Vers une modélisation orientée privacy des métadonnées d'un lac de données

Ali Hassan, Amine Mrabet, Patrice Darmon

Research & Innovation - Umanis
7, Rue Paul Vaillant Couturier, 92300 Levallois-Perret, France
{ahassan, amrabet, pdarmon} @umanis.com

Résumé. Afin d'améliorer la protection de données personnelles dans les lacs de données, nous proposons un méta-modèle qui comprend plusieurs aspects de privacy. Notre méta-modèle permet de décrire les contraintes nécessaires pour l'implémentation des procédures de protection de données personnelles (pseudonymisation/anonymisation). Il permet également de renforcer la conformité au RGPD dans les lacs de données en intégrant par exemple un journal de traitements sur les données personnelles et l'identification de finalité de chaque collection de données. Ce méta-modèle est présenté via un schéma conceptuel et implémenté via une base de données orientée graphe. La validation de notre proposition s'appuie sur les modélisations et discussions de plusieurs scénarios possibles de protection de données personnelles dans les lacs de données.

1 Introduction

Grâce à la révolution liée aux objets connectés, la taille des données collectées et l'analyse de ces données sont en croissance exponentielle avec différents enjeux : IA, statistique, publicité, etc. Les outils d'analyse des données classiques ne sont pas assez flexibles pour permettre aux utilisateurs finaux de réaliser les différents types d'analyses tels que les algorithmes d'apprentissage automatique, les statistiques, la visualisation de données, etc. pour l'exploration de données massives et hétérogènes. Plus précisément, l'entrepôt de données et l'OLAP sont des solutions couramment utilisées dans l'analyse stratégique des données. Les données sont extraites, transformées et chargées (processus ETL) selon des schémas prédéfinis (schéma multidimensionnel). Cependant, selon (Fang, 2015; Madera et Laurent, 2016), les entrepôts de données ne sont pas adaptés à l'analyse de données volumineuses parce que seules des exigences prédéfinies peuvent être satisfaites. Donc, les analystes ne peuvent pas réaliser des nouvelles analyses sophistiquées qui ne sont pas déjà définies.

Pour surmonter ce problème, le concept de lac de données a été proposé par (Dixon, 2010) et étendu par différents auteurs (Fang, 2015; Hai et al., 2016; Kafando et al., 2020; Sawadogo et Darmont, 2021; Megdiche et al., 2021; Ravat et Zhao, 2019b; Scholly et al., 2021; Walker et Alrehamy, 2015). La principale caractéristique d'un lac de données est le "schema-on-read" (Fang, 2015), donc les données ne sont traitées qu'au moment de l'utilisation. Cette caractéristique par contre entraîne et nécessite plus de coûts, d'efforts, des compétences et des outils bien plus techniques afin d'accéder et d'exploiter les données.

En effet, à cause de l'absence d'un schéma statique, l'interrogation et l'analyse des données dépendent d'un système de métadonnées efficace. C'est pourquoi la gestion des métadonnées joue un rôle essentiel dans les lacs de données afin d'éviter qu'ils ne se transforment en marécage de données "Data Swamp" inexploitables (Inmon, 2016).

Par ailleurs, la protection de la vie privée (le privacy) est une question essentielle dans tous les systèmes de stockages et d'analyse de données Hassan et al. (2021). Aujourd'hui, de plus en plus de données personnelles sont collectées, stockées et traitées dans les lacs de données ce qui pourrait entraîner une violation de la vie privée des personnes concernées.

Notre objectif dans ce papier est donc d'aider les administrateurs des lacs de données à respecter les exigences RGPD et à réduire les risques de violation de la vie privée, c'est pourquoi nous proposons un méta-modèle qui intègre le privacy. Un tel méta-modèle a pour enjeux :

— identifier les données personnelles ;
— décrire les différents contraintes techniques, de cohérence et métier qui concernent ces données. Cette description est nécessaire afin de faciliter l'implémentation des procédures de protection de données personnelles et de garantir la qualité des données ;
— représenter les traitements (registres) réalisés sur ces données (par exemple : anonymisation et pseudonymisation) ;
— prendre en compte les consentements, les finalités et les durées de conservation des données collectées.

L'article est structuré de la façon suivante. La section 2 présente la définition et les différentes architectures utilisées pour implémenter les lacs de données. La Section 3 présente l'état de l'art de méta modélisation des lacs de données. La Section 4 décrit notre proposition conceptuelle du méta-modèle orienté privacy des lacs des données. La Section 5 est consacrée à l'implémentation de notre proposition et présente comment notre modèle couvre plusieurs scénarios des traitements des données personnelles. Nous concluons dans la Section 6.

