qui détermine l'ouverture ou la fermeture des voies en cœur de parc aux différents types de circulation (motorisée, équestre, cycliste, pédestre). Stocké en-dehors de Geotrek, il est néanmoins directement dépendant du linéaire de celui-ci, et tout comme celui-ci est géré en segmentation dynamique ;
- une partie de la signalétique : les panneaux ayant été placés à l'occasion du passage du Tour de France en 2020 sont recensés dans les tables correspondantes.

Un certain nombre de fonctionnalités ne sont actuellement pas utilisées :

- la signalétique est majoritairement non renseignée, et pour l'instant stockée dans des fichiers de type Shapefile en-dehors de Geotrek ;
- le suivi des interventions de maintenance sur les équipements et sentiers n'est pas fait *via* Geotrek ;
- les équipements (barrières, tables de pique-nique, abris) sont peu renseignés sur Geotrek ;
- les données sur les activités de pleine nature sont stockées en-dehors de Geotrek, le « module Outdoor », un nouveau module de Geotrek-admin permettant de gérer des activités sportives non « linéaires » (escalade, alpinisme, canyoning) par sites de pratiques, n'a pas encore été adopté.

16 http://www.cevennes-parcnational.fr/fr/le-parc-national-des-cevennes/la-reglementation-du-coeur/la-circulation-et-le-stationnement-des

Le graphique suivant propose un résumé de l'organisation autour de Geotrek au sein du Parc national :

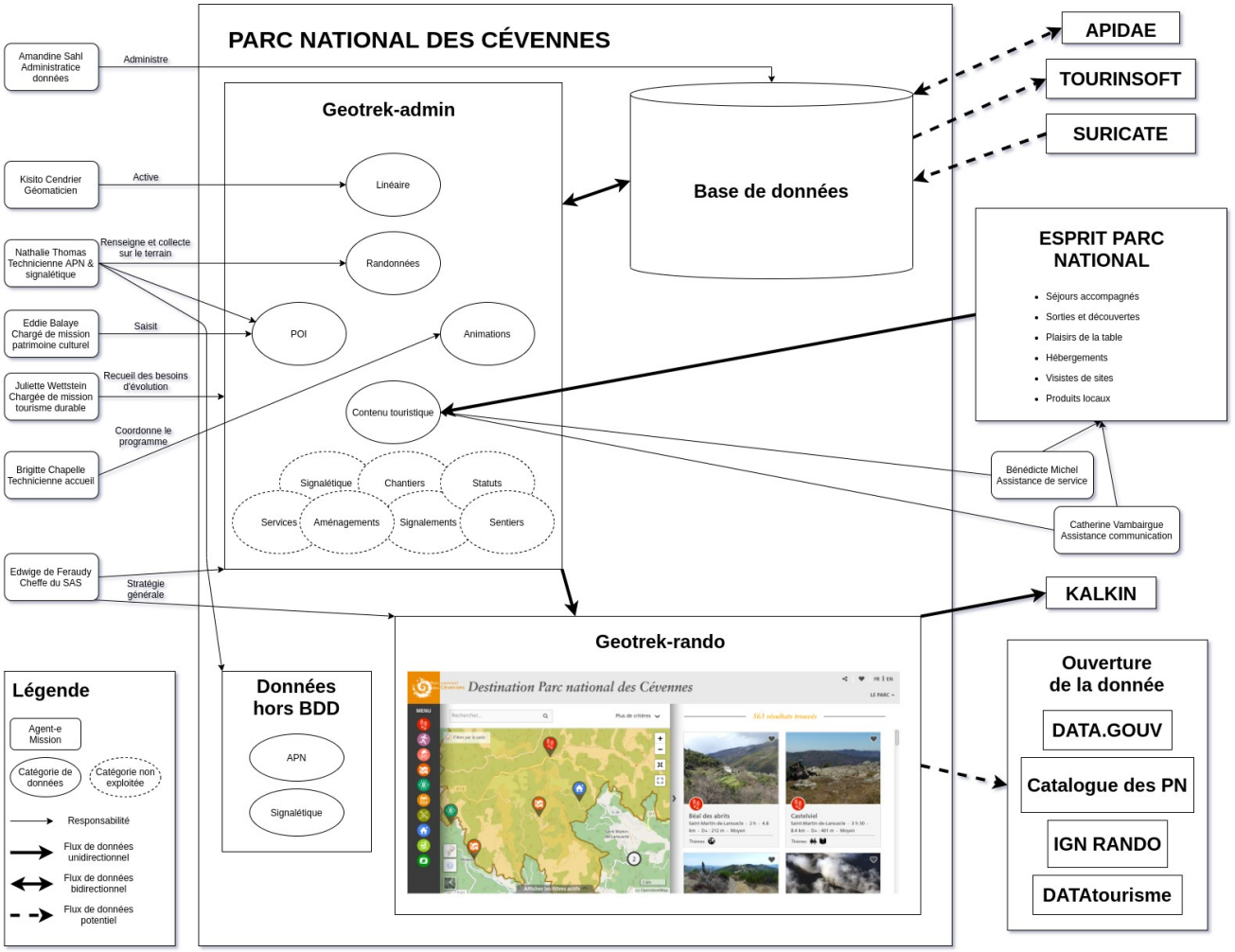


Figure 10: Utilisation de Geotrek au Parc national des Cévennes (I. Djepa Creutz)

Une version 3 de Geotrek-rando a été publiée il y a quelques mois[17], apportant notamment une modernisation importante de l'apparence de la plateforme. Le Parc national a décidé de ne pas migrer vers cette version juste avant la saison touristique, tant qu'elle ne serait pas assez éprouvée par des mois d'usage. Aussi, tout mon stage s'est fait en tenant compte de la version 2 de Geotrek-rando.

17 https://github.com/GeotrekCE/Geotrek-rando-v3

4. Donnée ouverte

4.1 État des lieux

Au-delà de la culture de travail des services informatiques des parcs nationaux, très favorable aux principes *open source*, comme le montre leurs dépôts GitHub[18], l'ouverture des données est une obligation légale pour les établissements publics.

En effet, la loi, dite Lemaire, n°2016-1321 du 7 octobre 2016 pour une République numérique[19] oblige les administrations à diffuser les données dont la publication présente un intérêt économique, social, sanitaire ou environnemental, ainsi que les bases de données produites.

Les parcs nationaux sont clairement en retard sur leurs obligations : seuls quatre d'entre eux, dont les Cévennes, possèdent un compte sur data.gouv.fr, et seuls des données relatives aux périmètres des zones cœur et d'adhésion y sont publiées[20].

4.2 Alcotra PITEM MITO

Dans le cadre du programme européen ALCOTRA[21](Alpes Latines COopération TRAnsfrontalière), un programme de coopération transfrontalière européen couvrant le territoire alpin entre la France et l'Italie (cinq départements français et quatre provinces italiennes), plusieurs PITEM (Plan Intégré ThEMatique) sont mis en œuvre. Les PITEM sont constitués de plusieurs projets simples organisés autour d'une thématique commune pour une durée de quatre ans. Le PITEM Modèles Intégrés pour le Tourisme Outdoor[22] (MITO) vise à favoriser un marché touristique mondial pour les sports de plein air en milieu naturel.

Le Parc national des Écrins est mandaté au sein de ce projet afin de réaliser un standard d'échange de données entre les différents partenaires français et italiens. Pour cela, il a réalisé une analyse du socle commun de données permettant de définir certaines activités extérieures, dont la randonnée. Fin 2020, ce standard a été validé par les partenaires du projet européen. Afin d'apporter une valeur ajoutée à ce projet, le Parc national des Écrins a souhaité travailler à partir de ce socle commun à la création d'un « schéma de données » concernant les itinéraires de randonnée.

18 https://github.com/PnX-Si
19 https://www.legifrance.gouv.fr/loda/id/JORFTEXT000033202746/
20 https://www.data.gouv.fr/fr/organizations/parc-national-des-cevennes/
21 https://www.interreg-alcotra.eu/fr
22 https://www.interreg-alcotra.eu/fr/decouvrir-alcotra/les-projets-finances/mito-modeles-integres-pour-le-tourisme-outdoor-dans-lespace

À terme, il est envisagé que ce schéma vienne enrichir la liste des schémas disponibles sur le site https://schema.data.gouv.fr/, ce qui permettrait de publier facilement des données standardisées et interopérables en donnée ouverte, notamment sur le site data.gouv.fr.

4.3 Groupe de travail

Un groupe de travail a été formé suite à ces réflexions, afin de discuter de la pertinence d'un schéma de données ainsi que de sa forme, et de l'implémenter. Ce groupe de travail est composé de multiples organisations intéressées par le projet, issues de différents secteurs :

- parcs nationaux et naturels régionaux (Écrins, Cévennes, Luberon) ;

- collectivités territoriales (régions PACA et *Liguria*, conseils départementaux des Hautes-Alpes, du Vaucluse, des Côtes-d'Armor) ;

- secteur touristique (ADN Tourisme, DATAtourisme, APIDAE Tourisme, Auvergne-Rhône-Alpes Tourisme) ;

- secteur du numérique (Etalab / DINUM, Apitux, April) ;

- secteur de l'information géographique (IGN).

Le début de mon stage a coïncidé avec la première réunion de ce groupe de travail, auquel j'ai donc été intégré dès le départ. Ce rendez-vous initial a permis d'échanger entre les membres du groupe, de préciser la vision et les attentes de chacun·e, ainsi que de valider le choix d'un schéma de données. La volonté collective de se doter d'un schéma semble forte car elle intervient dans un contexte d'augmentation des échanges de données autour des activités sportives et entre acteurs touristiques.

4.4 Intérêt du schéma de données

Etalab[23], le département de la direction interministérielle du numérique (DINUM) qui coordonne la stratégie de l'État dans le domaine de la donnée, a été associé au groupe de travail dès le départ. En effet, il administre une plateforme nationale de référencement des schémas produits par différents acteurs publics, et accompagne ceux-ci depuis la phase de discussion jusqu'à l'implémentation. Sur schema.data.gouv.fr sont référencés une vingtaine de schémas répondant à l'un des critères suivants :

«

- le schéma a été introduit par une disposition réglementaire ;
- la réutilisation des données décrites par le schéma bénéficie à un grand nombre ou de nombreux producteurs sont amenés à utiliser ce schéma.

»[24]

Concrètement, un schéma de données est un fichier qui décrit une structure de données, permet de valider des fichiers de données afin de vérifier leur conformité ou de créer des formulaires de saisie compatibles, etc. Tout ceci aide à la progression de la qualité des jeux de données produits, tout en assurant leur inter-compatibilité et en facilitant la réutilisation des données ouvertes. Un schéma est l'implémentation informatique d'un standard, c'est un outil technique. Un standard est, selon Wikipédia, « une exigence destinée à établir une compréhension commune de la signification ou de la sémantique des données, afin d'assurer un usage correct et approprié »[25].

Les schémas sont donc utilisés dans tous les domaines qui utilisent des standards : aussi bien dans le monde naturaliste (standards du Système d'Information Nature et Paysages, SINP[26]) que maritime (standards du Système d'Information sur le Milieu Marin[27]) et bien sûr informatique (spécifications de formats de fichier, comme GeoJSON[28]).

23 https://www.etalab.gouv.fr/
24 schema.data.gouv.fr
25 https://fr.wikipedia.org/wiki/Standards_et_normes_de_m%C3%A9tadonn%C3%A9es
26 http://standards-sinp.mnhn.fr/
27 https://sar.milieumarinfrance.fr/
28 https://datatracker.ietf.org/doc/html/rfc7946

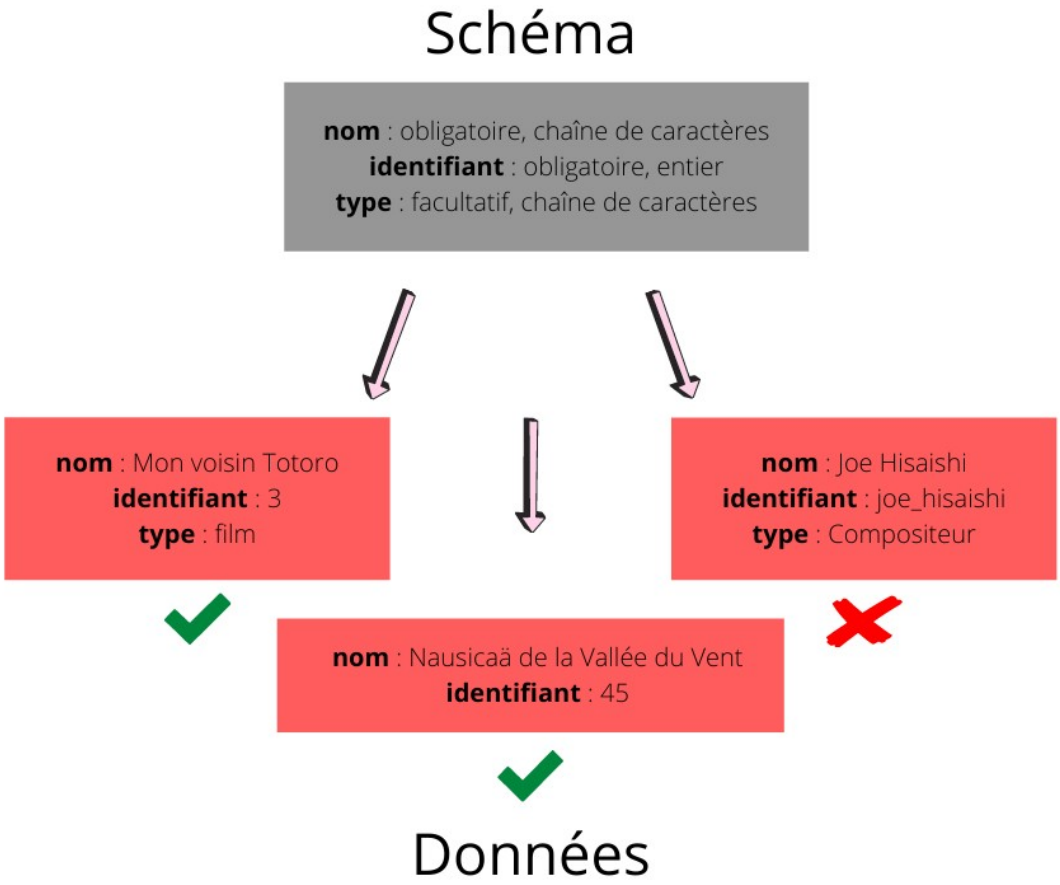


Figure 11: Validation de données grâce à un schéma (I. Djepa Creutz)

Pour mieux comprendre, voici une représentation...schématique de la validation de données par un schéma (fig. 11). Les films *Mon voisin Totoro* et *Nausicaä de la Vallée du Vent* respectent la structure et les contraintes imposées par le schéma, alors que l'identifiant de *Joe Hisaishi* ne respecte pas la contrainte de type « entier ». Les données ne sont donc pas valides.

4.5 Choix techniques et thématiques

Si le premier atelier a permis de définir l'orientation générale, le deuxième devait entrer dans le vif du sujet en déterminant le contenu du schéma : la liste des champs ainsi que leurs propriétés (caractère obligatoire ou non, type de données, etc). Camille Monchicourt, géomaticien du Parc national des Écrins et principal animateur de ce groupe de travail, avait préparé à cet effet une liste qu'il proposait en tant que base de travail.

Les discussions de ce deuxième atelier ont montré l'importance de la définition d'un schéma, notamment en mettant à jour les divergences d'interprétation autour du nom et du contenu des champs. Premièrement a été rappelée l'importance du terme « itinéraires de randonnées », et non seulement « randonnées ».

En effet, j'avais créé une première version du schéma de données au format Table Schema[29] en prévision de cet atelier, pour lequel j'avais omis le mot « itinéraires ». Pour les acteurs et actrices du tourisme, une randonnée faisait plutôt référence à un évènement daté, avec des participant·es, tandis qu'un itinéraire de randonnée correspondait bien à la scénarisation plus pérenne de la nature que nous cherchions à représenter : un point de départ, une suite de chemins, un point d'arrivée, un intérêt paysager, sportif, patrimonial particulier, etc. Corrections faites, le principal point d'achoppement fut en définitive les termes de « description », « ambiance » ou encore « instructions ».

En effet, les parcs des Écrins et des Cévennes travaillant principalement avec Geotrek[30], le vocabulaire utilisé dans ce logiciel (et donc standardisé parmi toutes ses instances) a infusé notre réflexion initiale. Ainsi, le champ *description_teaser* y représente une courte présentation, un court résumé de la randonnée et de ce qu'on peut espérer y trouver. Le champ *description* est lui consacré à des instructions de direction et d'orientation. Enfin, le champ *ambiance* est une description plus longue de la randonnée et de l'intérêt qu'elle présente. On voit qu'un élément de confusion est présent dès la conception : en se fiant seulement à leur nom, il serait logique de considérer que *description_teaser* est une version courte de *description*. Pourtant, ce n'est pas la manière dont sont utilisés ces champs. La disposition des contenus dans l'interface de Geotrek incite à considérer *description_teaser* comme une accroche et *ambiance* comme l'extension de cette accroche. Ces deux champs sont ainsi plutôt thématiques, alors que le champ *description* a un contenu plus technique, permettant de ne pas se perdre et de retrouver son chemin. On voit ci-dessous une capture d'écran de l'interface corroborant cette interprétation (fig. 12).