2 Préliminaires : lac de donnée

Plusieurs définitions de lacs de données ont été proposées par la communauté scientifique et industriel. Une des définitions la plus complète (à notre avis) est celle proposée dans (Ravat et Zhao, 2019a) : *"Un lac de données est une solution d'analyse de données volumineuses qui ingère des données brutes structurées de manière hétérogène provenant de diverses sources (locales ou externes à l'organisation) et stocke ces données brutes dans leur format natif, permet de traiter les données selon différentes besoins et fournit des accès aux données disponibles à différents utilisateurs (data scientistes, analystes de données, professionnels de la BI, etc.) pour l'analyse statistique, la Business Intelligence (BI), l'apprentissage automatique (ML), etc., et gouverne les données pour assurer la qualité, la sécurité et le cycle de vie des données"*. On peut déduire de cette définition que le lac de données ne se limite pas au stockage de données. Il supporte le traitement et l'interrogation des données à la demande. Le traitement à la demande comprend également l'intégration des données et l'indexation qui ne doivent être effectuées que si nécessaire au moment de l'accès aux données (Hai et al., 2021).

L'architecture fonctionnelle des lacs de données s'est rapidement développée d'une architecture mono-zone à une multi-zones. On peut distinguer dans la littérature deux versions de l'architecture multi-zones.

— **Une architecture basée sur les bassins (data pond)** (Inmon, 2016) : chaque bassin traite et stocke un type spécifique de données : brutes, à haute fréquence, provenant des logiciels, textuelles et archivées).
— **Une architecture basée sur les zones** : cette architecture affecte les données à une zone en fonction de leur degré de raffinement (Sawadogo et Darmont, 2021). Plusieurs variantes de cette architecture existent dans la littérature.
  — L'architecture proposée dans (LaPlante, 2016) comprend sept zones : une zone pour charger les données ((1) ingestion) et six zones pour stocker les données : (2) données brutes, (3) données raffinée, (4) données approuvées, (5) bac à sable de découverte pour les data scientistes, (6) zone de consommation pour les utilisateurs métier et (7) zone de gouvernance pour contrôler la sécurité et la qualité des données et gouverner les métadonnées.
  — L'architecture proposée dans (Ravat et Zhao, 2019a) contient quatre zones : (1) données brutes, (2) traitement, (3) accès et (4) gouvernance.
  — Dans l'architecture lambda, les auteurs de (John et Misra, 2017) séparent la zone de traitement des données en zones de traitement par lots (batch processing zone) et de traitement en temps réel (real-time processing zone).

3 État de l'art

Dans cette section, nous détaillons les métadonnées dédiées aux lacs de données. Ensuite, nous discutons comment le privacy est considéré dans les différents méta-modèles proposés.

3.1 Métadonnées dédiées aux lacs de données

Dans la littérature, nous pouvons distinguer deux méthodes principales de classement des métadonnées dédiées aux lacs de données :

— **Métadonnées fonctionnelles** : les métadonnées sont classées en trois catégories (Oram, 2015) :
  — *Les métadonnées techniques* concernent la structure et le format des données.
  — *Les métadonnées opérationnelles* concernent le traitement des données : l'historisation des traitements et les conséquences et les descriptions de traitements.
  — *Les métadonnées métier* concernent les objectifs et les contraintes métier.
— **Métadonnées structurelles** : les métadonnées sont classées en trois groupes (Sawadogo et al., 2019; Ravat et Zhao, 2019a) :
  — *Intra-métadonnées* : les informations qui concernent chaque dataset individuellement. Un dataset peut être une base de données relationnelle ou Nosql ou bien un fichier physique. Les intra-métadonnées peuvent être organisées en plusieurs sous-catégories : les caractéristiques des données, les métadonnées sémantiques, les métadonnées schématiques, les métadonnées de version et de représentation, les métadonnées de navigation (location), les métadonnées de traçabilité et de cycle de vie (linkage), les métadonnées d'accès (historiques d'accès des utilisateurs), les métadonnées de qualité (cohérence et complétude des données) et les métadonnées de sécurité (la sensibilité des données et le niveau d'accès).

— *Inter-métadonnées* : les informations qui concernent les relations entre les datasets. Elles sont classées en inclusions et chevauchement partiel de datasets, provenance, regroupement logique et similarité de contenu.
— *Métadonnées globales* : les informations qui concernent l'ensemble du lac de données (Sawadogo et al., 2019). Elles ne sont pas associées à un dataset spécifique. Elles comprennent trois sous-catégories : ressources sémantiques (des bases de connaissances telles que des ontologies, des taxonomies, etc.), les index et les logs (les interactions des utilisateurs avec le lac de données).

3.2 Méta-modèle de lac de données & Privacy

Des nombreux travaux dans la littérature ont proposé des modèles de métadonnées pour les lacs de données. Beaucoup d'entre eux se concentrent sur un sujet spécifique et ne prennent en charge que des cas d'utilisation limités (Kafando et al., 2020; Sawadogo et al., 2019; Spiekermann et al., 2018; Theodorou et al., 2019).

En ce qui concerne les modèles génériques proposés dans la littérature (Diamantini et al., 2018; Sawadogo et Darmont, 2021; Megdiche et al., 2021; Quix et al., 2016; Ravat et Zhao, 2019b; Scholly et al., 2021), grâce à la généricité de ces propositions, certains aspects de privacy peuvent être considérés. Par exemple, la modélisation de processus de protection de données personnelles (pseudonymisation et/ou anonymisation) est possible comme un processus générique. Par contre, d'autres aspects nécessaires pour le privacy ne sont pas pris en compte comme la durée de conservation de données ou bien la finalité qui justifie la collection de données personnelles. La proposition de (Megdiche et al., 2021) prend en considération le privacy où les auteurs associent à chaque dataset un niveau de sensibilité, ce qui permet d'identifier les droits d'accès afin de respecter les exigences de sécurité. Mais cette considération reste très limitée et il ne permet pas de couvrir tous les aspects de privacy.

A notre connaissance, seuls les travaux de (Walker et Alrehamy, 2015) se concentrent sur les données personnelles. L'objectif principal de ces travaux est de construire un lac de données personnelles qui donne aux utilisateurs le contrôle de la gestion et du partage de leurs données avec d'autres personnes et/ou organisations. Bien que ce travail repose principalement sur la gestion des données personnelles, il ne fait aucune proposition pour représenter et gérer les opérations de traitement sur ces données. En outre, les métadonnées proposées ne contiennent que les propriétés suivantes : (1) source, (2) contexte pour indiquer les catégories, les classes et les schémas de datasets, (3) type, (4) assertion pour indiquer si les données viennent des sources structurées/semi-structurées ou bien si elles nécessitent plus d'analyse pour les données non structurées, (5) abstraction personnalisée (facultative) qui est un ensemble de paires de clé-valeur afin de représenter les facettes des données brutes. Par conséquence, ces métadonnées proposées ne permettent pas de couvrir les différents aspects du privacy et des RGPD.

4 Méta-modèle conceptuel orienté privacy

Afin de surmonter le manque de considération du privacy rencontré dans les différents travaux de gestion de métadonnées dans les lacs de données, nous proposons dans cette section un méta-modèle conceptuel pour les lacs de données (figure 1). Notre proposition renforce la gestion des données personnelles et facilite le contrôle de la conformité au RGPD dans les lacs

des données. Ce modèle est inspiré des différents travaux de Ravat & Zhao : (Ravat et Zhao, 2019a,b; Zhao, 2021; Megdiche et al., 2021).

Afin d'expliquer notre modèle conceptuel, nous classifions les classes UML qui représentent les métadonnées dans cinq catégories : (1) structure, (2) relation, (3) traitement, (4) intégration/accès, (5) conformité au RGPD. Même si la cinquième catégorie est spécifique pour le privacy, ce dernier est également présent dans les autres quatre catégories. Dans la suite, nous détaillerons ces différentes catégories.

4.1 Structure

Cette catégorie englobe les classes coloriées en jaune dans la figure 1. Elle est utilisée pour décrire les données dans les différentes zones des lacs de données. Elle regroupe les datasets qui peuvent être structurés, semi-structurés ou non-structurés. Chaque dataset possède plusieurs propriétés qui concernent le nom du dataset, le type du dataset, une description, la date de création et des informations de connexion. Les datasets structurés et semi-structurés incluent une ou plusieurs entités qui à leur tour comprennent plusieurs attributs. Dans le contexte relationnel, un dataset correspond à une base de données, une entité correspond à une table et les attributs correspondent aux colonnes de la table. Un attribut peut être associé à un ou plusieurs contraintes. La classe "Contraint" permet de décrire les contraintes techniques et métiers. La description de ces contraintes est nécessaire et c'est un prérequis pour réaliser les traitements de données y compris l'anonymisation et la pseudonymisation. Les contraintes peuvent être de plusieurs types :

- **contraintes sur le format** : elles sont représentées par la propriété "Format" en utilisant les expressions régulières.
- **checkSum** : il est utilisé pour définir des formules que les valeurs de l'attribut doivent respecter.
- **cohérence intra-attribut** : elle impose qu'il faille maintenir la cohérence entre les valeurs du même attribut avant et après les traitements, par exemple, les valeurs identiques devraient être remplacées par la même valeur après la réalisation d'un traitement
- **non modifiable**
- etc.