Figure 12: Description d'un itinéraire

29 https://specs.frictionlessdata.io/table-schema/
30 https://rando.ecrins-parcnational.fr/

Cette confusion induite par le nom des champs et l'interface se retrouve dans le contenu. On voit ci-dessous que si les champs de l'itinéraire « Le Gardonnet »[31] correspondent bien à la répartition expliquée ci-dessus, ceux de « La Tour du Canourgue »[32] semblent moins clairement distincts. Sa *description* aurait pu être une *ambiance*, les instructions de direction étant très évasives.

name	Le Gardonnet	La Tour du Canourgue
description_teaser	Dans les pas des anciens de Saint-Hilaire à la Canonge.	Le sentier chemine sur les traces des seigneurs de la tour du Canourgue, traversant les vestiges d'un ancien village fortifié dans lequel se mêlent histoire et architecture.
description	Depuis le parking sous la mairie, prendre la piste qui descend sous le cimetière. À environ 250m, après le second lacet, quitter la piste en direction de l'Elze. En arrivant sur la route, après quelques mètres vers la droite, emprunter la piste à gauche en direction de la passerelle du Gardonnet. Après 750m, quitter la piste et descendre vers le fond du vallon à travers la forêt pour arriver à la passerelle qui permet de franchir le Gardonnet. Remonter sur l'autre versant vers le hameau de la Teissonnière. Déboucher sur la route, la suivre vers la droite à travers le hameau du Foussat puis en passant sous le hameau de la Canonge. Au panneau Malacombe par le Gardonnet, prendre la piste à droite. Après quelques mètres, prendre à gauche le sentier pour redescendre vers le Pont de Malacombe. Traverser le Gardonnet sur le pont. Au panneau l'Elze, monter à droite à travers chênes verts et châtaigniers en direction de l'Elze. 1. À environ 800m, en arrivant sur la piste, deux itinéraires sont possibles : l'un plus direct longe	Après avoir passé un bois de chênes verts, le sentier permet de découvrir plusieurs panoramas qui s'ouvrent sur l'ensemble de la vallée Française. Il traverse ensuite les ruines d'un hameau et monte jusqu'au promontoire du donjon du Castrum de Canourgue (fin XIIe, début XIIIe). Après le fossé de défense, il chemine sur les traces des bâtiments disparus... À l'intérieur du donjon sont reproduits des plans et scènes imagés de la vie médiévale.

31 https://destination.cevennes-parcnational.fr/rando-a-pied/le-gardonnet/
32 https://destination.cevennes-parcnational.fr/sentiers-de-decouverte/la-tour-du-canourgue/

	la butte, l'autre passe par la crête. 2. Au hameau de l'Elze, prendre la route vers la droite. 3. Retrouver le sentier en direction du Temple de Saint-Hilaire et de votre point de départ.	
ambiance	Ce sentier relie deux parties de la commune de Saint-Hilaire-de-Lavit. Longtemps parcouru par les habitants, par les troupeaux et par le facteur pour aller d'un hameau à l'autre ou vers les moulins, il est resté caché pendant de nombreuses années. Il recèle une grande diversité de points de vue entre le fond des vallons et les passages de crête, montre les hameaux, l'usage des terres et de l'eau.	Les fouilles archéologiques entreprises à la tour du Canourgue ont permis de démontrer qu'il s'agissait du donjon d'un château vraisemblablement abandonné à la fin du XIVe siècle ou au début du XVe. Il s'y adossait, hors fortification, tout un ensemble d'habitat dont les ruines sont actuellement effacées par la végétation. Les castra médiévaux avaient une fonction de prestige et un rôle défensif : le castrum du Canourgue a-t-il dû se défendre un jour ? Des érudits évoquent depuis deux siècles la présence d'un réseau de "tours à signaux" destiné à assurer la sécurité des habitants. Le Canourgue a-t-il fonctionné ainsi ?

Si les choses ne sont déjà pas claires au sein d'une même organisation (le Parc national des Cévennes) et d'un même logiciel (Geotrek), il est évident que la confusion ne peut que grandir avec le partage et l'échange de données, lorsque se multiplient les points de vue et les réutilisations de celles-ci. L'intérêt d'un standard de données est parfaitement démontré par cet exemple, qui a occupé une bonne part des discussions lors du deuxième atelier. Pour un participant, le terme « description » renvoyait forcément à des instructions de direction, et serait déjà en quelque sorte « standard » dans le monde de la randonnée. Le terme « description_courte », qui avait été choisi à partir de « description_teaser » était donc pour lui inapproprié. Pourtant le champ *description* sur le site IGN Rando[33] correspond bien à une présentation générale de la randonnée, l'équivalent du champ *ambiance* de Geotrek… Quant au site de la FFRP[34], il utilise… « descriptif » pour présenter les instructions de direction ! En l'absence d'ontologie faisant référence, nous avons dû faire des choix, dans l'idée que ce schéma puisse être accepté, et donc utilisé, par le plus grand nombre.

33 https://ignrando.fr/fr/parcours/26607-les-aiguilles-de-chabrieres
34 https://www.mongr.fr/trouver-prochaine-randonnee/itineraire/gr-70-le-chemin-de-stevenson

En l'occurrence, la discussion sur ces trois champs n'a pas été conclue de manière très claire, mais afin de pouvoir avancer sur le schéma j'ai retenu les trois termes qui semblaient convenir au plus de membres du groupe de travail : « instructions », « presentation », « presentation_courte ».

Il existe plusieurs formats de schémas, dont deux sont privilégiés par Etalab : Table Schema et JSON Schema[35]. Pour mieux comprendre les différences entre ces deux formats, voici un tableau comparatif :

Table Schema	JSON Schema
Adapté aux données tabulaires type csv	Adapté aux données hiérarchisées type JSON
Le format csv est aisément manipulable sans connaissances informatiques avancées (logiciels de bureautique)	Le format JSON est considéré comme un standard du web pour l'échange de données
Frictionless, l'organisation maintenant le Table Schema, propose également un validateur « officiel »	Les implémentations du schéma sont indépendantes de l'organisation qui maintient celui-ci, et toutes potentiellement différentes
Permet de stocker un objet GeoJSON dans un champ grâce au type « geojson »	Permet d'utiliser des données au format (Geo)JSON, très populaire pour le partage de données spatiales

J'avais d'abord fait le choix du Table Schema, et publié une première version sur GitHub (https://github.com/PnX-SI/schema_randonnee/tree/v0.2.0), mais nous avons décidé lors du deuxième atelier de privilégier le format JSON Schema pour sa compatibilité plus grande avec le format GeoJSON.

4.6 JSON Schema

Un JSON Schema est un fichier JSON qui décrit la structure d'un autre fichier JSON. Pour cela plusieurs mots-clefs sont nécessaires :

- $schema : doit correspondre à l'identifiant de la version du JSON Schema auquel ce schéma se conforme ;
- $id : permet d'identifier le schéma, doit être unique au sein d'un système où cohabitent plusieurs schémas ;
- title et description : explicitent le but du schéma et ce qu'il décrit ;
- type : premier mot-clef contraignant, sa valeur doit correspondre au contenu du fichier que l'on souhaite valider. Est-ce un entier, un objet JSON, une chaîne de caractères..?

35 https://json-schema.org/

Enfin, si la valeur de ce premier mot-clef « type » est « objet », nous pouvons commencer à décrire les propriétés de celui-ci :

```
"uuid": {
  "type": ["string", "null"],
  "format": "uuid",
  "title": "Identifiant unique de type UUID",
  "description": "Identifiant unique généré de préférence par la BDD source",
  "examples": [
    "123e4567-e89b-12d3-a456-426614174000"
  ]
},
"url": {
  "type": ["string", "null"],
  "format": "uri",
  "title": "URL de la fiche source de l'itinéraire",
  "examples": [
    "https://destination.cevennes-parcnational.fr/sentiers-de-decouverte/mas-cevenol-de-la-roquette/"
  ]
},
"id_osm": {
  "type": ["integer", "null"],
  "title": "Identifiant de la relation OSM correspondante",
  "examples": [
    1913426
  ]
},
```

Figure 13: Définition des propriétés d'un itinéraire

Chaque propriété doit être décrite par plusieurs mots-clefs (fig. 13), qu'ils soient contraignants pour les données ou bien seulement informatifs pour les humain·es qui liront le schéma. J'ai utilisé sept mots-clefs différents pour décrire les propriétés :

- *type* : définit le type de la propriété (entier, chaîne de caractères, etc.) ;
- *title* : titre de la propriété ;
- *description* : utile lorsque le titre ne suffit pas ;
- *examples* : permet d'intégrer des exemples au schéma ;
- *enum* : liste des valeurs autorisées pour la propriété ;
- *format* : complète le *type* (date, uuid...)
- *pattern* : expression régulière à laquelle la valeur du champ doit se soumettre.

Avec ceci, nous pouvons décrire une structure de données simple, presque tabulaire, mais en JSON. Cependant, le but était de valider des données GeoJSON, dont la structure est plus complexe.

Il faut en effet considérer qu'un fichier à valider sera en fait une *Feature Collection*[36], elle-même composée de plusieurs *Features*[37] (plusieurs itinéraires de randonnée). Chaque *Feature* a sa propre géométrie qui, en GeoJSON, n'est pas au même niveau de l'arborescence que les autres propriétés. Enfin, il fallait nous assurer que le schéma respecte bien la spécification GeoJSON...elle-même décrite par un JSON Schema[38].

C'est pourquoi j'ai utilisé les *definitions*, un mot-clef proposé par JSON Schema qui permet de définir des propriétés et de les réutiliser à d'autres endroits sans avoir à dupliquer le code[39], ce qui est toujours une pratique conseillée en programmation. Il suffit ensuite de renvoyer vers ces *definitions* avec le mot-clef *$ref*[40] aux endroits souhaités (fig. 14). *$ref* autorise également les références à d'autres JSON Schema, ce qui permet de diviser son schéma en plusieurs fichiers et ainsi d'améliorer la lisibilité et faciliter la maintenance du schéma.

Ces possibilités m'ont été particulièrement utiles pour intégrer la spécification GeoJSON car elle a permis de l'extraire en partie du fichier schéma principal. Deux fichiers JSON Schema issus de la spécification officielle sont donc stockés localement aux côtés du schéma principal et décrivent respectivement une *LineString* et une *MultiLineString*.

Figure 14: Références aux définitions GeoJSON

36 https://datatracker.ietf.org/doc/html/rfc7946#section-3.3
37 https://datatracker.ietf.org/doc/html/rfc7946#section-3.2
38 https://github.com/geojson/schema
39 https://json-schema.org/understanding-json-schema/structuring.html#defs
40 https://json-schema.org/understanding-json-schema/structuring.html#ref

La description d'une *Feature Collection* est elle aussi directement issue de la spécification GeoJSON, mais étant donné que les propriétés de chaque *Feature* sont spécifiques au schéma, je n'ai eu d'autre choix que de copier la structure. Une référence externe n'était pas possible. Une *Feature Collection* a forcément des *features*, sous forme d'*array*. Les *items* de cette *array* sont des objets JSON qui ont forcément (*required*) comme *properties* un *type*, des *properties* et une *geometry*. La géométrie est une *LineString* ou une *MultiLineString* (via une référence aux définitions plus haut). Les propriétés sont également définies dans les définitions en début de schéma. J'ai fait ce choix car il permet de rendre plus claire la structure de la *Feature Collection* sans la noyer avec des centaines de lignes correspondant uniquement aux propriétés. On a ainsi une vue d'ensemble qui permet de modifier la structure aisément. Les propriétés de chaque *Feature* sont donc sous le mot-clef *definitions* au début du schéma (fig. 15).

```
"definitions": {
  "properties_randonnee": {
    "$id": "properties_randonnee",
    "type": "object",
    "required": [
      "id_local",
      "proprietaire",
      "nom_itineraire",
      "pratique",
      "depart",
      "arrivee",
      "instructions"
    ],
    "properties": {
      "id_local": {
        "type": "string",
        "title": "Identifiant de l'objet dans sa BDD source",
        "examples": [
          "37037"
        ]
      },
      "proprietaire": {
        "type": "string",
        "title": "Structure(s) productrice(s) de l'itinéraire",
        "examples": [
          "Parc national des Cévennes",
```

Figure 15: Définition de properties_randonnee

Seules sept propriétés ont été définies par le groupe de travail comme obligatoires : *id_local, proprietaire, nom_itineraire, pratique, depart, arrivee, instructions*. La géométrie est elle aussi obligatoire, mais gérée à l'échelon supérieur de l'arborescence JSON. Ces sept propriétés constituent le minimum requis pour qu'un itinéraire soit considéré comme valide.

Les autres propriétés sont facultatives, mais doivent tout de même se plier à des contraintes si elles existent. Enfin, aucune autre propriété que celles énumérées dans le schéma ne peut exister dans les données sans les invalider.

Afin de favoriser l'usage de ce schéma par le maximum d'organisations possibles, nous avons souhaité limiter sa complexité technique. Aussi, nous avons décidé d'aplatir l'arborescence JSON au maximum, par exemple en séparant un objet « parking » avec deux propriétés « infos » et « géométrie » en deux propriétés « parking_info » et « parking_geometrie », cette dernière au format WKT[41]. De même, les listes des communes traversées par l'itinéraire et leurs codes postaux auraient pu être stockées ensemble sous forme d'une seule liste « communes » constituée d'objets JSON aux propriétés « nom » et « code_insee ». Nous avons fait le choix de deux champs « communes_nom » et « communes_code » sous forme de chaîne de caractères, même pas de liste, pour faciliter la prise en main du schéma. Seuls les médias ont échappé à cet aplatissement, car il n'était pas possible de les représenter correctement autrement.

4.7 Validateur et GitHub Action

Contrairement au Table Schema pour lequel un validateur est tout indiqué (celui de *Frictionless*), il a fallu faire un choix parmi de nombreuses implémentations du JSON Schema. Cela a aussi posé la question de la version de JSON Schema à laquelle nous voulions nous conformer. J'avais initialement créé le schéma selon la dernière version disponible, la 2020-12. La spécification GeoJSON est elle rédigée selon la version draft-07. Cette différence n'étant pas prise en charge par certains validateurs, j'ai finalement rétrogradé le schéma en version draft-07 pour des raisons de compatibilité. C'est d'ailleurs la version recommandée par Ajv, un des validateurs les plus populaires, à moins d'avoir besoin de certaines fonctionnalités plus récentes, ce qui n'est pas notre cas.

Ajv est le validateur que nous avons choisi pour sa simplicité et le fait qu'il utilise JavaScript, un langage que je maîtrise. J'ai donc créé un script de validation de données qui :

- charge des sous-schémas (en l'occurrence les schémas GeoJSON) ;
- les ajoute à un schéma principal ;
- compile l'ensemble ;
- teste la validité d'un fichier JSON par rapport au schéma compilé ;
- affiche le résultat du test, avec la liste des non-conformités le cas échéant.

Ce script utilise Node.js[42] et est destiné à une utilisation locale.

41 https://docs.opengeospatial.org/is/18-010r7/18-010r7.html
42 https://nodejs.org/fr/

Pour améliorer le contrôle qualité en cas de mise à jour du schéma ou du fichier exemple présent sur le dépôt, j'ai également créé un GitHub *workflow* et une GitHub Action afin que, en plus des tests normalement effectués de manière locale, un *push* des fichiers *schema.json* ou *exemple-valide.json* sur le dépôt exécute ce même script Ajv et affiche les résultats de la validation. Les actions déjà existantes ne permettaient pas de gérer des sous-schémas, c'est pourquoi j'ai dû en créer une nouvelle. La seule différence fondamentale avec le script local est la façon dont les paramètres comme le chemin des schémas et des données à valider sont renseignés. Alors que le script de validation locale prend ces paramètres « en dur », cette GitHub Action se réfère à des variables définies dans le *workflow* utilisé pour l'appeler. Ainsi, toute référence au schéma des itinéraires de randonnée ou à GeoJSON est absente de l'action, et elle peut théoriquement être réutilisée par n'importe qui pour d'autres usages.

4.8 Export et publication des données Geotrek

Le but de ce schéma étant de standardiser la publication de jeux de données d'itinéraires de randonnées, une fois la version 1.0.0 publiée (https://github.com/PnX-SI/schema_randonnee/tree/v1.0.0) nous avons mis en place un export automatique des données directement depuis Geotrek vers la plateforme data.gouv.fr, conformément au schéma évidemment.

Pour cette tâche, je me suis fondé sur le travail de Cendrine Hoarau et Camille Monchicourt, du Parc national des Écrins. Cendrine Hoarau, étudiante en M1 Géographies Numériques et stagiaire dans les Écrins, avait fait plusieurs essais d'export depuis Geotrek, et d'automatisation de la publication des données. Son travail, qu'elle a présenté sur le blog du Parc national des Écrins[43], m'a beaucoup aidé et nous avons longuement discuté de la meilleure manière pour réaliser cet export. Le résultat de ce travail commun est une vue SQL (fig. 16), adaptée au modèle de données de Geotrek, appelée par un script ogr2ogr[44] qui convertit directement le résultat de la vue depuis le SQL en format GeoJSON.

Au Parc national des Cévennes, une vue parfaitement adaptée à nos données est interrogée par une tâche *cron* quotidienne[45], qui met à jour le fichier GeoJSON sur un de nos serveurs (fig. 17). Cet emplacement est accessible publiquement par une url[46], utilisée ensuite sur le site data.gouv.fr en tant qu'url source de notre jeu de données[47].

43 https://si.ecrins-parcnational.com/blog/2021-06-publier-opendata-continu.html
44 https://gdal.org/programs/ogr2ogr.html
45 https://doc.ubuntu-fr.org/cron
46 https://geotrek-admin.cevennes-parcnational.net/media/upload/randos_pnc.geojson
47 https://www.data.gouv.fr/fr/datasets/randonnees-du-parc-national-des-cevennes/

```sql
SELECT
    t.topo_object_id::varchar(250) AS id_local,
    sources.noms_source AS proprietaire,
    (SELECT contact FROM constants LIMIT 1) AS contact, -- adresse mail à renseigner dans les constantes
    NULL AS uuid, -- pas d'uuid prévu dans Geotrek
    -- construction de l'url valable pour Geotrek-rando V2
    (SELECT url_rando FROM constants LIMIT 1) || lower(unaccent(replace(tp.practice_name, ' ', '-'))) || '/'
    || lower(unaccent(replace(btrim(regexp_replace(t."name", '[^\w -]', '', 'g')), ' ', '-'))) || '/' AS url,
    -- pour Geotrek-rando V3, essayer quelque chose comme : 'urlportail/trek/' || 't.topo_object_id' || '-' || unacc
    NULL AS id_osm,
    t."name" AS nom_itineraire,
    tp.practice_name AS pratique, -- uniquement valable si vos noms de pratiques correspondent déjà au schéma, sinon
    lower(route.route::text) AS type_itineraire,
    c.liste_noms::text AS communes_nom,
    c.liste_codes::text AS communes_code,
    btrim(t.departure) AS depart,
    btrim(t.arrival) AS arrivee,
    round(t.duration::numeric,1)::real AS duree,
    balisage.b::text AS balisage,
    top.length::integer AS longueur,
    difficulty.difficulty AS difficulte,
    top.max_elevation AS altitude_max,
    top.min_elevation AS altitude_min,
    top.ascent AS denivele_positif,
    top.descent AS denivele_negatif,
    t.description AS instructions,
    t.ambiance AS presentation,
    t.description_teaser AS presentation_courte,
    themes.liste::text AS themes,
    t.advice AS recommandations,
    handi.liste::text AS accessibilite,
    t.access AS acces_routier,
    t.public_transport AS transports,
    advised_parking AS parking_info,
    ST_AsText(st_snaptogrid(st_transform(parking_location, 4326), 0.000027::double precision)) AS parking_geometrie,
    date(top.date_insert)::text AS date_creation,
    date(top.date_update)::text AS date_modification,
    medias.liste AS medias,
    parent.parent_id::varchar AS itineraire_parent,
    sol.liste::text AS type_sol,
    NULL::boolean AS pdipr_inscription,
    NULL::text AS pdipr_date_inscription,
    -- réduction de la précision des coordonnées à 5 décimales, simplification de la géométrie pour réduire le nombr
    st_simplifypreservetopology(st_snaptogrid(st_transform(top.geom, 4326), 0.000027::double precision), 0.000027::d
FROM selected_t t
```

Figure 16: Vue d'export des données de Geotrek conforme au schéma

```bash
# Export en GeoJSON depuis la vue v_treks_schema avec ogr2ogr

DB_HOST=db_host
PORT=5432
DB=db_name
USER=db_user
PASSWORD=db_path
EXPORT_PATH=/tmp

ogr2ogr -f "GeoJSON" "${EXPORT_PATH}/itineraires_rando.geojson"
\ PG:"host=${DB_HOST} port=${PORT} dbname=${DB} user=${USER} password=${PASSWORD}"
\ -nln randos_pnc
\ -sql "select * from public.v_treks_schema" cp $EXPORT_PATH/itineraires_rando.geojson
\ /opt/geotrek-admin/var/media/upload/randos_pnc.geojson
```

Figure 17: Script Unix shell d'export des données Geotrek

Les itinéraires de randonnée du Parc national sont ainsi publiés quotidiennement sur la plateforme référence de l'ouverture des données publiques. En se rendant sur le dépôt du schéma, les utilisatrices/teurs peuvent aisément comprendre la structure des données et travailler plus efficacement avec elles.