Un autre aspect de privacy décrit dans cette catégorie "structure" est l'évaluation de la sensibilité des données. Cette sensibilité est évaluée via les deux propriétés "sensitivity_level" et "sensitivity_type" qui sont associées à chaque attribut. La propriété "sensitivity_level" définit le niveau de sensibilité d'un attribut personnel afin d'identifier le traitement nécessaire à mettre en place pour protéger les données. La propriété "sensitivity_type" détermine le type des données : identifiant (ID), quasi-identifiant (QID), attribut sensible (AS) ou attribut non personnelles (ANS). Il est nécessaire de distinguer les ID/QID des AS parce que le modèle de la protection de la vie privée à appliquer est différent selon le type de données personnelles : suppression pour les ID, k-anonymat pour les QID et L-diversité pour les AS. La détermination de niveau et de type de sensibilité des données nécessite une étape de cartographie de données du lac de données.

Ces deux propriétés de la classe "Attribute" permettent de calculer le niveau de sensibilité et le type de sensibilité au niveau entités et au niveau datasets (pour les datasets structurés et semi-structurés). Par contre, pour les datasets non-structurés, les deux propriétés sont associées directement au dataset.

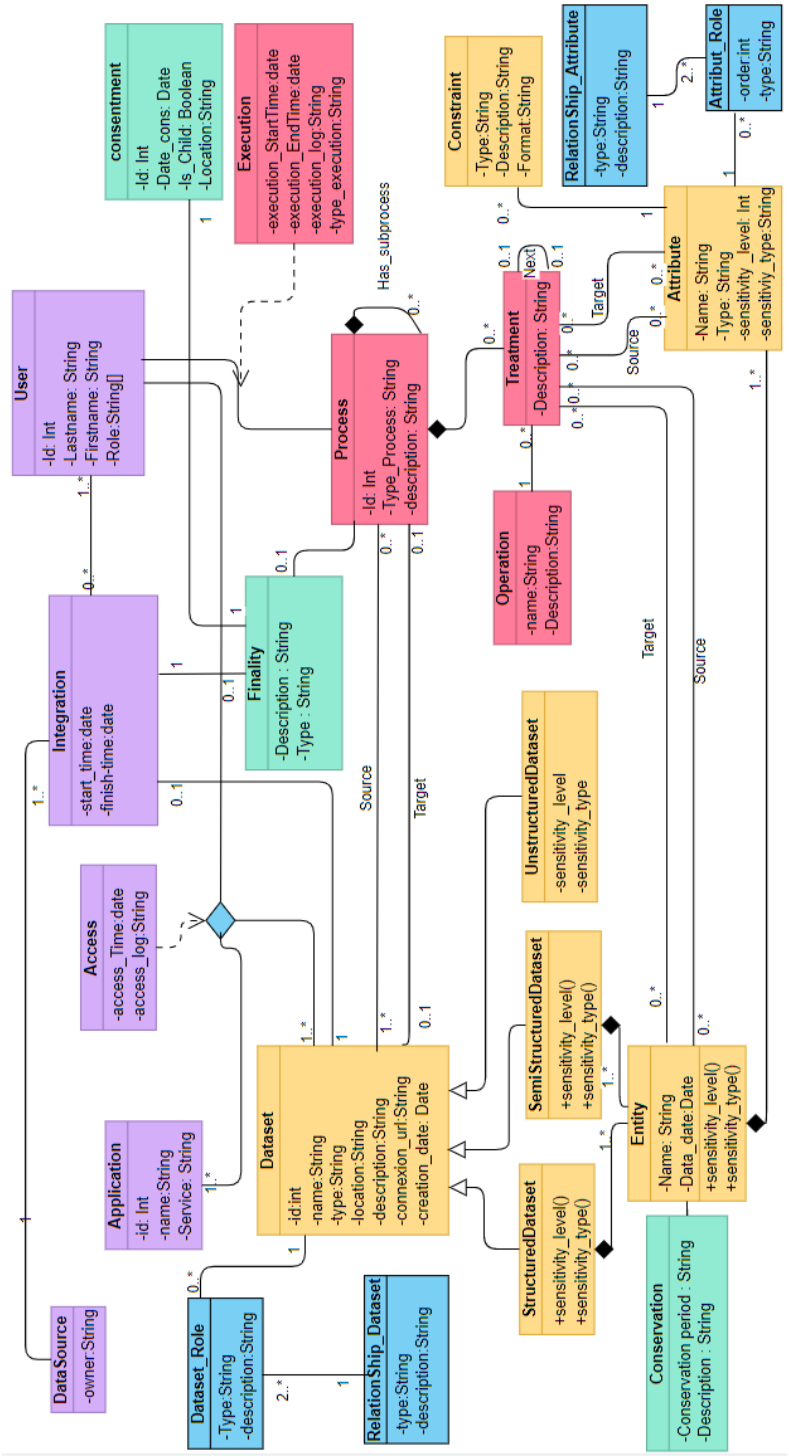

FIG. 1 – *Méta-modèle conceptuel du lac de données orienté privacy.*

4.2 Relation

Cette catégorie englobe les classes coloriées en bleu dans la figure 1. Elle est utilisée pour décrire les relations entre les structures de données dans les différentes zones des lacs de données. Elle représente les relations à deux niveaux : (1) entre les datasets et (2) entre les attributs.

1. **Entre les datasets** : les classes "Dataset_Role" et "RelationShip_Dataset" permettent de représenter les différentes relations inter-datasets définies dans la littérature (Ravat et Zhao, 2019a) : inclusion et chevauchement partiel de datasets, provenance, regroupement logique et similarité de contenu. La classe "RelationShip_Dataset" définit la relation entre les datasets et la classe "Dataset_Role" précise le rôle de chaque dataset dans cette relation. Par exemple, pour une relation de type "inclusion", la classe "Dataset_Role" détermine quel dataset est le global et quel dataset est inclus.
2. **Entre les attributs** : les classes "Attribut_Role" et "RelationShip_Attribute" déterminent les contraintes de cohérence inter-attributs. Ces contraintes imposent qu'il faille maintenir la cohérence entre les valeurs des attributs différents avant et après les traitements. Ces attributs peuvent être dans la même entité ou dans des entités différentes ou même dans des datasets différents. Nous définissons les types de cohérence inter-attributs les plus utilisés dans les bases de données :
   — **cohérence de type "exacte"** : les valeurs identiques des différents attributs devraient être remplacées par la même valeur après la réalisation d'un traitement. Par exemple, la cohérence des identifiants (ids) des clients entre les données commerciales et les données de comptabilité est nécessaire même après la réalisation d'une anonymisation/pseudonymisation des données.
   — **cohérence de type "contient"** : les valeurs des certains attributs contiennent les valeurs d'autres attributs. Par exemple, le numéro de sécurité sociale "NSS" contient le sexe, l'année et le mois de naissance. Donc, la cohérence entre ces attributs est nécessaire. La classe "Attribut_Role" peut donner un rôle comme "Principale" à l'attribut "NSS" et donner un rôle comme "Inclus" au sexe, à l'année et au mois de naissance. Ces quatre rôles peuvent s'associer via la classe "RelationShip_Attribute" pour exprimer cette relation (contrainte de cohérence) entre ces quatre attributs.
   — **cohérence de type "concaténation"** : elle est un cas particulier de la cohérence de type "contient". Par exemple, l'attribut "titulaire" d'un compte bancaire concatène les valeurs des attributs "civilité", "prénom", "nom". La propriété "order" de la classe "Attribut_Role" permet de définir l'ordre de concaténation des attributs.
   — **cohérence de type "hiérarchique"** : cette contrainte est utilisée pour exprimer une cohérence hiérarchique entre les attributs. Cette cohérence n'est pas visible directement dans les valeurs. Par exemple, la cohérence entre le nom de la région et le nom du département est basée sur la division géographique du pays.
   — **cohérence de type "calculé"** : certains attributs sont calculées à partir d'autres attributs. Par exemple, l'âge est calculé à partie de la date de naissance.