5. Geotrek-admin-aggregator

5.1 Le besoin

Le conseil départemental de la Lozère a récemment décidé, en s'inspirant du Parc national des Cévennes et du Parc naturel régional des Grands Causses[48] voisin, de la mise en place d'un Geotrek sur son territoire. La Lozère souhaite que les communautés de commune utilisent ce logiciel pour la gestion de leurs réseaux de sentiers et la publication de leurs itinéraires de randonnée.

Jusqu'ici deux Geotrek cohabitaient dans le département : celui du Parc national sur tout le Sud-Lozère, et celui du Parc naturel régional de l'Aubrac sur le plateau de l'Aubrac, au Nord-Ouest du département (fig. 18). Le Parc naturel de l'Aubrac utilise en fait le Geotrek du Parc naturel des Grands Causses : les deux partagent une frontière au bord du Lot et collaborent étroitement.

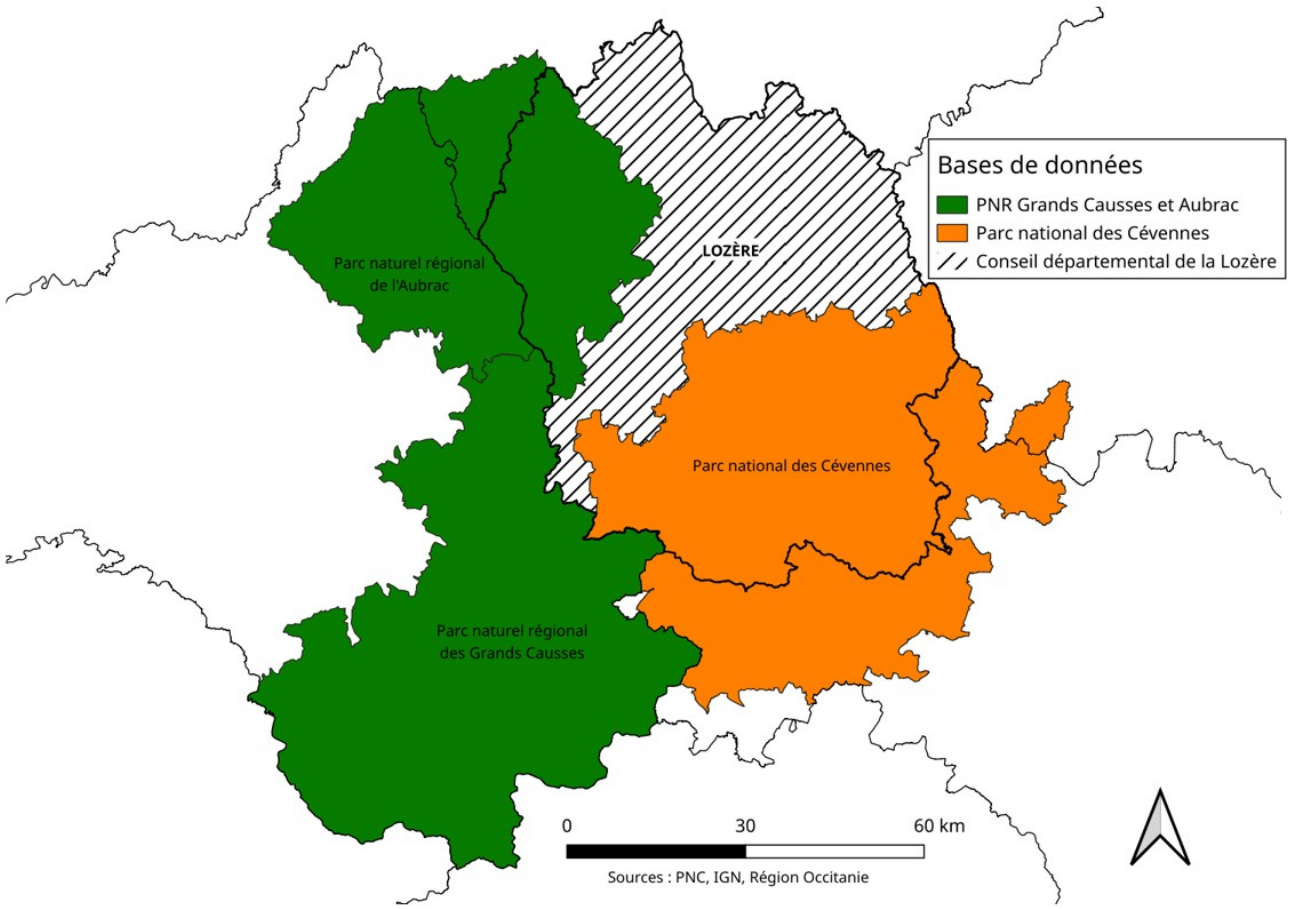


Figure 18: Carte de situation des parcs par rapport à la Lozère

48 https://rando.parc-grands-causses.fr/

Quelle que soit la zone du département concernée, ce sont presque toujours les communautés de communes qui ont la charge de renseigner leurs itinéraires de randonnée sur l'un ou l'autre Geotrek. En effet, si les départements ont comme compétence obligatoire la constitution d'un Plan Départemental des Espaces, Sites et Itinéraires[49] (PDESI), les communautés de communes se voient très souvent confier la compétence facultative de gestion de celui-ci à leur échelle, via les Réseaux Locaux d'Espaces, Site et Itinéraires (RLESI).

Les parcs et le conseil départemental sont vite tombés d'accord sur le fait qu'il fallait décider d'une répartition claire des communautés de communes entre les trois Geotrek, pour qu'aucune ne se retrouve à effectuer de double saisie. En effet, c'est pour l'instant la norme pour les communes dans l'aire d'adhésion du Parc national : en plus de renseigner sur notre Geotrek, elles doivent transmettre les mêmes informations au site internet du conseil départemental. C'est à la fois une charge de travail supplémentaire qui pourrait être évitée, et une mauvaise pratique en matière de qualité de la donnée. Cela multiplie les possibilités d'erreur, et fait coexister deux données informatiquement différentes, mais qui représentent la même chose. Chaque mise à jour des données pose exactement les mêmes problèmes en matière de charge de travail et de mauvaise pratique.

La répartition s'est donc faite assez naturellement : les communautés de commune dont le territoire est majoritairement compris dans un des deux parcs sont rattachées à celui-ci, les deux communautés de communes extérieures aux parcs sont rattachées au Geotrek du conseil départemental et les deux communautés de communes dont le territoire est très partagé sont rattachées au Geotrek départemental, ce qui permet d'équilibrer le nombre de communautés de communes sur chaque Geotrek. Sur la carte ci-dessous (fig. 19), la communauté de communes Millau Grand Causses fait partie de l'Aveyron, elle n'est présente sur la carte que pour rappeler qu'elle renseigne des données sur le Geotrek commun des Parcs naturels de l'Aubrac et des Grands Causses.

49 https://www.sportsdenature.gouv.fr/publications/outils-mobilisables-pour-perenniser-acces-lieux-de-pratique/pdesi

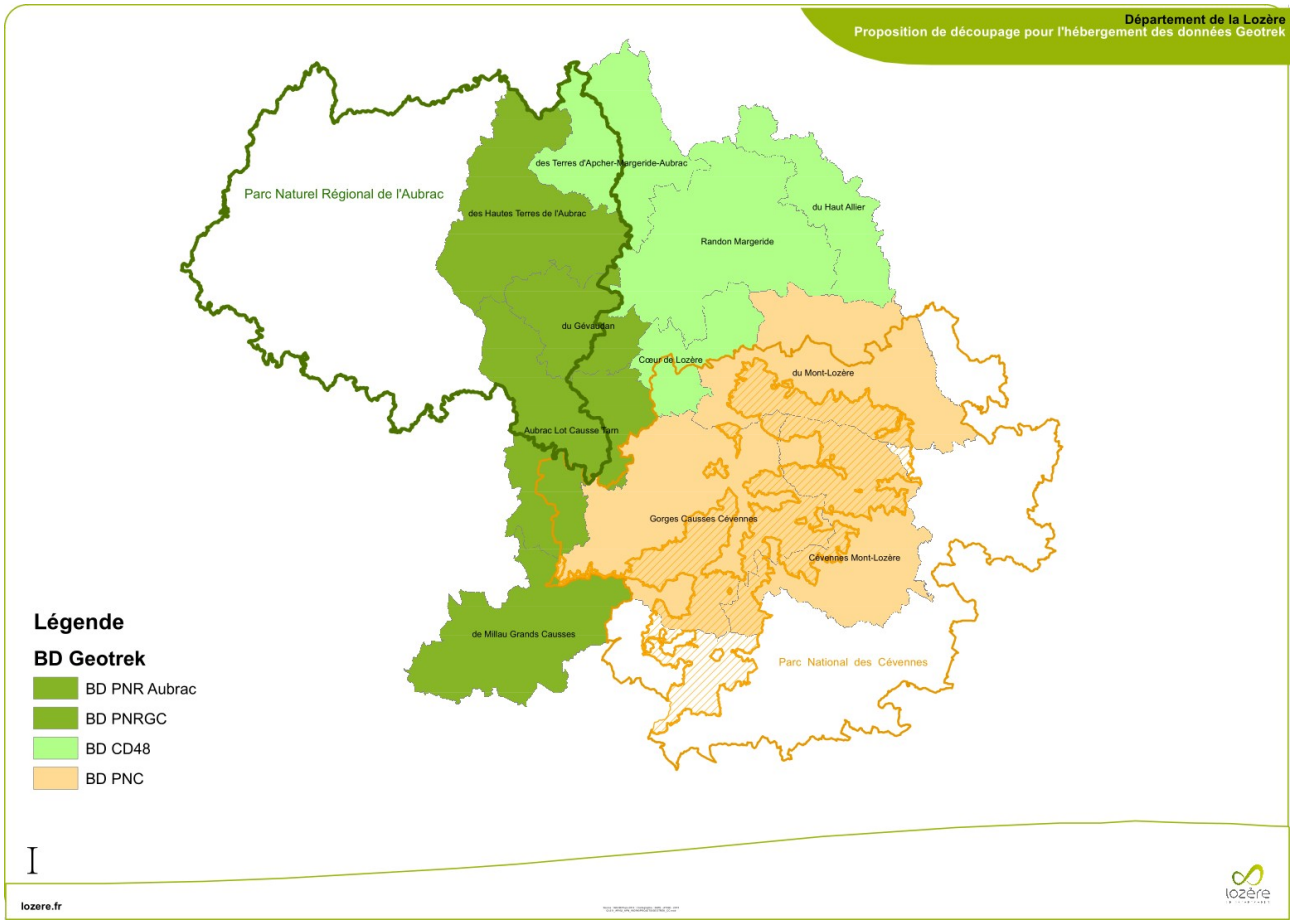


Figure 19: Carte de couverture des trois Geotrek lozériens – CD Lozère

Cette répartition pose d'autres questions.

Si la communauté de communes des Terres d'Apcher-Margeride-Aubrac renseigne ses randonnées exclusivement sur le Geotrek départemental, comment est-ce que le Parc naturel de l'Aubrac peut-il les valoriser sur sa plateforme Rando Aubrac[50] ? La situation de la communauté de communes Cœur de Lozère par rapport au Parc national est identique.

Le conseil départemental a quant à lui pour ambition de valoriser l'ensemble des randonnées de Lozère. Il faut donc que, d'un côté comme de l'autre, les données renseignées par les communautés de communes soient mutualisées, afin que les trois organisations puissent valoriser les données qui les intéressent.

De l'autre côté du Parc national, la situation est moins complexe mais soulève les mêmes problèmes. Le conseil départemental du Gard a également développé une instance de Geotrek, pour l'instant surtout utilisée pour ses fonctionnalités de gestion plutôt que de valorisation touristique. Leurs données de gestion sont bien plus complètes que celles du Parc national, et il serait bénéfique pour celui-ci d'y avoir accès.

50 https://www.rando-aubrac.fr/

Le territoire du Parc national n'est pas le seul endroit où plusieurs structures se retrouvent confrontées aux mêmes questions. Tant que Geotrek n'était utilisé que par des parcs, aucun chevauchement de territoires n'était possible, mais avec l'extension de son utilisation à de plus en plus de collectivités territoriales, cette situation se répète à de nombreux endroits. C'est ainsi qu'un groupe de travail a été créé en juillet pour évaluer les besoins, dans l'idée d'aller vers le financement d'une commande de prestation de développement à Makina Corpus.

5.2 Geotrek(-rando) aggregator

En 2019 déjà, le conseil départemental des Hautes-Alpes avait financé une fonctionnalité similaire : Geotrek-aggregator[51]. Cet outil permet d'agréger les données provenant de plusieurs Geotrek-admin dans une seule plateforme Geotrek-rando.

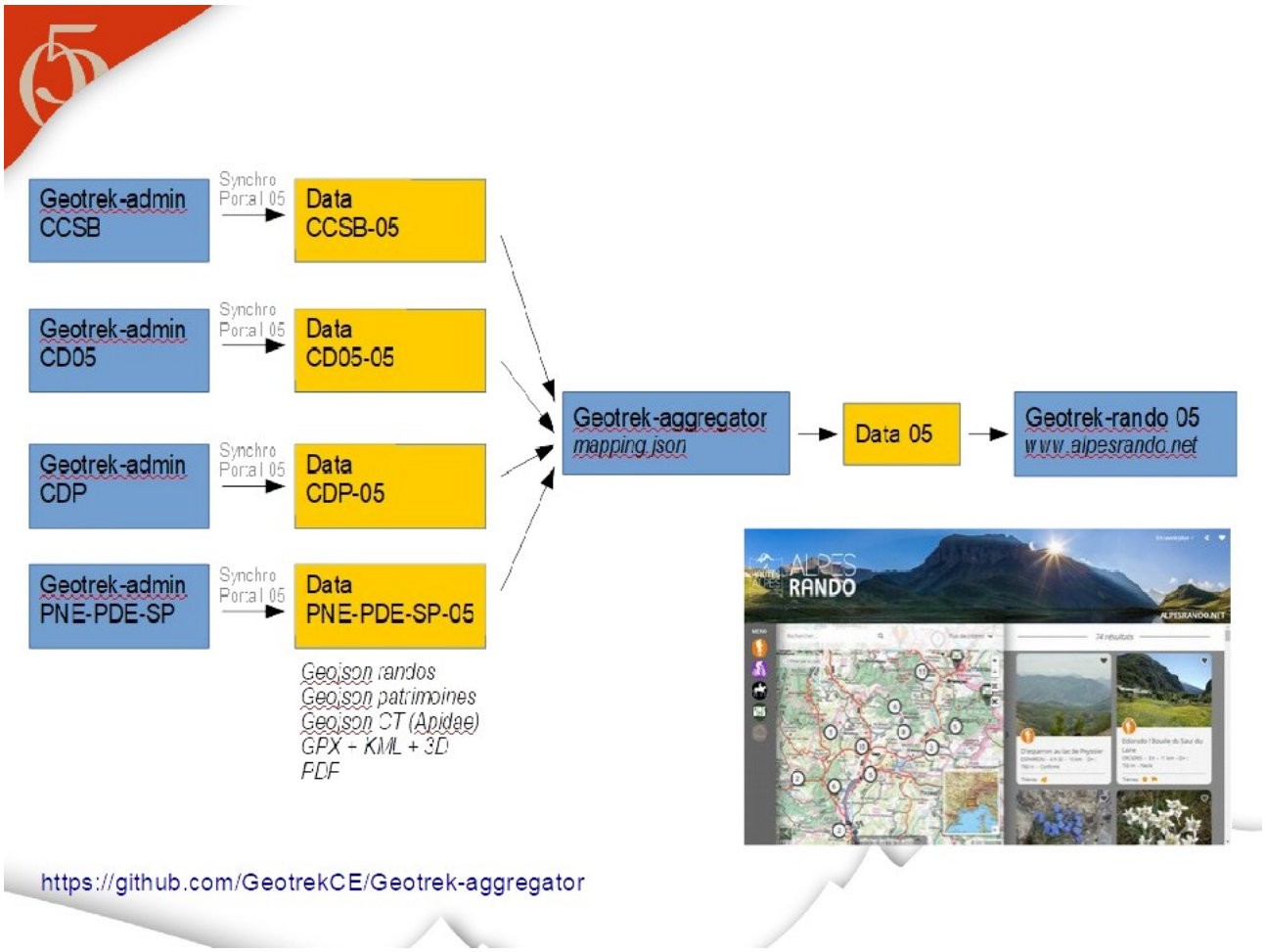


Figure 20: Fonctionnement du Geotrek-rando aggregator - CD Hautes-Alpes

51 https://github.com/GeotrekCE/Geotrek-aggregator

Le schéma ci-dessus (fig. 20) explique le fonctionnement de l'agrégateur :

- un attribut *targetportal* dont la valeur est « Portail 05 » permet d'exporter les données voulues sous forme de fichiers GeoJSON ;
- le Geotrek-aggregator prend en charge le *mapping*, c'est-à-dire la mise en correspondance des ontologies, des fichiers provenant des quatre Geotrek-admin concernés, et produit un fichier unique avec les données qui en résultent ;
- le Geotrek-rando utilise ce fichier pour afficher les données sur l'application.