4.3 Traitement

Cette catégorie englobe les classes coloriées en rouge dans la figure 1. Elle est utilisée pour décrire les différents traitements des données réalisés dans les zones de traitements en temps réel (real-time processing zone) et par lots (batch processing zone). Cette catégorie représente

tout type de processus (anonymisation, pseudonymisation, nettoyage et prétraitement pour IA, etc.) appliqué sur les datasets. Chaque processus peut posséder un ou plusieurs sous processus. La propriété "description" du processus explique généralement pourquoi un processus est créé, quel est le contexte et pour quel objectif. Chaque processus a un dataset d'entrée (Source) et un dataset de sortie (Target). Les processus qui modifient les données d'un dataset sont associés à ce dataset comme "Source" et "Target" à la fois.

L'utilisateur, qui déclenche le processus, est identifié via l'association entre la classe "Process" et la classe "User". La classe association "Execution" comprend la date, le log et le type d'exécution du processus. Ainsi, l'ensemble d'objets de cette classe "Execution" présente la journalisation de tous les traitements des données appliqués dans le lac de données.

Un processus appliqué sur un dataset est composé d'un ou plusieurs traitements ordonnés sur les tables ou les attributs de ce dataset. L'ordre de l'application des traitements est défini via l'association "Next" entre les classes "Treatment". Chaque traitement réalise une opération qui est présentée par la classe "Operation". Une liste prédéfinie des opérations est décrite dans (Megdiche et al., 2021) : filtrage, formatage, agrégation, calcul, consolidation, fusion, jointure. Selon les auteurs, la protection des données (chiffrement, anonymisation, pseudonymisation) est réalisée par des opérations de même type (consolidation). Ce dernier comprend également la conversion et la correction des données ce qui ne permet pas de décrire de manière détaillée et de distinguer les traitements réalisés dans le lac de données. Nous ne nous limitons pas à cette liste. Nous proposons de l'enrichir avec des opérations plus précises et spécifiques pour les différents types de traitements surtout en ce qui concerne les données personnelles pour pouvoir vérifier que le processus respecte les contraintes de cohérence. Par exemple, la cohérence "exacte" impose d'utiliser la même opération pour traiter les attributs concernés. Donc, une description détaillée des opérations appliquées sur ces attributs est nécessaire afin de vérifier cette cohérence.

Parmi les opérations que nous proposons d'ajouter à cette liste :

— opérations d'anonymisation : généralisation, génération aléatoire, permutation, ... ;
— opérations de pseudonymisation : hachage, chiffrement ;
— opérations des modèles de protection de la vie privée : K-anonymat (garantie que les valeurs de QID existent au moins k fois dans un ensemble des données) et L-diversité (qui garantie l'existence d'au moins 'L' valeurs bien représentées pour chaque AS.

4.4 Intégration/Accès

Cette catégorie englobe les classes coloriées en violet dans la figure 1. Cette catégorie décrit les échanges (entrées/sorties) entre le lac de données et son environnement. Plus précisément, elle modélise les intégrations et les accès des données dans la zone de données brute/ingestion et la zone d'accès/consommation respectivement.

Grâce aux propriétés "start_time_date" et "finish_time_date" de la classe "Integration", l'ensemble d'objets de cette classe construisent une historisation de toutes les intégrations des données dans le lac de données. De la même manière, grâce à la propriété "access_Time" de la classe association "Access", l'ensemble d'objets de cette classe représente une journalisation des connexions effectuées par des utilisateurs ou par des applications. La propriété "Role" permet d'identifier plusieurs catégories des utilisateurs et leur attribuer des droits selon leur rôle dans la gestion de lac de données (analyste, Délégué à la protection des données "DPO", Chief Data Officer "CDO", etc.).

4.5 Conformité au RGPD

Cette catégorie englobe les classes coloriées en vert dans la figure 1. Elle complète les différents aspects de privacy déjà considérés dans les autres catégories en intégrant des concepts répondant aux exigences du RGPD. Elle concerne les différentes zones du lac de données. Cette catégorie contient les classes :

- **"Finality"** : afin de respecter l'article 5.1.b du RGPD : *"Les données à caractère personnel doivent être collectées pour des finalités déterminées, explicites et légitimes, et ne pas être traitées ultérieurement d'une manière incompatible avec ces finalités ..."*, notre modèle permet de décrire les finalités de chaque intégration (collection) et traitement de données personnelles dans le lac de données.
- **"Conservation"** : selon l'article 5.1.e du RGPD : *"Les données à caractère personnel doivent être conservées sous une forme permettant l'identification des personnes concernées pendant une durée n'excédant pas celle nécessaire au regard des finalités pour lesquelles elles sont traitées..."*, chaque responsable de traitement est dans l'obligation de déterminer une durée de conservation des données personnelles. Cette durée doit être cohérente et justifiée au regard de l'objectif de leur traitement. C'est pourquoi notre modèle permet de préciser pour chaque entité une durée de conservation, en sachant que les durées de conservation maximale sont définies pour certaines finalités par des dispositions légales et réglementaires.
- **"Consentment"** : les articles 7 et 8 du RGPD sur les "Conditions applicables au consentement" et sur les "Conditions applicables au consentement des enfants en ce qui concerne les services de la société de l'information" indiquent que le responsable du traitement doit être en mesure de démontrer que chaque personne concernée a donné son consentement au traitement de ses données à caractère personnel. Notre méta-modèle respecte ces exigences en intégrant le concept de consentement associé à la finalité. Ainsi, les liens entre les intégrations et les traitements réalisés dans le lac de données d'une part et les consentements d'autre part sont déterminés via les finalités.

5 Implémentation et discussion

Afin de montrer la faisabilité de notre proposition, nous avons mis en œuvre notre modèle de métadonnées en utilisant une base de données orientée graphes Neo4j. Cette implémentation concerne un lac de données dans le secteur d'assurance qui contiennent plusieurs données personnelles : nom, prénom, email, date de naissance, sexe, numéro de sécurité sociale, ville, etc. Les data scientistes se basent sur ce lac de données afin de réaliser des analyses du comportement des clients. Cet usage est générique et pourrait exister dans d'autres secteurs. Aujourd'hui, les administrateurs du lac de données sont dans l'obligation de protéger la vie privée des clients ainsi que respecter le RGPD et les contraintes de la CNIL. Dans la suite de cette section, nous discutons comment notre méta-modèle permet aux administrateurs de :

- fournir un registre qui décrit le traitement de données personnelles (voir 5.1)
- vérifier si les traitements respectent les niveaux de sensibilité des données (voir 5.1)
- respecter la durée de conservation des données personnelles (voir 5.2)
- vérifier si l'anonymisation respecte les contraintes de cohérence inter-attributs et intra-attribut afin de garder l'utilité des données après l'anonymisation (voir 5.2).

5.1 Registre des activités de traitement dans un lac de données

Notre méta-modèle définit toutes les informations nécessaires pour construire le registre des activités de traitement de données personnelles imposé dans l'article 30 du RGPD. L'implémentation de notre méta-modèle permet de réaliser ce registre des activités. La préparation des données pour l'analyse du comportement dans notre lac de données consiste à pseudonymiser les données. La figure 2 montre le sous-graphe de notre implémentation concernant cette activité de pseudonymisation. Cette dernière prend comme entrée le dataset "base client" et elle donne en sortie un dataset pseudonymisé "base client (AI)". Ce sous-graphe comporte tous les éléments de la fiche de registre définis dans l'article 30 du RGPD :

- **le nom et les coordonnées du responsable du traitement** : ces informations sont décrites via les nœuds qui représentent la classe "User" du méta-modèle. Dans la figure 2, "Amine" est le responsable qui lance le traitement de données.
- **les finalités du traitement** : ces informations sont directement associées au traitement via les nœuds qui représentent la classe "Finality". La finalité de traitement dans la figure 2 est "Analyse du comportement".
- **les catégories de personnes concernées** : ces informations sont présentées via les nœuds des classes "Dataset" et "Entity" associés au processus et à ses traitements. Dans la figure 2, le traitement concerne les données des clients.
- **les catégories de données personnelles** : ces informations sont présentées via les nœuds de la classe "Attribute". Les catégories de données personnelles sont identifiées via la propriété "sensitivity_type". Dans la figure 2, tous les attributs de l'entité "client" sont des données personnelles.
- **les catégories de destinataires** : ces informations sont décrites via les nœuds des classes "User" et "Application" qui accèdent à la sortie du processus. Dans la figure 2, "Ali" accède au résultat de la pseudonymisation (dataset "base client (AI)").
- **les transferts de données à caractère personnel vers un pays tiers** : ces informations sont décrites via la propriété "location" de dataset résultant du traitement.
- **la durée de conservation** : cette information est directement associée aux entités du résultant de traitement via les nœuds qui représentent la classe "Conservation". Dans la figure 2, une durée de conservation de cinq ans est déterminée pour les données pseudonymisées des clients.
- **une description générale des mesures de sécurité technique et organisationnelle** : une telle description peut être intégrée dans les propriétés "description" de toutes les classes qui concernent les activités de traitement :"Process", "Treatment" et "Operation". De plus, la propriété "sensitivity_level" des classes "Attribute" et "UnstructuredDataset" et les fonctions "sensitivity_level()" des classes "Entity" et "Dataset" peuvent jouer un rôle dans cette description des mesures de sécurité.