Cette solution fonctionne donc uniquement pour de l'affichage de données : en effet, l'export au format GeoJSON fait perdre tout un ensemble d'informations par rapport à la base de données initiale. De plus, le seul but étant l'affichage à destination du public, seules les données à vocation touristique sont conservées. Enfin, ce Geotrek-aggregator est incompatible avec la version 3 de Geotrek-rando, ce qui le rend très prochainement obsolète. Le conseil départemental des Hautes-Alpes par exemple, un des principaux utilisateurs de cet agrégateur[52], est obligé de conserver la version 2 de Geotrek-rando, alors que son partenaire le Parc national des Écrins a changé de version[53].

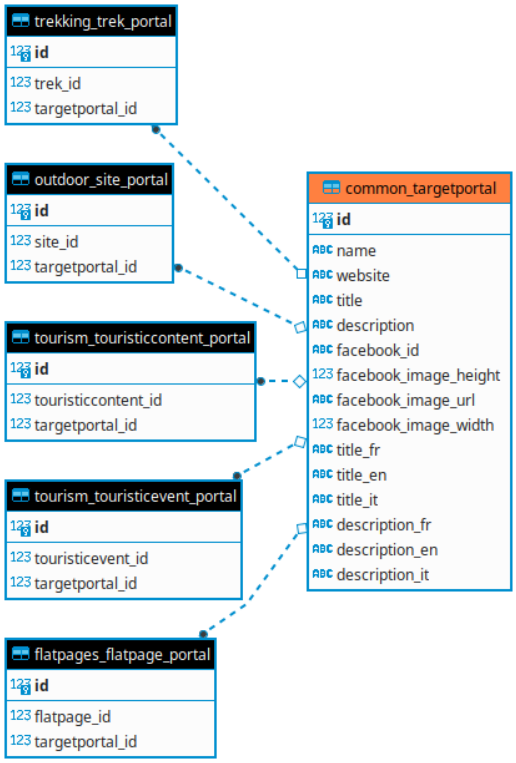


Figure 21: Relations de la table common_targetportal, utilisée par le Geotrek-aggregator

52 https://www.alpesrando.net/
53 https://rando.ecrins-parcnational.fr/

On voit ci-dessus (fig. 21) que seules quelques tables ont le champ nécessaire à la discrimination des données à exporter vers un autre portail. Il n'existe pour l'instant pas de moyen d'agréger des données à l'échelle de Geotrek-admin, notamment les données de gestion.

Pour le Parc national c'est un enjeu important. En effet, si une communauté de communes gardoise renseigne le statut de ses sentiers, leur entretien, etc. dans le Geotrek départemental, ce sont des données qui nous intéressent également !

Par exemple, un Parc national exerce une autorité légale sur sa zone cœur : il est le garant du respect du droit de l'environnement[54], et peut aussi modifier la réglementation si besoin. La définition du plan de circulation étant une de ses prérogatives, connaître le statut foncier des voies en zone cœur est important.

Autre exemple : si le Parc n'a pas vocation à développer des itinéraires de randonnée - c'est plutôt la compétence des collectivités locales -, il crée toutefois des « sentiers de découverte »[55] en partenariat avec les communes : des itinéraires très courts, dont le but est de faire découvrir le territoire grâce à des panneaux explicatifs.

Pour une gestion plus efficace de tous ces types de situations, un accès direct aux données de gestion des sentiers est très intéressant. Cela permettrait par exemple de connaître la date du dernier entretien, le contenu des panneaux directionnels, ou encore le statut foncier des voies.

5.3 Geotrek-admin-aggregator

En l'absence de fonctionnalités permettant une agrégation au niveau de Geotrek-admin, nous avons donc décidé de produire un prototype fonctionnel d'agrégateur : le Geotrek-admin-aggregator. Pour simplifier, à partir de maintenant les termes « agrégateur » et « aggregator » feront uniquement référence à ce prototype, et j'utiliserai le terme Geotrek-rando aggregator si jamais je devais encore en faire mention.

Le but est double :

- à court terme, permettre d'agréger les données du Gard et de la Lozère ;
- à moyen terme, préparer le terrain pour le groupe de travail sur l'agrégateur (et le potentiel groupement de commande) et Makina Corpus. Identifier les écueils et des pistes d'améliorations pour faciliter le processus ultérieur de développement.

Cela restera un prototype et ne sera vraisemblablement pas utilisable en production en-dehors du Parc national des Cévennes, même s'il sera en théorie universel.

54 http://www.parcsnationaux.fr/fr/des-connaissances/protection-et-reglementation/police-de-lenvironnement
55 https://www.cevennes-parcnational.fr/fr/des-decouvertes/parcourir-le-parc/les-musees-et-sentiers-de-decouverte

5.3.1 Réalisation

La première chose qu'il a fallu faire est prendre en main le modèle de données de Geotrek. Comme déjà montré plus haut, il est construit autour de la table *core_path*, au sein de laquelle est stocké le linéaire constituant la fondation de Geotrek. C'est sur cette table que s'effectue la segmentation dynamique. On voit ensuite que *core_topology* est la seule table directement reliée à *core_path*, et contient toutes les géométries des objets Geotrek : des itinéraires (*TREK*), des POI, de la signalétique (*SIGNAGE*), des zones géographiques (CITYEDGE*, DISTRICTEDGE*...), etc.

Ensuite, les tables sont réparties en grandes catégories : *common*, *trekking*, *tourism, infrastructure, signage…* Le modèle relationnel est assez classique, composé de tables « catégorielles » et de tables « d'individus » : *trekking_trek* et *trekking_poi* contiennent des listes d'itinéraires et de points d'intérêts, qui constituent la donnée finale, ce que j'appelle individus ; *trekking_practice* et *tourism_touristiccontenttype* contiennent des listes de catégories (de pratique ou de type d'infrastructure touristique) utilisées pour qualifier les enregistrements d'autres tables (par exemple *trekking_trek* ou *trekking_touristiccontent*). Enfin, des tables de relation « n à n » permettent par exemple de faire le lien entre des itinéraires et des thèmes : un itinéraire peut avoir plusieurs thèmes, et inversement (*trekking_trek_themes* lie *trekking_trek* et *common_theme*).

Une fois que le modèle est maîtrisé, si on importe sans précautions toutes les données voulues, de *core_path* à *trekking_trek_themes* en passant par *signage_blade*, un problème apparaît : la segmentation dynamique. En effet, importer de nouveaux *core_path* risque très fortement d'exécuter un certain nombre de déclencheurs PostgreSQL (ou *triggers*), évaluant si les nouveaux tronçons intersectent ceux les existants, puis découpant tout cela selon les points d'intersection. D'expérience, l'ajout, même manuel sous QGIS, de tronçons à *core_path* a des conséquences imprévisibles, et un nettoyage est quasi systématiquement indispensable. Le problème se situe surtout autour des *trekking_trek*, dont la géométrie peut être fortement modifiée en étant *snappée* sur d'autres *core_path* que ceux sur lesquels elle était calquée par exemple.

Il faut donc ne pas importer *core_path*, et faire de *core_topology* la nouvelle fondation de notre modèle. Pour cela, il est nécessaire de désactiver la segmentation dynamique, ce que permet une option de Geotrek : TREKKING_TOPOLOGY_ENABLED. En définissant sa valeur comme *False*, elle désactive la segmentation dynamique, et donc le besoin d'importer *core_path*. Cependant, cela a comme conséquence que les POI, dont le lien aux itinéraires dépend de la segmentation dynamique, n'apparaissent plus dans l'onglet « À proximité » de Geotrek-rando. Une solution non pérenne est de modifier le code source de Geotrek-admin afin d'inverser la prise en compte de l'option TREKKING_TOPOLOGY_ENABLED.

Dans le fichier *geotrek-admin/lib/python3.6/site-packages/geotrek/trekking/models.py*, il suffit de modifier les fonctions *poi_types(), topology_all_pois()* et *topology_treks()* en ajoutant « not » avant « settings.TREKKING_TOPOLOGY_ENABLED ». Cela permet de conserver les liens spatiaux entre points d'intérêt et itinéraires, tout en désactivant la segmentation dynamique. Il s'agit bien sûr d'un contournement non recommandé en production qui peut avoir des conséquences non prévues.

Mais cela pose bien sûr un autre problème : impossible de désactiver cette option dans une instance déjà en place, car la segmentation dynamique est toujours souhaitée pour le reste des données… Il faut donc passer par une base de données dédiée à l'agrégation. C'est le fonctionnement que nous avons choisi pour la Lozère (fig. 22) :

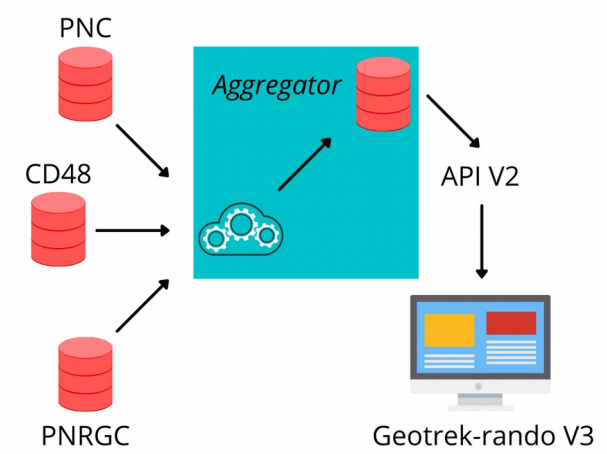


Figure 22: Fonctionnement prévu de l'agrégateur en Lozère

Trois bases de données avec segmentation dynamique activée seront dédiées aux trois parties du territoire, et une base de données avec segmentation dynamique désactivée sera dédiée à l'agrégation, et ensuite à alimenter un portail Geotrek-rando. Il s'agit donc d'une approche de base à base, s'appuyant en majeure partie sur des requêtes SQL.

Du côté de la base du Parc national, il faudra également récupérer les données qui nous intéressent depuis les bases de la Lozère (et potentiellement du Gard). Hors de question de désactiver la segmentation dynamique, alors comment faire ?

S'agissant des itinéraires de randonnée, l'ajout des *core_path* correspondants sera vraisemblablement réalisé de manière manuelle, avec QGIS. Côté Lozère, une seule commune sera à la fois rattachée au Geotrek départemental et au sein de l'aire d'adhésion : Saint-Bauzile, de la communauté de communes Cœur de Lozère. Actuellement, un seul itinéraire traversant cette commune est valorisé sur Destination Cévennes, et il s'agit d'un GR (sentier de Grande Randonnée) qui sera de toute façon présent dans la base du Parc national car traversant une grande partie de son territoire. Côté Gard, si les communautés de communes gardoises cessaient de renseigner le Geotrek du Parc national pour renseigner le Geotrek départemental, il y aurait plus de travail à effectuer.

Pour autant, nous ne partons pas de zéro puisque de nombreux itinéraires, et leurs *core_path* correspondants, sont déjà présents dans notre base. L'ajout et la mise à jour manuelle des itinéraires semble donc réaliste.

S'agissant des autres données soumises à la segmentation dynamique, elles ne dépendent pas de *core_path* de la même manière. Comme ce sont des géométries ponctuelles (points d'intérêts, panneaux signalétiques…), elles n'ont pas besoin de tronçons spécifiques auxquelles se rattacher : elles seront liées au tronçon le plus proche de leur emplacement.

Quant aux données non soumises à la segmentation dynamique (hébergements, animations, commerces…), aucune question n'est à se poser de ce point de vue-là, et l'importation peut se faire sans problème. En effet, un simple calcul de distance entre ces points et la géométrie d'un itinéraire permet à Geotrek de les afficher dans les recommandations de celui-ci (fig. 23).

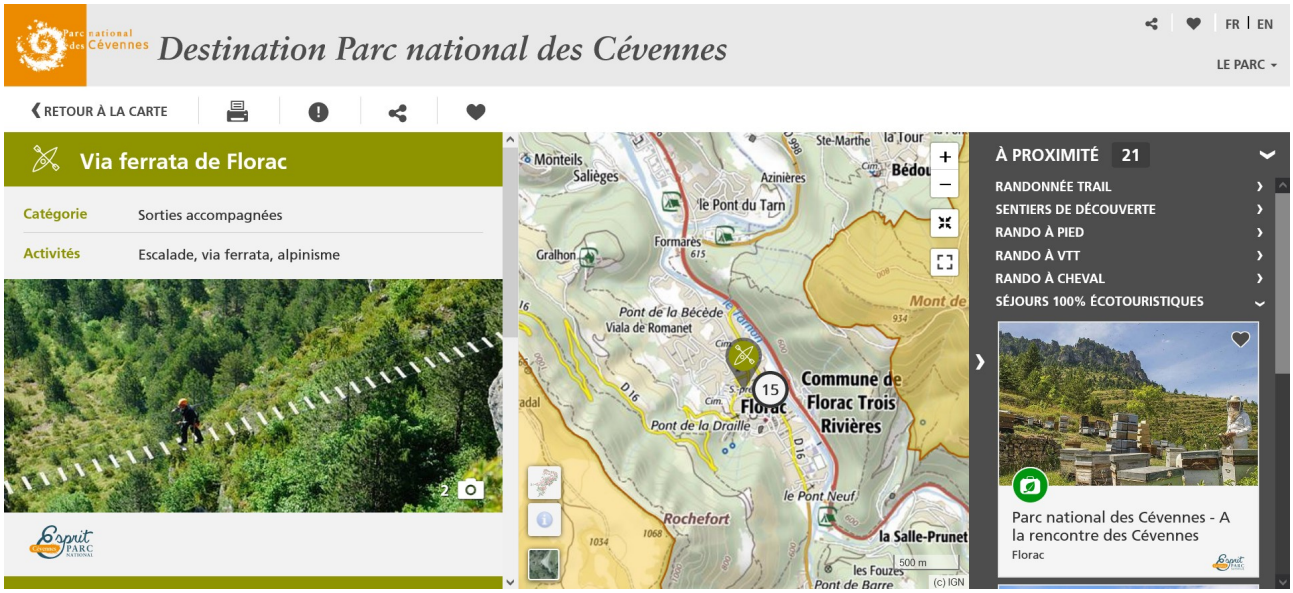


Figure 23: Suggestions de données proches d'une activité - Geotrek-rando

5.3.2 Le SQL

Les premiers essais d'agrégation ont été faits avec les bases du Parc national des Écrins et du Parc national des Cévennes (nous n'avions pas encore accès aux bases de la Lozère ou du Parc naturel régional des Grands Causses).

Une des premières questions qu'il a fallu se poser est celle de la traçabilité des données. Après avoir agrégé des données issues de plusieurs bases, comment reconnaître celles qui proviennent des Cévennes et celles des Écrins ? C'est indispensable pour que les mises à jour se fassent correctement. Si un itinéraire de randonnée a été supprimé du Geotrek des Écrins, il faut pouvoir le supprimer sans risquer de supprimer des itinéraires toujours valides des Cévennes.

Nous avons essayé deux solutions :

- le stockage systématique de l'ancien identifiant, de sa correspondance avec le nouvel identifiant et de la source ;
- passer par des *Universal Unique Identifier* (UUID), un système d'attribution d'identifiants uniques sans contrôle central[56].

La première solution a l'avantage de garantir une traçabilité parfaite, et l'exclusion de tout risque d'erreur de correspondance entre une donnée dans sa base source et dans la base *aggregator*.

L'inconvénient est une relative complexité du procédé : ou bien il faut créer deux nouveaux champs dans chaque table concernée (*db_source* et *id_old* par exemple), ou alors il faut créer une table dédiée pour rassembler toutes les correspondances *via* trois champs (*db_source*, *id_new* et *id_old*). Une table dédiée nécessite moins de modifications de structure de la base de données, mais entraîne une jointure supplémentaire pour toute tentative d'identification.

La deuxième solution a l'avantage de ne nécessiter qu'un seul champ *uuid* à ajouter aux tables concernées, et d'avoir une visée plus large que le seul agrégateur. En effet, si on en revient à l'ouverture des données, un UUID est un moyen de garantir la traçabilité de la donnée, y compris hors du contexte de Geotrek. C'est d'ailleurs un des champs prévus dans le schéma de données.

Le principal inconvénient est qu'il faut générer des UUID dans la base source **avant** toute opération d'agrégation. Cela nécessite donc un accord entre les différentes organisations parties prenantes de l'agrégateur pour mettre en place ce système, qui n'est pas prévu dans Geotrek pour l'instant. Cependant, nous avons soumis l'idée à Makina Corpus, qui se penchera sur la question. Il serait bien plus robuste que l'agrégateur utilise des champs nativement présents dans Geotrek, plutôt que de devoir en créer de nouveaux. Un inconvénient très mineur est que l'unicité d'un UUID ne peut être mathématiquement garantie, même si elle l'est statistiquement. Parmi 2,71 x 10 e18 UUID v4 la probabilité de collision étant de 50 %, nous avons de la marge !

Nous avons décidé de conserver la solution des UUID afin de nous placer dans un cadre plus large, et peut-être initier l'intégration de ce mode d'identification dans Geotrek.

56 https://fr.wikipedia.org/wiki/Universally_unique_identifier

La première étape pour agréger des données est donc la création d'un champ *uuid* pour les tables :

core_topology	*tourism_touristicevent*	*trekking_weblink*
common_attachment	*feedback_report*	*signage_signage*
tourism_informationdesk	*trekking_trek*	*signage_blade*
tourism_touristiccontent	*trekking_poi*	*signage_line*

dans la base de données *aggregator*, ainsi que dans les bases de données source (accompagnée de la génération des UUID pour celles-ci bien sûr).

Pour connecter les bases entre elles, nous avons utilisé des *Foreign Data Wrapper* (FDW), une fonctionnalité de PostgreSQL[57] (non standard SQL) qui permet d'interroger une base de données B depuis une base A en important B dans un schéma de A (fig. 24). Cela fonctionne aussi bien avec des bases locales que distantes, les paramètres classiques de connexion comme l'hôte et le port étant tous disponibles.

```
CREATE EXTENSION IF NOT EXISTS postgres_fdw;

--DROP SERVER IF EXISTS server_pnc CASCADE;
CREATE SERVER IF NOT EXISTS server_pnc
        FOREIGN DATA WRAPPER postgres_fdw
        OPTIONS (host 'localhost', port '5432', dbname 'geotrek_pnc');

CREATE USER MAPPING FOR user
    SERVER server_pnc
    OPTIONS (user 'user', password 'password');

--DROP SCHEMA IF EXISTS pnc;
CREATE SCHEMA pnc;
IMPORT FOREIGN SCHEMA public
    FROM SERVER server_pnc
    INTO pnc;
```

Figure 24: Création d'un Foreign Data Wrapper

Une fois l'extension installée, il faut créer un serveur, que nous nommons ici *server_pnc* pour le Parc national des Cévennes (PNC), qui se connecte à la base *geotrek_pnc* de manière locale et sur le port 5432. Il faut ensuite se donner les droits d'accès à cette base externe en renseignant l'identité de l'utilisateur sous le nom duquel PostgreSQL se connectera au serveur. Enfin, nous pouvons créer un schéma *pnc* dans notre base, et importer dans celui-ci tout le contenu du schéma *public* depuis le serveur.

Cette procédure nous permet d'effectuer librement des requêtes sur les données de la base du Parc national des Cévennes, depuis notre base de données *aggregator*. Attention cependant : il n'y a pas de mise à jour automatique via ce lien. En cas de changement de la structure ou du contenu du schéma ainsi importé, il faudra relancer une partie de ces commandes.

57 https://wiki.postgresql.org/wiki/Foreign_data_wrappers

Une fois les données stockées dans un schéma, il a fallu sélectionner celles que nous trouvions intéressantes à importer dans notre schéma principal, le schéma *public*. La liste est la suivante :

- les itinéraires de randonnée ;
- les points d'intérêt ;
- le contenu touristique (terme spécifique à Geotrek désignant l'ensemble des infrastructures touristiques d'intérêt : musées, hébergements, restaurants, etc.) ;
- les évènements touristiques (appelés animations au Parc national, ils sont gérés *via* Geotrek) ;
- les points d'accueil (offices de tourisme, maisons du Parc, etc.) ;
- la signalétique ;
- les médias ;
- les liens web (informations supplémentaires à ajouter aux itinéraires, comme le lien vers le site des transports en commun locaux, ou le site de Météo France) ;
- les signalements (effectués depuis un formulaire sur Geotrek-rando, ils permettent de faire remonter un problème sur un itinéraire).

Pour cela, un total de 33 tables doit être importé, dont la liste est disponible dans la documentation du dépôt.

La liste des données à importer est bien entendu modifiable, notamment par les agent·es décisionnaires sur ces sujets : au Parc national ce peut être la cheffe du Service Accueil et Sensibilisation ou encore la chargée de mission activités de pleine nature, au conseil départemental la responsable de mission activités de pleine nature, etc. En tant que responsables du système d'information, notre mission est d'assister et conseiller nos collègues tout au long de ces processus que nous mettons en œuvre, mais les décisions qui ne relèvent pas de la technique comme le choix des itinéraires à valoriser, ou des types de données que nous souhaitons importer ne sont pas prises par nous.

Par exemple, l'Office Français de la Biodiversité[58], l'établissement public dédié à la protection et la restauration de la biodiversité auquel sont rattachés les parcs nationaux, a lancé la marque Esprit parc national[59] pour valoriser les partenaires touristiques considérés comme compatibles avec les valeurs des parcs nationaux. Ainsi, seuls les hébergements, restaurants, prestataires sportifs labellisés Esprit parc national sont valorisés sur Destination Cévennes. Si nous importons des données depuis le Gard, il est probable que nous ne voulions pas de tout le contenu touristique non labellisé et devions opérer un tri : une tâche technique...qui découle d'une décision politique.

58 https://ofb.gouv.fr/
59 https://www.espritparcnational.com/

Construire les requêtes d'insertion de ces données nécessite beaucoup de jointures. La table *trekking_trek* en nécessite six, *feedback_report* cinq, etc. Les champs à importer sont également nombreux (cinquante pour *trekking_trek*). Les requêtes ne sont pas complexes dans l'absolu, mais fastidieuses à écrire, et ainsi très sujettes aux erreurs (fig. 25) :

```
-- Insertion TREKKING_TREK
INSERT INTO trekking_trek
    (published, publication_date, "name", review, topo_object_id, departure, arrival,
    description_teaser, description, ambiance, "access", disabled_infrastructure, duration,
    advised_parking, parking_location, public_transport, advice, points_reference, eid,
    eid2, difficulty_id, practice_id, route_id, structure_id, reservation_id, reservation_system_id,
    arrival_fr, arrival_en, ambiance_fr, ambiance_en, departure_fr, departure_en, access_fr, access_en,
    advised_parking_fr, advised_parking_en, disabled_infrastructure_fr, disabled_infrastructure_en,
    published_fr, published_en, advice_fr, advice_en, name_fr, name_en,
    public_transport_fr, public_transport_en, description_fr, description_en,
    description_teaser_fr, description_teaser_en, uuid)
SELECT
    published, publication_date, "source"."name", review, ct.id, departure, arrival, description_teaser,
    "source".description, ambiance, "access", disabled_infrastructure, duration,
    advised_parking, parking_location,public_transport, advice, points_reference, eid,
    eid2, dl.id, tp.id, tr.id, st.id, reservation_id, crs.id,
    arrival_fr, arrival_en, ambiance_fr,ambiance_en, departure_fr, departure_en, access_fr, access_en,
    advised_parking_fr, advised_parking_en, disabled_infrastructure_fr, disabled_infrastructure_en,
    published_fr, published_en, advice_fr, advice_en, "source".name_fr, "source".name_en,
    public_transport_fr, public_transport_en, "source".description_fr, "source".description_en,
    description_teaser_fr, description_teaser_en, "source".uuid
FROM pnc.trekking_trek "source"
LEFT JOIN core_topology ct ON "source".uuid = ct.uuid
LEFT JOIN (SELECT a.id, cc.* FROM trekking_difficultylevel a
           LEFT JOIN common_correspondances cc
           ON cc.table_name ILIKE 'trekking_difficultylevel' AND a.difficulty ILIKE cc.name_new) dl
ON dl.bdd_source ILIKE csp."name" AND "source".difficulty_id = dl.id_old::integer
LEFT JOIN (SELECT a.id, cc.* FROM trekking_practice a
           LEFT JOIN common_correspondances cc
           ON cc.table_name ILIKE 'trekking_practice' AND a."name" ILIKE cc.name_new) tp
ON tp.bdd_source ILIKE csp."name" AND "source".practice_id = tp.id_old::integer
LEFT JOIN (SELECT a.id, cc.* FROM trekking_route a
           LEFT JOIN common_correspondances cc
           ON cc.table_name ILIKE 'trekking_route' AND a.route ILIKE cc.name_new) tr
ON tr.bdd_source ILIKE csp."name" AND "source".route_id = tr.id_old::integer
LEFT JOIN (SELECT a.id, cc.* FROM authent_structure a
           JOIN common_correspondances cc
           ON cc.table_name ILIKE 'authent_structure' AND a."name" ILIKE cc.name_new) st
ON st.bdd_source ILIKE csp."name" AND "source".structure_id = st.id_old::integer
LEFT JOIN (SELECT a.id, cc.* FROM common_reservationsystem a
           JOIN common_correspondances cc
           ON cc.table_name ILIKE 'common_reservationsystem' AND a."name" ILIKE cc.name_new) crs
ON crs.bdd_source ILIKE csp."name" AND "source".structure_id = crs.id_old::integer;
```

Figure 25: Requête d'insertion de la table trekking_trek

La structure présentée ici subit peu de variations selon les tables que l'on essaie d'importer. Après nous être assuré-es du bon fonctionnement de ces requêtes, nous avons identifié les blocs de code qui pourraient être générés automatiquement plutôt qu'écrits au cas par cas.

Si on tente de remplacer ces blocs par des variables, cela donnerait par exemple (fig. 26) :

```
INSERT INTO {table_name} ({columns_names{table_name}})
SELECT {columns_notforeignkeys_names} + {columns_foreignkeys_names}
FROM {source_name}.{table_name} "source"
LEFT JOIN {related_table} rt ON rt.uuid = source.uuid
LEFT JOIN (SELECT a.id, cc.* FROM {category_table} LEFT JOIN common_correspondances cc
ON cc.{table_name} ILIKE '{table_name}' AND a.{category_name} ILIKE cc.name_new) dl
ON dl.bdd_source ILIKE  '{source_name}' AND "source".{category_column} = dl.id_old
etc.
```

Figure 26: Paramétrisation d'une requête d'insertion

On voit qu'il est possible de réduire à la portion congrue ce qui est commun entre toutes les tables, et qu'on peut paramétriser ou automatiser une bonne partie des requêtes. À partir de là, il devenait plus simple d'abandonner le pur SQL et de commencer à construire une application en Python. Cela va faciliter la génération paramétrisée de requêtes SQL ainsi que leur exécution.

5.3.3 Application Flask

Amandine Sahl, ma tutrice, a créé les bases de cette application en utilisant Flask, un micro framework open source[60]. Flask est issu d'un poisson d'avril présentant un framework capable de faire fonctionner une application web complète en six lignes de code, sans installation ni dépendances. L'idée fut tellement appréciée qu'elle en devint réalité : c'est aujourd'hui un des frameworks Python les plus utilisés avec Django, sa simplicité d'utilisation étant son principal atout.

Nous utilisons également SQLAlchemy[61], un *mapping* objet-relationnel, soit un programme convertissant un modèle relationnel (nos bases PostgreSQL) en modèle objet. Il nous permet de manipuler nos bases directement dans un modèle objet, via des classes Python. L'extension Flask-SQLAlchemy[62] nous aide dans cette tâche.

L'application a deux objectifs :

- gérer la mise en correspondance des catégories entre bases ;
- importer et mettre à jour les données dans la base *aggregator*.

5.3.4 Le *mapping*

Comme mentionné plus haut, il faut distinguer deux types de tables : les tables catégorielles, et les tables « d'individus ». Un individu étant une donnée « finale » dans le cadre de notre système d'information. Un itinéraire, un point d'intérêt, ont vocation à être présentés sur Destination Cévennes. Une catégorie de pratique ou un thème n'ont pas de valeur ou d'intérêt intrinsèque, ils servent à qualifier les individus.

À priori, un individu dans la base des Écrins ne peut pas être le même qu'un individu dans la base des Cévennes, et il en va de même pour la Lozère et les Cévennes : si la répartition des communautés de communes est efficace, il ne devrait pas y avoir de points d'intérêt ou d'itinéraires en commun entre deux bases. Une exception étant par exemple les itinéraires longs de type sentiers de Grande Randonnée, traversant par définition plusieurs zones géographiques.

60 https://flask.palletsprojects.com/en/2.0.x/
61 https://www.sqlalchemy.org/
62 https://flask-sqlalchemy.palletsprojects.com/

Au contraire, les thèmes peuvent être partiellement ou même strictement identiques entre deux bases. Il n'y a pas d'intérêt à importer tous les thèmes depuis la base des Écrins si nous avons déjà les mêmes, juste parce qu'ils viennent des Écrins. Cela ne leur donne pas de valeur propre. « Archéologie » dans les Écrins peut être strictement équivalent à « Archéologie » dans les Cévennes, et nous ne voulons en garder qu'une seule occurrence.

Il faut donc construire un *mapping*, une mise en correspondance des catégories provenant de diverses bases.

Pour cela, les gestionnaires de la base *aggregator* doivent d'abord remplir toutes les tables catégorielles de celle-ci avec les catégories qu'elles/ils souhaitent y voir figurer. Il est possible de reprendre les catégories existant déjà dans les bases source, mais le remplissage doit idéalement être complet avant de poursuivre. Les tables concernées sont :

authent_structure

common_theme

common_recordsource

common_label

common_filetype

common_targetportal

common_reservationsystem

trekking_difficultylevel

trekking_weblinkcategory

trekking_practice

trekking_accessibility

trekking_route

trekking_poitype

trekking_treknetwork

tourism_touristiceventtype

tourism_informationdesktype

tourism_touristiccontentcategory

feedback_reportstatus

feedback_reportactivity

feedback_reportproblemmagnitude

feedback_reportcategory

C'est ensuite que l'application peut être démarrée. C'est une application libre, son code peut donc être téléchargé sur le dépôt GitHub du Parc national des Cévennes : https://github.com/PnCevennes/geotrek_admin_agg/.

Trois modules sont installables via *pip* : *flask*, *flask-sqlalchemy* et *psycopg2*. Un fichier *package-lock.json* permet d'installer les dépendances nécessaires au fonctionnement de l'application web de *mapping* avec npm.

La première étape est de créer le schéma de la base de données, c'est-à-dire de créer les deux tables qui nous serviront à gérer ce *mapping* dans PostgreSQL. La commande *flask create_db_schema* se sert des modèles renseignés dans models.py pour créer ces tables (fig. 27) :

```python
class GeotrekAggCorrespondances(DB.Model):
    __tablename__ = "geotrekagg_correspondances"
    __table_args__ = {"schema": "public"}
    id = DB.Column(
        DB.Integer,
        primary_key=True
    )
    bdd_source = DB.Column(DB.Unicode)
    table_origin = DB.Column(DB.Unicode)
    id_origin = DB.Column(DB.Integer)
    label_origin = DB.Column(DB.Unicode)
    id_destination = DB.Column(DB.Integer)
    UniqueConstraint(bdd_source, table_origin, id_origin)

class GeotrekAggSources(DB.Model):
    __tablename__ = "geotrekagg_sources"
    __table_args__ = {"schema": "public"}
    id = DB.Column(
        DB.Integer,
        primary_key=True
    )
    bdd_source = DB.Column(DB.Unicode, unique=True)
    url = DB.Column(DB.Unicode)
```

Figure 27: Extrait de la commande create_db_schema

La commande *flask add_source* permet d'ajouter une base de données source à l'agrégateur (fig. 28) :

- un nom et une url sont ajoutés à la table geotrekagg_sources ;
- un FDW est créé grâce aux informations de connexion renseignées.

```python
@click.command("add_source")
@click.option('-n', '--name', 'name', required=True, type=str)
@click.option('-h', '--host', 'host', required=True, type=str)
@click.option('-p', '--port', 'port', type=int, default=5432, show_default=True)
@click.option('-d', '--db_name', 'db_name', required=True, type=str)
@click.option('-U', '--user', 'user', required=True, type=str)
@click.option('-P', '--password', 'password', required=True, type=str)
@click.option('-u', '--url', 'url', required=True, type=str)
@click.option('-o', '--overwrite', 'overwrite', is_flag=True)
@with_appcontext
def add_source(name, host, port, db_name, user, password, overwrite, url):
    """
        Création d'une source pour l'aggrégateur

        Args:
            name ([string]): nom de la source
            host ([string]): hote de la base geotrek
            port ([int]): port de postgresql de l'hote
            db_name([string]): nom de la base de données
            user ([string]): utilisateur (ayant des droits de lecture sur la base)
            password ([string]): mot de passe de l'utilisateur
            url ([string]): url de la source (Geotrek-admin)
            overwrite ([]): écraser la source si déjà existante
    """
```

Figure 28: Extrait de la commande add_source

La commande *flask import_mapping* (fig. 29) permet d'importer les catégories d'une base de données source (précisée en option de la commande) et d'effectuer un *mapping* automatique entre celles-ci et les catégories renseignées dans la base de données *aggregator*.

Le script commence par supprimer les catégories qui ne sont plus présentes dans la base de données source. Il insère ensuite toutes les catégories existant dans la base source mais pas encore dans la table *geotrekagg_correspondances*. Enfin, si le nom de la catégorie correspond au nom d'une catégorie existant dans la base *aggregator* (accents et casse non prises en compte), il remplit automatiquement le champ *id_origin* avec l'identifiant de la catégorie correspondante.

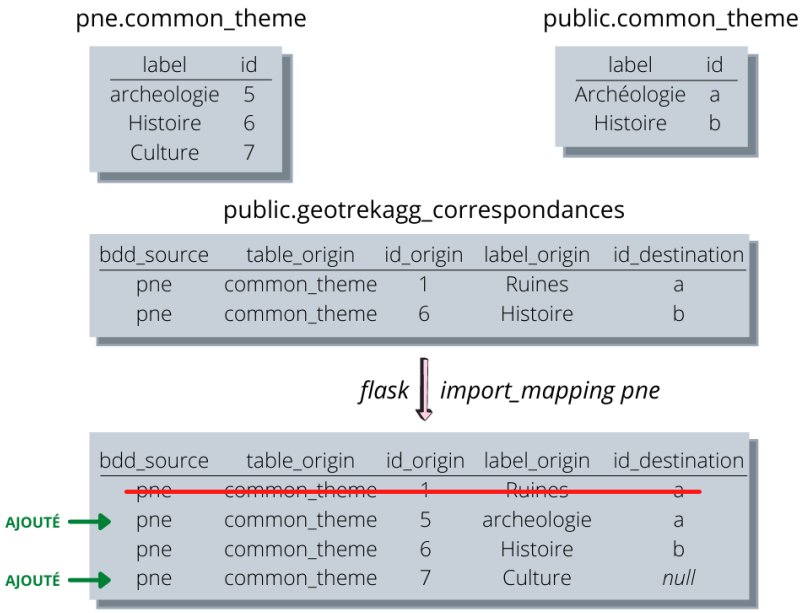


Figure 29: Fonctionnement de l'import des catégories

Dans l'exemple ci-contre (fig. 29), la catégorie « Ruines », n'existant plus dans *pne.common_theme*, est supprimée de *geotrekagg_correspondances*, alors que les catégories « archeologie » et « Culture » sont ajoutées. La catégorie « Histoire » n'ayant pas été modifiée, elle n'est pas altérée. La fonction *auto_mapping* rattache automatiquement « archeologie » à « Archéologie » et lui attribue « a » comme *id_destination*. La catégorie « Culture » n'existant pas dans le schéma public (le schéma par défaut de la base *aggregator*), aucun *id_destination* n'est renseigné. Deux choix s'offrent alors aux gestionnaires : ajouter une catégorie « Culture » au schéma public si elle est souhaitée ; rattacher manuellement la catégorie « Culture » à une catégorie existante, en l'occurrence « Histoire » ou « Archéologie ». Laisser une correspondance non renseignée est fortement déconseillé car cela pourrait entraîner la non-importation de toutes les données utilisant cette catégorie dans la base de données source.

Une fois le *mapping* automatique effectué, l'application permet de faire un *mapping* manuel via une interface web (fig. 30). Il suffit pour cela de démarrer l'application avec la commande *flask run* et de se connecter à *localhost:5000*. Il s'agit d'une représentation graphique de la table *geotrekagg_correspondances*, à cette différence près que les identifiants des catégories ne sont pas affichés, et que le nom de la catégorie de l'*aggregator* l'est. Ce dernier est obtenu par jointure avec le champ *id_destination*.

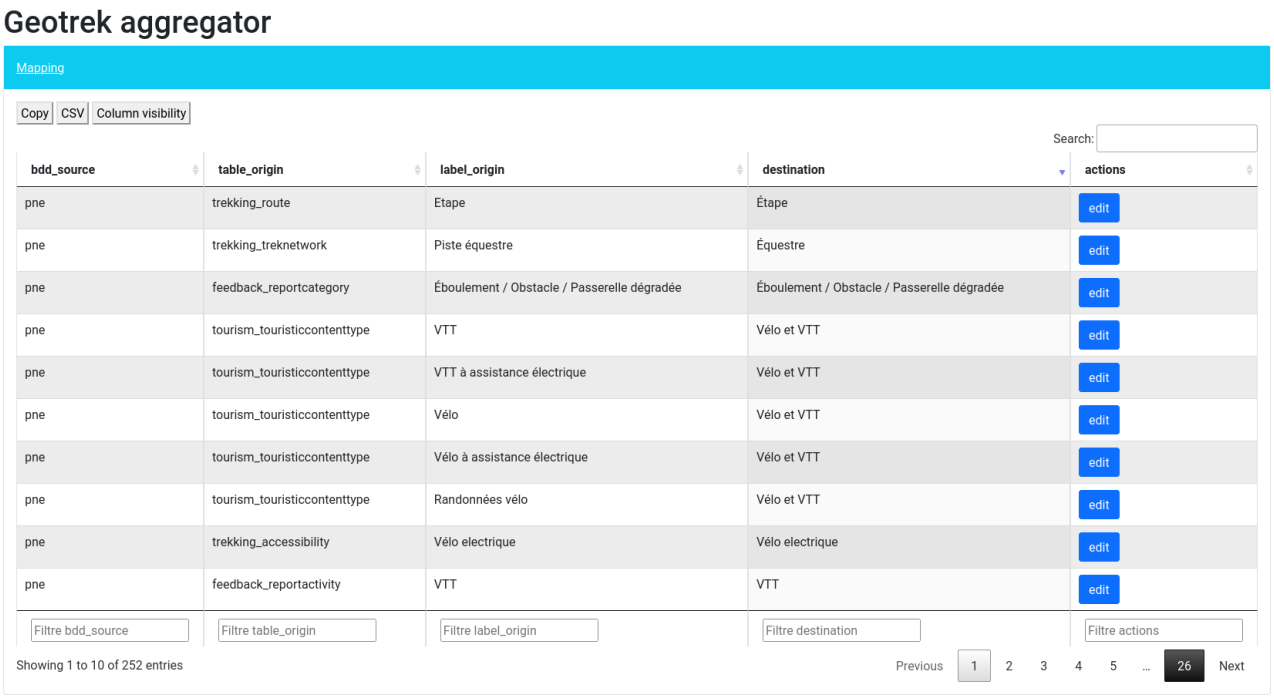


Figure 30: Interface web de mapping

Un clic sur le bouton « edit » donne accès à une liste déroulante de toutes les valeurs possibles pour la correspondance, c'est-à-dire toutes les valeurs existantes dans la table à laquelle appartient la catégorie. Il n'est pas possible d'ajouter un nouveau nom de catégorie, car cela reviendrait à créer un nouvel enregistrement dans la table, pour lequel il faudrait aussi remplir les autres colonnes, etc. Ce n'est pas l'endroit pour cela, et il est donc important que le travail de définition et renseignement des catégories souhaitées ait eu lieu avant de lancer cette application.

5.3.5 L'agrégation

Une fois toutes les correspondances correctement faites, l'importation peut être lancée. Pour ce faire, deux fonctions SQL doivent être créées avec la commande *flask create_functions*, afin de générer automatiquement des sous-requêtes pour les champs qui sont aussi des clefs étrangères (fig. 31). Chaque jointure est ainsi remplacée par une sous-requête placée directement dans le corps du *SELECT* de la requête d'insertion.

```
-------FONCTION D'OBTENTION DU NOUVEL ID D'UNE CATEGORIE
        CREATE OR REPLACE FUNCTION public.geotrekagg_get_category_id(
            _initial_id integer,
            _table_origin character varying,
            _db_source character varying
        )
        RETURNS integer
        LANGUAGE plpgsql
        AS $function$
        BEGIN
            RETURN  (
                SELECT id_destination
             FROM geotrekagg_correspondances gc
             WHERE
                    id_origin = _initial_id
                    AND table_origin = _table_origin
                    AND bdd_source = _db_source
            );
        END;
        $function$;
```

Figure 31: Requête de création d'une des fonctions SQL

Ainsi, la requête d'insertion de la table *trekking_trek* générée par la commande *flask populate_gta pne* ressemble maintenant à (fig. 32) :

```
INSERT INTO trekking_trek (  - - liste de toutes les colonnes - -)
            SELECT *
            FROM (
                SELECT "published","publication_date","name","review",
                        geotrekagg_get_foreign_key (
                            topo_object_id::varchar, 'trekking_trek','core_topology', 'topo_object_id', 'id', 'pne'
                        ) as topo_object_id
                    ,"departure","arrival","description_teaser","description","ambiance","access",
                    "disabled_infrastructure","duration","advised_parking","parking_location",
                    "public_transport","advice","points_reference","eid","eid2",
                    geotrekagg_get_id_correspondance (
                        difficulty_id, 'trekking_difficultylevel', 'pne'
                    ) as difficulty_id
                    ,geotrekagg_get_id_correspondance (
                        practice_id, 'trekking_practice', 'pne'
                    ) as practice_id
                    ,geotrekagg_get_id_correspondance (
                        route_id, 'trekking_route', 'pne'
                    ) as route_id
                    ,geotrekagg_get_id_correspondance (
                        structure_id, 'authent_structure', 'pne'
                    ) as structure_id
                    ,"arrival_en","arrival_fr","description_en","description_fr","advice_en",
                    "advice_fr","description_teaser_en","description_teaser_fr","ambiance_en",
                    "ambiance_fr","departure_en","departure_fr","disabled_infrastructure_en",
                    "disabled_infrastructure_fr","access_en","access_fr","public_transport_en",
                    "public_transport_fr","published_en","published_fr","advised_parking_en",
                    "advised_parking_fr","name_en","name_fr","reservation_id",
                    geotrekagg_get_id_correspondance (
                        reservation_system_id, 'common_reservationsystem', 'pne'
                    ) as reservation_system_id
                    ,"uuid"
                FROM pne.trekking_trek
            ) a
             WHERE  NOT structure_id IS NULL ;
```

Figure 32: Requête d'insertion de trekking_trek

Si toutes les catégories sont bien renseignées et qu'aucune erreur n'a été faite, l'application commence par supprimer de la base *aggregator*, table par table, toutes les données dont l'uuid est aussi présent dans la base source. Nous avons trouvé que c'était la meilleure façon de gérer les mises à jour de champs : vérifier et mettre à jour chaque champ de chaque donnée individuellement nous a paru trop lourd à mettre en place en SQL. Le programme supprime donc toutes les données, pour les réinsérer par la suite, ce qui garantit qu'elles soient totalement à jour.

```
for table in reversed(IMPORT_MODEL):
        table_object = MappingObject(
            DB=DB,
            data_source=source,
            table_name=table,
            table_def=IMPORT_MODEL[table]
        )
        sql_d[table] = table_object.generate_sql_delete()

    # Alimentation du dictionnaire sql_i table par table
    for table in IMPORT_MODEL:
        table_object = MappingObject(
            DB=DB,
            data_source=source,
            table_name=table,
            table_def=IMPORT_MODEL[table]
        )
        sql_i[table] = table_object.generate_sql_insert()

    # Essai d'exécution des requêtes, puis commit de celles-ci en cas de succès
    try:
        DB.session.execute(f"""
            ALTER TABLE core_topology DISABLE TRIGGER core_topology_latest_updated_d_tgr;

            DELETE FROM common_attachment p
            USING {name}.common_attachment s
            WHERE s.uuid = p.uuid;
        """)
        for key, value in sql_d.items():
            click.echo(f" -- Deleting table {key}...")
            DB.session.execute(value)
            click.echo(f" -- {key} data deleted!\n")
        for key, value in sql_i.items():
            click.echo(f" -- Importing table {key}...")
            DB.session.execute(value)
            click.echo(f" -- {key} data inserted!\n")
        DB.session.execute("""
            UPDATE core_topology SET date_update = NOW()
            WHERE id IN (SELECT id FROM core_topology ORDER BY date_update DESC LIMIT 1);

            ALTER TABLE core_topology ENABLE TRIGGER core_topology_latest_updated_d_tgr;
        """)
        DB.session.commit()
        click.echo(click.style('Transaction committed', fg='green'))
```

Figure 33: Extrait de la commande populate_gta

Cet extrait (fig. 33) montre que le SQL est construit à partir de la variable IMPORT_MODEL, issue du fichier *env.py*. IMPORT_MODEL est une représentation de l'architecture relationnelle des tables que nous importons sous forme de dictionnaire. Les clefs de chaque sous-dictionnaire permettent de construire les requêtes pour les champs « spéciaux », c'est-à-dire les clefs étrangères et les champs soumis à une contrainte de non-nullité. C'est aussi de cette manière que nous gérons les tables de correspondance comme *trekking_trek_themes* ou *trekking_orderedtrekchild*. Ces tables correspondent aux relations de plusieurs à plusieurs : un itinéraire peut avoir plusieurs thèmes et inversement.

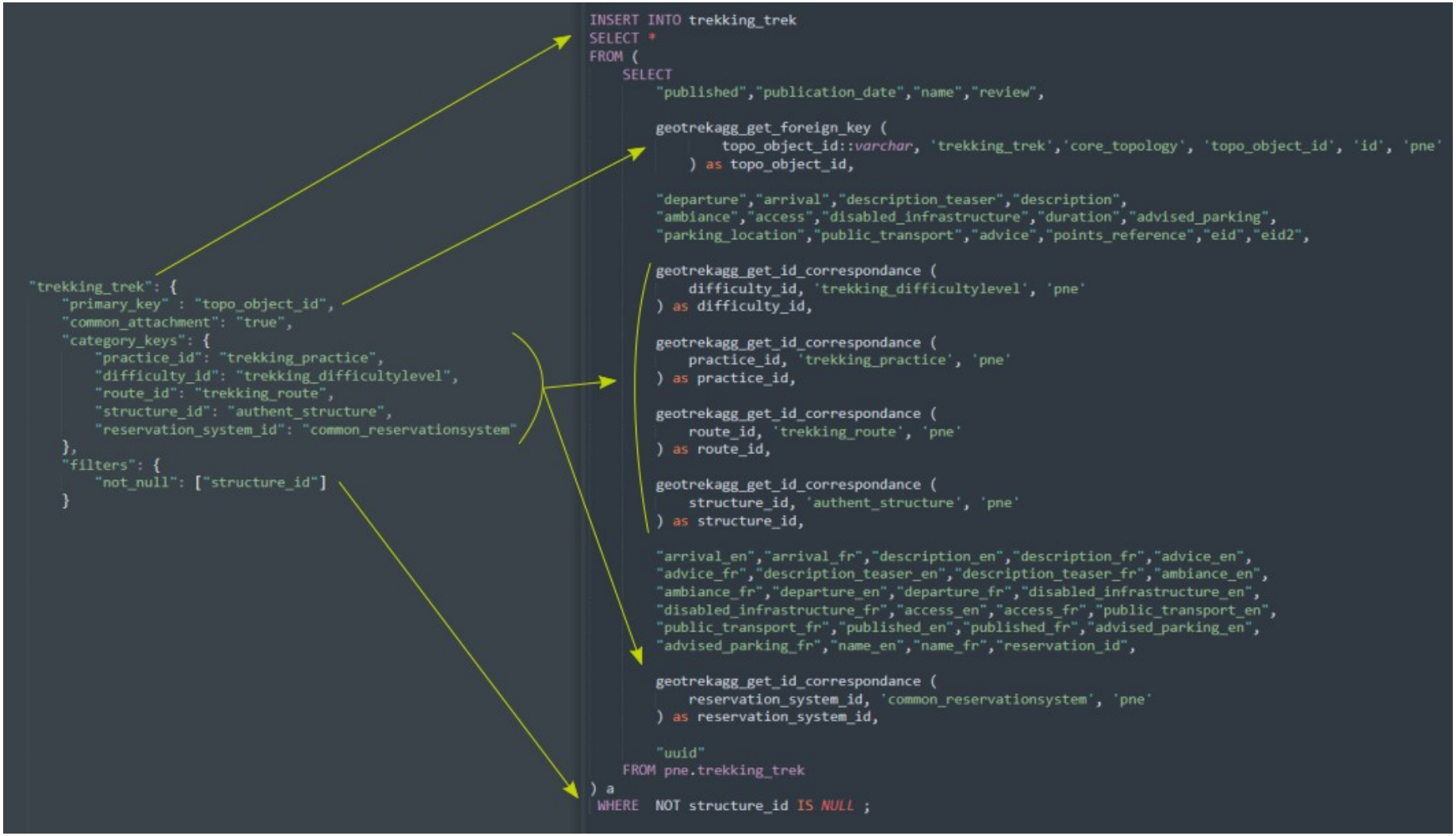


Figure 34: Utilisation du dictionnaire IMPORT_MODEL pour construire la requête d'insertion

On voit bien ici le lien entre le dictionnaire et la construction du SQL, qui se fait concrètement via un ensemble de fonctions et de classes Python qu'il serait fastidieux de détailler ici. Il aurait sûrement été possible de tirer parti de SQLAlchemy pour arriver au même résultat sans l'intermédiaire d'un dictionnaire rempli manuellement, mais cette solution nous a semblé relativement simple et efficace pour un prototype.

Une des difficultés rencontrées a été la nécessité d'insérer et de supprimer les tables dans un ordre précis. En effet le jeu des clefs étrangères ne permet pas de supprimer les données de *trekking_trek* avant celles de *trekking_trek_themes* par exemple. Et l'ordre inverse est le bon lors de l'insertion. Après avoir longtemps manipulé le dictionnaire et essayé de nouvelles clefs...la solution était simplement de lire le dictionnaire à l'envers pour construire la requête de suppression.

Les opérations de suppression et d'insertion sont effectuées au sein d'une session SQLAlchemy. C'est-à-dire qu'elles sont essayées sur la base de données, mais pas exécutées. De cette manière, toute erreur qui surviendrait interromprait la session et aucune donnée n'aurait été supprimée ou insérée. C'est un système de gestion d'erreurs de type « tout ou rien » : pour qu'une modification soit apportée à la base *aggregator* il faut que toutes les modifications soient valides. Cet élément de sécurité est rudimentaire mais robuste.

Une fois la commande *populate_gta* réussie, la base *aggregator* est prête à être affichée dans l'interface de Geotrek-admin !

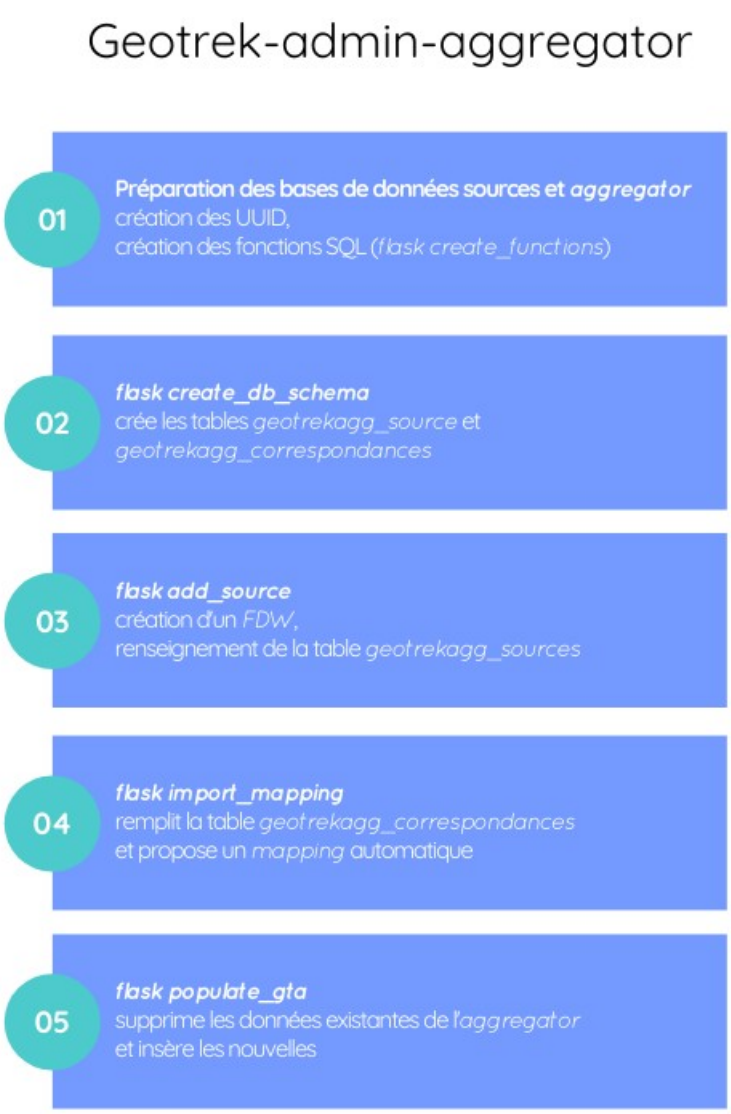


Figure 35: Récapitulatif du fonctionnement de l'application

5.3.6 Bilan de geotrek-admin-aggregator

La publication de Geotrek-rando V3 change la façon dont Geotrek-rando et Geotrek-admin interagissent en passant par une réelle API. Pour l'instant Geotrek-rando V2 utilise une API « statique » fournissant des fichiers GeoJSON. Dans le cadre du développement de Geotrek-rando V3, une API V2 a été créée, elle est maintenant « dynamique » et autorise l'usage de paramètres. Le Parc naturel régional des Grands Causses et le Parc national des Écrins ont déjà adopté cette version, et le Parc national des Cévennes devrait également l'adopter en 2022. L'approche de base à base de ce Geotrek-admin-aggregator pourrait donc être modernisée en utilisant l'API version 2 de Geotrek-admin.

Mis à part cette amélioration possible, l'application fonctionne correctement et nous sommes arrivé·es à agréger efficacement plusieurs bases de données aux structures différentes (le Parc national des Écrins utilise le module Outdoor et une langue de plus que nous, l'italien) dans un agrégateur fonctionnel.

Cependant, nous nous sommes récemment rendu compte d'un problème avec l'utilisation des UUID : la manière dont nous supprimons les données de la base *aggregator* ne permet pas de supprimer des données importées précédemment mais depuis supprimées de la base source. En effet, nous ne supprimons que les données dont l'UUID est présent dans la base source. Avec cette méthode, si un itinéraire de randonnée a disparu de la base des Écrins, il restera éternellement présent dans la base *aggregator*. C'est un problème à corriger ultérieurement.

Enfin, si nous devions agréger les données du Gard, un autre problème se poserait à nous : le réseau de sentiers de référence évolue en ce moment-même et nous devons l'intégrer à notre propre référentiel linéaire.

6. Linéaire

6.1 Le linéaire du Parc national

Dans Geotrek, aucun linéaire n'est fourni avec le logiciel : charge à l'organisation gestionnaire de choisir et importer le sien. Dès 2014, le Parc national des Cévennes a décidé de travailler avec les données de la BD TOPO[63] de l'IGN, une référence pour les établissements publics, pour le plan de circulation. Cependant, le contenu de cette base n'est pas parfaitement topologique, c'est-à-dire que de nombreuses imperfections existent à la jointure entre les tronçons. Amandine Sahl a donc topologisé ces données afin de les rendre compatibles avec Geotrek. Toutes les données de l'emprise de l'aire optimale d'adhésion du Parc national des Cévennes ont été traitées et importées.

Depuis, de nombreux ajouts ou corrections manuelles sont venues altérer ce réseau : ici un sentier qui n'était pas présent, là un chemin dont la géométrie semblait trop éloignée des relevés GNSS[64], etc. Aujourd'hui le référentiel du Parc national n'a plus grand-chose à voir avec la BD TOPO : d'abord parce que les opérations de topologisation ont fortement modifié la structure et même le nombre de tronçons, ensuite parce que ces opérations ont empêché toute mise à jour vers des versions plus récentes de la base, et enfin en raison des ajouts et corrections manuelles effectuées au fil du temps. Le Parc national a donc son propre référentiel, issu de la BD TOPO de l'IGN tout en ayant divergé de celle-ci.

Ce référentiel a deux utilités selon la zone géographique :

- dans l'aire d'adhésion (zone cœur comprise), il sert à créer des itinéraires de randonnée, dont les géométries doivent être calquées sur lui ;
- dans la zone cœur, le plan de circulation est défini à partir de lui.

63 https://geoservices.ign.fr/bdtopo
64 *Global Navigation Satellite System*

6.2 PDESI et PDIPR

Par ailleurs, on rappelle que les collectivités locales ont la charge de la création et du maintien du Réseau Local d'Espaces, Site et Itinéraires (RLESI) sur leur territoire. Les RLESI sont des réseaux de chemins et sentiers considérés comme d'intérêt touristique pour la randonnée. Il ne s'agit pas d'itinéraires dans le sens où il n'y a pas de notion de point de départ et d'arrivée : sont uniquement inscrits au RLESI des linéaires sans scénarisation touristique. Le Plan Départemental des Itinéraires de Promenade et de Randonnée[65] (PDIPR), lui, est défini selon le PDESI (et donc les RLESI).

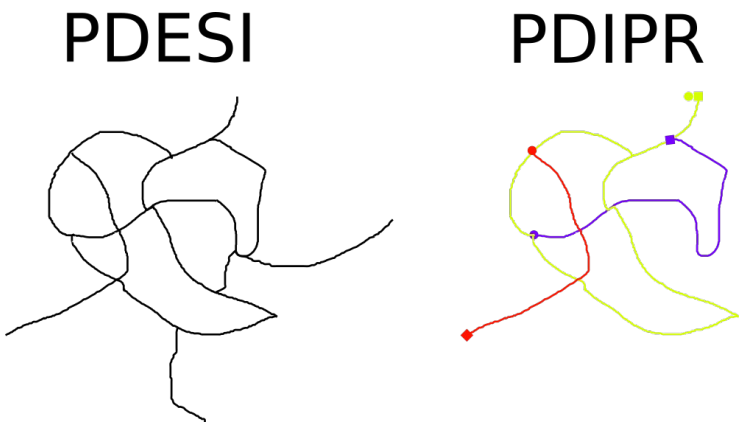


Figure 36: Relation entre PDESI et PDIPR

On voit sur ce schéma (fig. 36) la différence entre le PDIPR et le PDESI : les itinéraires inscrits au PDIPR doivent forcément emprunter le réseau constituant le PDESI. Par contre toutes les voies du PDESI ne sont pas forcément empruntées par un itinéraire inscrit au PDIPR. Le PDIPR est ainsi une scénarisation touristique du PDESI. Les collectivités locales peuvent proposer l'inscription d'itinéraires au PDIPR en remplissant un dossier de candidature adressé au conseil départemental. Les RLESI ont donc vocation à devenir les réseaux linéaires de référence pour la randonnée et la valorisation touristique.

Les RLESI ne sont pas forcément créés à partir de la BD TOPO : ils peuvent être entièrement constitués à partir de relevés GNSS sur le terrain par exemple, de données OpenStreetMap, du cadastre bien sûr, et même d'un mélange de plusieurs de ces sources. A partir de là, lorsqu'une communauté de communes souhaite renseigner un itinéraire construit sur son RLESI, la géométrie de celui-ci a de bonnes chances de différer du référentiel du Parc national. Des chemins peuvent être totalement absents de notre référentiel, auquel cas il faut les ajouter manuellement.

65 https://www.sportsdenature.gouv.fr/publications/outils-mobilisables-pour-perenniser-acces-lieux-de-pratique/pdipr

On peut aussi observer des divergences dans les tracés, souvent seulement de quelques mètres (surtout sous couvert forestier, qui empêche la géolocalisation par image satellite et perturbe les relevés GNSS), mais parfois de plusieurs dizaines.

Dans ce cas, une décision doit être prise : conserver le tracé initial (Parc national), le remplacer intégralement par le nouveau (collectivité), ou bien évaluer à partir d'images satellites, de données OpenStreetMap et de relevés GNSS à quels endroits quel tracé semble le plus juste (fig. 37). C'était jusqu'à présent le fonctionnement adopté.

Figure 37: Illustration des différences entre bases de données IGN et OSM

Or, la question ne devrait pas se poser en-dehors de la zone cœur. En effet, le Parc national n'a aucune compétence légale pour définir les itinéraires ou les chemins d'intérêt. La valorisation des itinéraires sur la plateforme Destination Cévennes est son choix, mais le tracé et l'entretien de ceux-ci est une compétence des collectivités locales. Partant de ce fait, les RLESI devraient devenir la référence unique en matière de géométrie sur le territoire qu'ils couvrent. C'est d'ailleurs dans ce sens que vont les différentes conventions signées avec les collectivités locales autour de la mise à disposition de Geotrek et des itinéraires de randonnée. En-dehors de la zone cœur, le Parc national n'a d'ailleurs besoin d'aucune autre voie que celles inscrites au RLESI, puisque seuls les itinéraires de randonnée utilisent le linéaire. Il semble donc logique, et c'est ce vers quoi nous travaillons, de remplacer notre linéaire de référence par les RLESI partout où c'est possible.

En zone cœur la question est plus compliquée car nous avons besoin d'un linéaire tendant vers l'exhaustivité pour gérer le plan de circulation, qui est la référence légale pour toutes les voies. Les RLESI ne sont pas exhaustifs car ils ne recensent que les chemins considérés comme intéressants : nous ne pouvons donc pas remplacer l'intégralité de notre référentiel par le RLESI. Il faut à la fois intégrer le RLESI sur les sentiers en faisant partie, et conserver le reste de notre linéaire.

La segmentation dynamique implique que chaque nouvelle intersection entre deux tronçons découpe ceux-ci au point de contact. On ne peut donc pas se contenter d'une simple opération de remplacement du linéaire, car de nombreux tronçons risqueraient de se retrouver découpés de manière imprévisible. La majorité des objets de Geotrek sont associés à des tronçons du linéaire, et les changements dans le linéaire affectent directement leur géométrie. C'est pourquoi nous devons mettre en place une méthodologie qui permette de minimiser les effets de bord de l'intégration d'un nouveau linéaire.

6.3 Méthode

Pour cela, je me suis attelé à ce que j'appellerai la réconciliation de deux linéaires : la fusion de deux linéaires selon des critères prédéfinis, tout en respectant les principes topologiques.

Le déroulement de cette partie du stage a été moins fluide que les autres, car dépendant en grande partie de nos partenaires. En effet, même si la Commission Départementale des Espaces, Sites et Itinéraires (CDESI) de Lozère a été créée en 2006 et celle du Gard peu de temps après, peu de communautés de communes ont réellement mis en place un RLESI depuis. Par exemple, la communauté de communes du Piémont Cévenol prévoit la livraison de son RLESI pour 2022, le pôle nature 4 saisons du Mont Aigoual a fini de mettre le sien en place en 2017, et le RLESI du pôle de pleine nature du Mont Lozère est en train d'être finalisé. Du côté de la Lozère, il y a pour l'instant peu d'engouement pour la création de RLESI, les collectivités locales ne se sont pas encore emparées de cet outil.

Globalement, les données ne sont pas prêtes pour une intégration, et les versions provisoires des réseaux qui nous ont été confiées sont amenées à évoluer. De la qualité de la topologie des RLESI dépendra notamment une partie de la démarche de traitement et d'intégration. C'est ce point qui nous a poussé à nous concentrer sur les données provenant du département du Gard. En effet, le linéaire est déjà rentré dans leur instance de Geotrek, et de ce fait respecte les mêmes règles topologiques que le celui du Parc national.

L'utilisation de Postgis et de requêtes SQL est tout indiquée pour cette tâche. L'objectif est d'identifier parmi les tronçons du référentiel du Parc national :

- ceux qui sont à supprimer, car en doublon avec des tronçons du RLESI ;
- ceux qui sont à conserver, car non présents dans le RLESI.

En plus de cela, les données attributaires doivent pouvoir être transférées des tronçons du RLESI aux tronçons correspondants de notre référentiel, en respectant le modèle de données de Geotrek.

Voici un exemple (fig. 38) de ce à quoi les données peuvent ressembler :

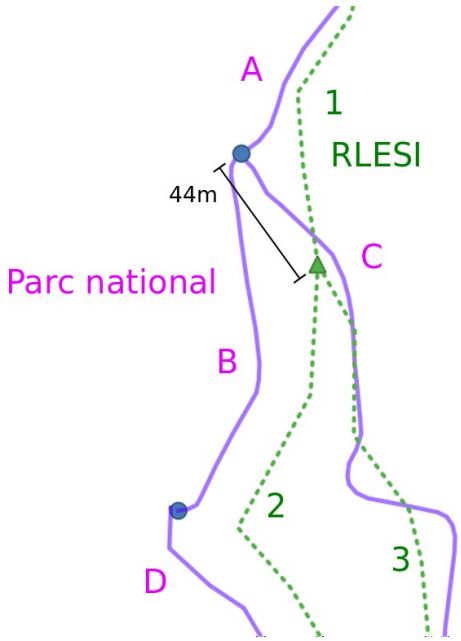


Figure 38: Comparaison du référentiel du Parc national et du RLESI

Les deux réseaux décrivent visiblement les mêmes sentiers. On voit bien la structure en fourche qui le laisse penser. Les deux points centraux des fourches sont donc censés représenter la même intersection. Puisqu'on souhaite prendre le RLESI comme référence, il faudrait donc que le tronçon *A* voie sa géométrie changée pour la géométrie du tronçon *1*. Pareil pour les tronçons *C* et *3*. En revanche, le tronçon *B* n'est pas équivalent à tout le tronçon *2*, mais seulement, disons, à 50 % de sa longueur totale. Il faut donc déterminer un point sur le tronçon *2* qui corresponde à l'intersection entre les tronçons *B* et *D* et le sépare en *2a* et *2b*. Le tronçon *B* verra donc sa géométrie calquée sur celle du tronçon *2a*, et le *D* sur celle de *2b* (en considérant qu'ils ont la même extrémité hors du cadre).

Ceci est plutôt simple à identifier de manière visuelle, par un·e humain·e. Et encore, si on affiche les données de l'IGN en-dessous de ces linéaires (fig. 39), un dilemme apparaît :

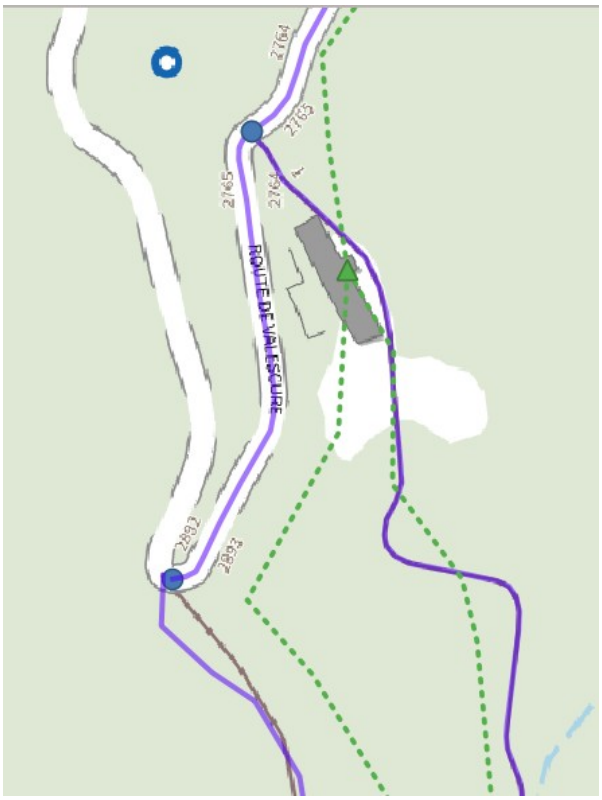


Figure 39: Comparaison du RLESI et du référentiel du Parc national par rapport aux données IGN actuelles

Le fait que notre référentiel soit identique à cette couche de l'IGN est normal, puisqu'il est fondé sur des données du même organisme. En revanche, le décalage avec les données du RLESI est très important… 44m entre les deux points censés représenter la même intersection. De plus, le tronçon *2* traverse le bâtiment, ce qui est bien sûr peu réaliste. Il est possible que ce soient les données de l'IGN qui soient non représentatives de la réalité, bien que la présence d'une route goudronnée (dont la géolocalisation est généralement très fiable) rende le fait improbable. La comparaison avec les données d'OpenStreetMap, et les images satellites de l'IGN semblent confirmer qu'ici, c'est le RLESI qui est imprécis, d'une quarantaine de mètres ! En plus du problème technique d'identification des tronçons en doublon et des tronçons uniques, s'ajoute un problème politique : que faire lorsque le réseau qu'on a décidé de considérer comme une référence est visiblement très incorrect ? Cela fera partie des questions à trancher lors de la phase effective de réconciliation.

Au niveau technique, un tel éloignement entre les tronçons rend impossible toute reconnaissance automatique de doublon. Les tolérances que j'ai essayé n'allaient pas au-delà de dix mètres, afin d'éviter un trop grand nombre de faux positifs. Dans notre exemple, tous les tronçons seraient considérés comme uniques, et la réconciliation aurait pour résultat un imbroglio de tronçons (rappelons que la segmentation dynamique impose que chaque intersection découpe les tronçons qui en font partie). On voit donc qu'un traitement manuel sera de toute façon indispensable à la réconciliation, et que les algorithmes créés ne pourront que prétendre fournir une aide à la décision.

Les traitements SQL ont donc pour but de construire une matrice constituée de chaque tronçon du linéaire qu'on souhaite modifier par rapport aux tronçons de l'autre linéaire. Dans notre cas il s'agit du référentiel du Parc national que nous confrontons au RLESI. Chaque relation (ou couple) de tronçons Parc/RLESI est accompagnée d'indicateurs d'aide à la décision.

id_parc	id_rlesi	cas	taux_similarité	geom_parc/rlesi	geom_rlesi/parc
25	14	doublon_total	1	geom	geom
25	897	doublon_partiel	0,8	geom	geom
26	*null*	unique_total	*null*	geom	*null*

Dans cet exemple, le tronçon numéro 25 du linéaire Parc est un total doublon du tronçon 14 du RLESI. Le champ *geom_parc/rlesi* contient la géométrie du tronçon Parc comparée à celle du tronçon RLESI : comme le tronçon Parc est un doublon sur toute sa longueur, *geom_parc/rlesi* = *geom_parc*. À l'inverse, le champ *geom_rlesi/parc* montre la géométrie du tronçon RLESI comparée au tronçon Parc. Le tronçon Parc numéro 26 n'est proche d'aucun tronçon du RLESI : il est donc totalement unique.

La construction de cette matrice se fait en plusieurs étapes :

- topologisation, si nécessaire, du RLESI ;
- comparaison de chaque tronçon Parc à chaque tronçon RLESI ;
- découpage et reconstitution des géométries « comparées » ;
- calcul des indicateurs.

Dans le cas où les données du RLESI auraient besoin d'être topologisées, la méthode suivie est celle décrite par Mathieu Leplâtre dans cet article : https://makina-corpus.com/blog/metier/2013/utiliser-les-topologies-postgis-pour-nettoyer-un-filaire-de-voirie. L'extension *topology*[66] de Postgis semble encore peu connue, car la majorité des tutoriels et explications à ce propos trouvées sur internet redirigent vers cet article, pourtant très vieux à l'échelle de l'informatique (2013). J'ai reproduit sa méthode à l'identique, et cela a donné des résultats satisfaisants sur des données lozériennes. Les données gardoises n'ayant pas besoin de topologisation, j'ai pu sauter à la deuxième étape.

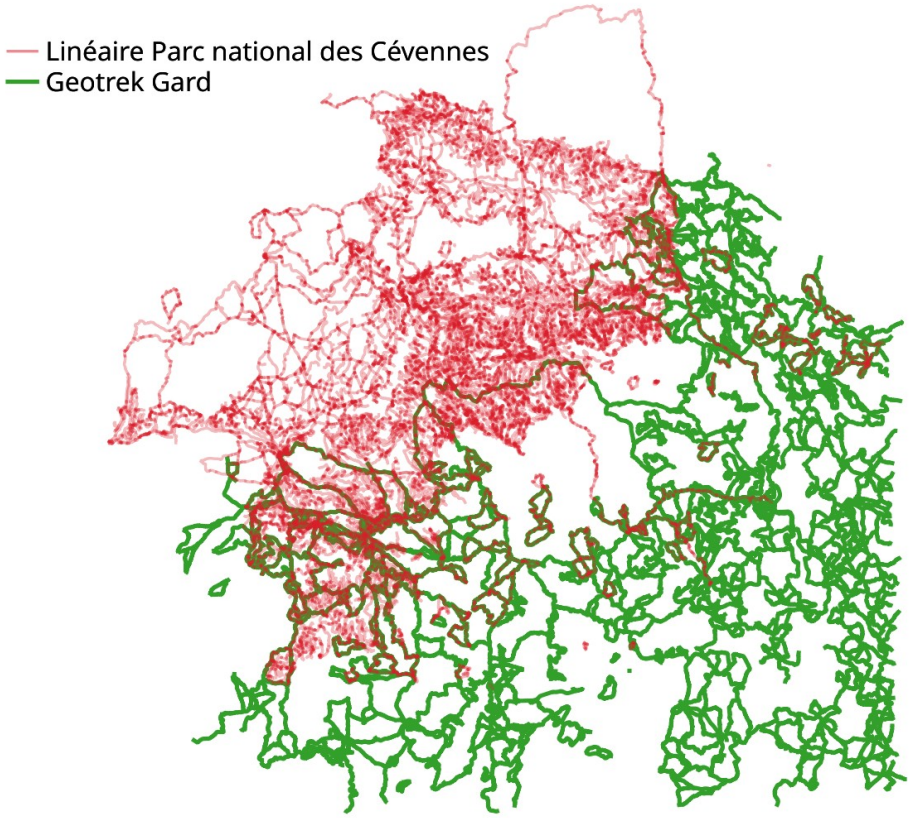


Figure 40: Visualisation des réseaux du Gard et du Parc national

J'ai essayé deux approches de comparaison des tronçons.

La première consiste à décomposer tous les tronçons du Parc national en l'ensemble des points qui les composent grâce à la fonction Postgis *ST_DumpPoints()*. Un simple calcul de distance avec *St_Dwithin()* permet ensuite d'identifier les points étant à moins de 10 mètres d'un tronçon RLESI. J'obtiens donc la liste des points du tronçon Parc x à moins de 10 mètres du tronçon RLESI y : à partir de là il est simple de recréer une géométrie en *LineString* ou *MultiLineString* (selon si les points sont consécutifs ou pas au sein du tronçon Parc originel) qui constitue notre géométrie « comparée ».

66 https://postgis.net/docs/Topology.html

Le principal problème de cette méthode est qu'elle est très sensible à la densité de points qui compose une ligne : plus la densité est élevée, plus elle sera précise, et moins elle est élevée, plus elle sera approximative. De plus, elle provoquait des incohérences visuelles lors de l'affichage des géométries de la matrice de décision sur QGIS, en laissant des trous dans les tronçons autour des intersections.

La deuxième méthode, qui a ma préférence, consiste à créer un tampon autour de chaque tronçon RLESI. On peut ensuite découper les tronçons Parc sur ces tampons avec *ST_Split()*. Il suffit alors d'identifier pour chaque tronçon Parc divisé quelle est la partie à l'intérieur du tampon et la partie à l'extérieur. En stockant l'identifiant du tampon, on obtient les géométries « comparées » des tronçons Parc par rapport aux tronçons RLESI. L'avantage que j'ai trouvé à cette méthode est qu'il y a une continuité entre les parties « uniques » et les parties « doublon » de chaque tronçon, puisqu'elles ont été divisées en un même endroit par un bord du tampon. Cette méthode ne laisse pas de trous entre deux points, au contraire de la première.

Quelle que soit la méthode utilisée, le calcul des indicateurs est ensuite sensiblement le même. Le rapport entre la longueur du tronçon Parc à moins de dix mètres du tronçon RLESI et la longueur totale du tronçon Parc nous indique à quel point le tronçon Parc est en doublon avec le tronçon RLESI. Ce rapport permet de classer chacun des couples/relations Parc/RLESI dans différentes catégories de « cas » : « 100 % doublon », « 100 % unique », « partiellement doublon », « partiellement unique ». D'autres indicateurs peuvent également être calculés, comme l'aire du polygone formé entre les géométries comparées du tronçon Parc x par rapport au tronçon RLESI y et du tronçon RLESI y par rapport au tronçon Parc x, etc. La mise en commun de tous ces indicateurs permet in fine de déterminer quels couples Parc/RLESI sont à considérer comme du bruit, des résidus du traitement qui n'ont pas de réelle pertinence.

id_parc	id_rlesi	cas	taux_similarité	geom_parc/rlesi	geom_rlesi/parc	bruit
25	50	doublon_total	1	geom	geom	false
25	12	doublon_partiel	0,01	geom	geom	true
25	935	doublon_partiel	0,005	geom	geom	true

Exemple : dans ce cas, on voit que la seule relation intéressante est celle entre les tronçons Parc 25 et RLESI 50. Si le taux de similarité de cette relation est de 1, et que le taux de similarité des autres est de 0,01 ou 0,005, cela indique à priori que les tronçons RLESI 12 et 935 ne sont proches du tronçon Parc 25 qu'à la marge, probablement à ses extrémités. C'est un « faux » doublon : du bruit.

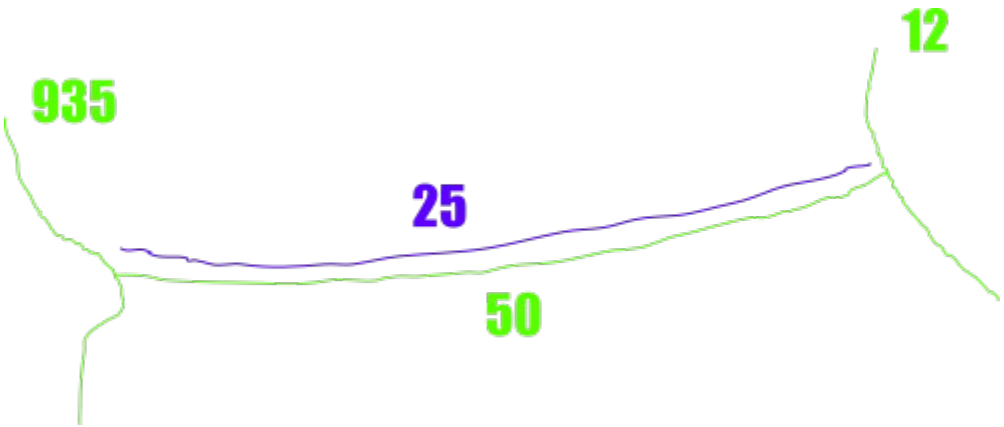


Figure 41: Exemple de faux positifs

Ci-dessus (fig. 41), une représentation visuelle de cet exemple, où on voit bien que le couple 25/50 est le seul qui vaille d'être pris en compte. La matrice de décision finale ne comprend donc que les relations déterminées comme n'étant pas du bruit.

Enfin, les géométries de la matrice sont visualisables dans un logiciel SIG comme QGIS. Ci-dessous (fig. 42 et 43) est affiché le RLESI du Gard, catégorisé selon le caractère unique ou non des bouts de tronçons qui le composent.

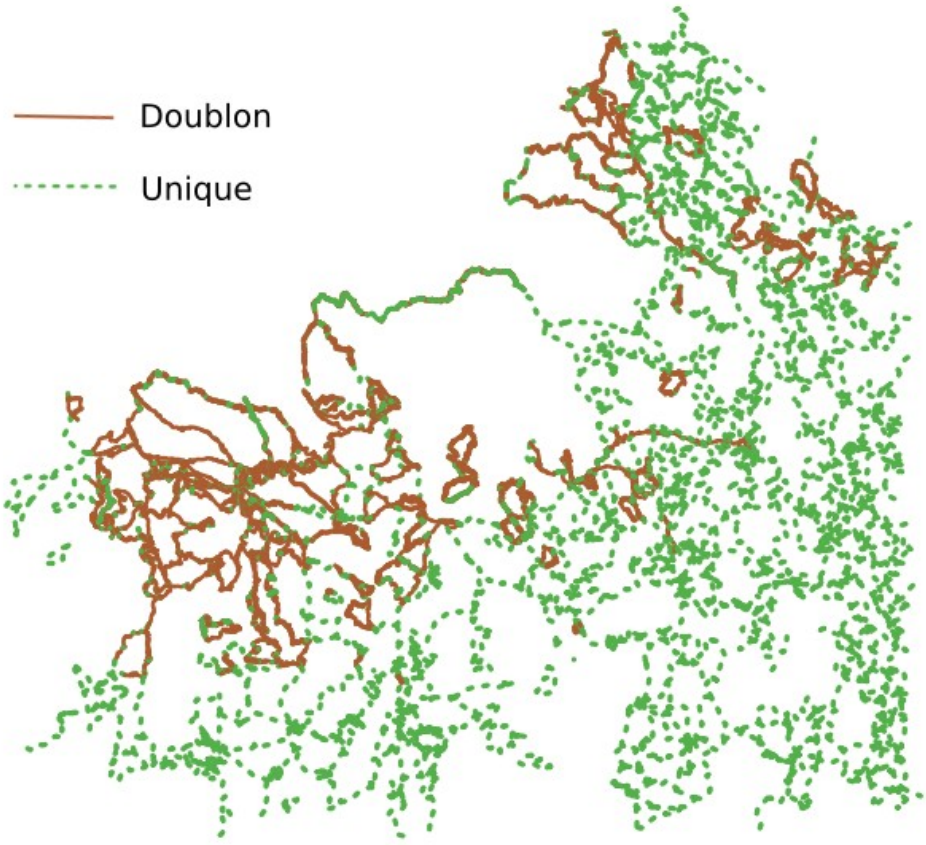


Figure 42: Visualisation de tous les tronçons selon leur unicité

Figure 43: Zoom sur la visualisation précédente

La matrice de décision est un outil précieux pour la suite de la méthodologie. Elle permet d'abord de visualiser de manière précise les différents statuts doublon ou unique, et donc de vérifier que le résultat des traitements est conforme à ce qu'on attendait. Cette vérification visuelle est d'autant plus utile que certains cas particuliers sont complexes à identifier et traiter automatiquement. Enfin, elle permet d'identifier les géométries sur lesquelles il faudra plaquer les tronçons considérés comme en doublon.

L'intervention humaine sur cette matrice permet de passer d'indicateurs de décision à un ensemble d'opérations à réaliser. Des scripts prennent ensuite le relais afin de réaliser ces opérations, qui sont de deux grands types :

- modification de géométrie ;
- ajout/suppression.

Pour les tronçons en doublon, dont il faut modifier la géométrie, nous pourrons soit utiliser une méthode de *snapping*, soit mettre à jour la géométrie avec un *UPDATE*. Le problème dans tous les cas est que les extrémités se déplacent avec, ce qui est le comportement attendu mais qui a comme conséquence de déconnecter d'autres tronçons de ces mêmes extrémités. En effet, si une intersection est partagée entre des tronçons uniques, dont on ne modifiera pas la géométrie, et des tronçons en doublon, dont on modifie la géométrie, elle disparaîtra car les deux extrémités qui la formaient ne seront plus en contact.

Un autre enjeu consiste donc à identifier ces tronçons ne faisant pas partie du RLESI (uniques), mais dont il faudra tout de même modifier la géométrie pour préserver la cohérence topologique avec les tronçons du RLESI.

Nous n'avons pas encore débuté l'implémentation de ces opérations faute de temps, mais celle-ci est prévue sous peu.

7. Bilan et perspectives

Je suis très satisfait du déroulement de mon stage, car j'en ai beaucoup retiré. Le fil conducteur de l'échange des données Geotrek m'a d'abord fait travailler sur le concept de schéma de données, du côté thématique autant que du côté technique, participer à un groupe de travail avec de nombreux partenaires venant de la France entière, et découvrir le logiciel Git. Le Geotrek-admin-aggregator m'a permis de contribuer à une application Python – même si je manquais encore d'expérience pour pouvoir la construire depuis zéro, j'ai réussi à m'approprier son fonctionnement et à améliorer ses fonctionnalités –, d'approfondir ma pratique de Git et des principes du versionnement, ainsi que d'échanger avec encore d'autres partenaires comme les conseils départementaux ou d'autres spécialistes de la géomatique comme Makina Corpus et OpenIG[67]. Enfin, la réconciliation des linéaires m'a fait plonger dans du SQL plus complexe que ce que j'avais fait jusqu'à présent, utiliser des facettes de Postgis que je ne connaissais pas encore, et échanger avec des partenaires supplémentaires comme Cartosud[68], une agence de géomatique. En somme, un stage très complet !

Le triple encadrement m'a beaucoup apporté, et les points hebdomadaires se sont vraiment enrichis des compétences et points de vue divers qui y étaient exposés. Les trois missions ont toutes des stades d'avancement différents : le schéma de données est en quelque sorte terminé, même si sa vie ne fait que commencer, l'agrégateur est un prototype qui ne demande qu'à être essayé, et la réconciliation des linéaires est encore un chantier en cours.

Afin de continuer d'avancer sur ce dernier point, le Parc national m'a proposé de signer un CDD de la fin de mon stage jusqu'au 30 décembre, et c'est avec plaisir que j'ai accepté. Cette offre valide le travail effectué depuis avril, ce qui ajoute à ma satisfaction ! La principale mission de ce contrat sera la réconciliation effective des linéaires du Parc national, du Gard et de la Lozère. Cependant, il sera aussi l'occasion de suivre les développements du schéma, qui doit encore être validé par le groupe de travail – aucun retour n'a encore été fait – et Etalab, et de l'agrégateur, Makina Corpus a prévu de se pencher sur notre application. La documentation doit être finalisée pour permettre sa réutilisation.

Enfin, un rendez-vous automnal marquera le début de mon contrat : les rencontres nationales de la communauté Geotrek les 14 et 15 octobre[69]. Ce sera une belle occasion d'échanger de vive voix avec des partenaires divers autour du travail réalisé ces cinq derniers mois !

67 https://www.openig.org/
68 http://cartosud.fr/
69 https://www.crige-paca.org/events/geotrek/

Bibliographie

Andrade Victor. (2020, septembre). *Perception des espaces naturels protégés sur le territoire du Pays des Écrins : étude de la pratique de la randonnée pédestre au prisme de l'outil Geotrek.* (Mémoire). https://geotrek.ecrins-parcnational.fr/ressources/stages/2020-09-rapport-stage-M1-Victor-ANDRADE-OTI-Pays-des-Ecrins.pdf

le Lann Lise. (2019, septembre). *Ingénierie & développement pour l'évolution d'une application open source sur les pratiques outdoor* (Mémoire). https://geotrek.ecrins-parcnational.fr/ressources/stages/2019-08-rapport-stage-M2-Lise-Le-Lann-PNE.pdf

Leplatre Mathieu. (2013, 18 octobre). *Utiliser les topologies PostGIS pour nettoyer un filaire de voirie*. Makina Corpus. https://makina-corpus.com/blog/metier/2013/utiliser-les-topologies-postgis-pour-nettoyer-un-filaire-de-voirie

Leplatre Mathieu. (2014, 17 février). *La segmentation dynamique*. Makina Corpus. https://makina-corpus.com/blog/metier/2014/la-segmentation-dynamique

Viadere Michaël. (2018, septembre). *Application de l'outil Geotrek pour la Communauté de Communes Pyrénées Vallées des Gaves* (Mémoire). https://geotrek.ecrins-parcnational.fr/ressources/stages/2018-09-rapport-stage-M2-Michael-VIADERE-CC-PVG.pdf

Glossaire

Ajv : validateur de JSON Schema.

BD TOPO : base de données vectorielle produite par l'IGN et couvrant l'ensemble du territoire français avec une précision métrique.

cron : utilitaire Unix permettant d'exécuter automatiqement des scripts et commandes en définissant le moment et la récurrence.

data.gouv.fr : plateforme de diffusion des données publiques de l'État français.

Etalab : département de la direction interministérielle du numérique (DINUM) qui coordonne la stratégie de l'État dans le domaine de la donnée.

Geotrek : suite logicielle composée de Geotrek-adlin et Geotrek-rando.

Geotrek-admin : application métier de gestion des données.

Geotrek-rando : application publique de valorisation touristique.

Git : logiciel de gestion de versions décentralisé le plus utilisé.

Makina Corpus : entreprise créatrice et mainteneuse de Geotrek.

GitHub : site de partage de code et de développement collaboratif, utilise le logiciel Git pour le versionnement.

Node.js : environnement d'exécution open source pour JavaScript.

Open source : désignation des logiciels dont le code source est public, réutilisable et redistribuable assez librement.

OpenStreetMap : base de données cartographique libre.

Postgis : extension de PostgreSQL permettant de manipuler des données spatiales.

PostgreSQL : système de gestion de base de données relationnelle et objet libre.

Réseau : ensemble de lignes entrelacées, ou ensemble de relations.

SI : Système d'Information.

Signalétique : ensemble des poteaux, lames, panneaux et balises permettant de se repérer sur les sentiers.

Abréviations

ALCOTRA : Alpes Latines COopération TRAnsfrontalière

CDESI : Commission Départementale des Espaces, Sites et Itinéraires

GNSS : *Global Navigation Satellite System*, seules les constellations GPS (États-Unis), GLONASS (Russie) et Galileo (Union Européenne) sont fonctionnelles

GR : sentier de Grande Randonnée, marque déposée par la Fédération Française de Randonnée Pédestre au même titre que GRP (sentiers de Grande Randonnée de Pays) et PR (sentiers de Promenade et Randonnée)

IGN : 'Institut national de l'information géographique et forestière

PDESI : Plan Départemental des Espaces, Sites et Itinéraires

PDIPR : Plan Départemental des Itinéraires de Promenade et de Randonnée

PITEM MITO : Plan Intégré ThEMatique Modèles Intégrés pour le Tourisme Outdoor

RLESI : Réseaux Locaux d'Espaces, Site et Itinéraires

WKT : Well-known text, format textuel de représentation d'objets géométriques vectoriels

Index des figures