Par ailleurs, ce sous-graphe aide les administrateurs à vérifier si les traitements appliqués sur les attributs correspondent aux niveaux de sensibilité. Par exemple, le processus de la préparation des données pour l'IA (pseudonymisation) protège les attributs qui ont un niveau de sensibilité élevé (nom, prénom, email, NSS, code et date_naissance). Le sous-graphe de la figure 2 montre que ce processus prend en entrée les identifiants directs des clients (nom, prénom, email et NSS) qui ne sont pas dans la sortie. Autrement dit, ce processus applique un traitement de suppression sur ces identifiants. Il montre également qu'il y a un chiffrement sur l'attribut code (identifiant interne chez l'assureur) et une généralisation sur l'attribut

date_naissance (quasi-identifiant). Aucun traitement n'est présenté sur les quasi-identifiants ayant un bas niveau de sensibilité (sexe, ville et région) par contre une vérification de la qualité de pseudonymisation est réalisée à la fin de processus de préparation des données.

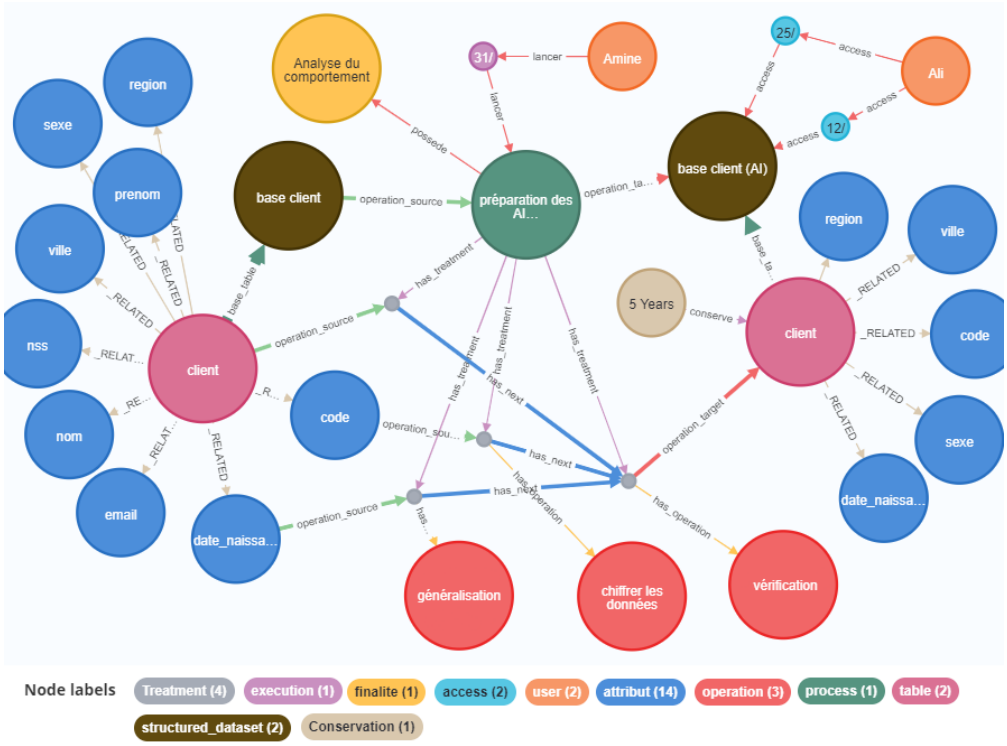

FIG. 2 – *sous-graphe de préparation des données pour l'IA (pseudonymisation)*

5.2 Anonymisation et contraintes de cohérence

En se basant sur la classe "Conservation" associée à la classe "Entity" ainsi que la propriété "Data_date" (date de données) de la classe "Entity", l'administrateur du lac de données est capable d'identifier les entités qui dépassent la durée de conservation définie. En outre, l'implémentation de notre modèle permet aux administrateurs de vérifier s'il y a un processus d'anonymisation ou suppression qui a été exécuté sur ces entités via la classe association "Execution". Autrement dit, notre proposition permet la vérification du respect de la durée de conservation. Cette vérification est nécessaire pour la conformité au RGPD.

L'objectif de l'anonymisation est de protéger les données personnelles en conservant l'utilité des données, c'est pour cette raison que les contraintes de cohérence intra-attribut et inter-attributs sont définies. Donc notre proposition permet de vérifier si les processus d'anonymisation respectent ces contraintes. Comme le présente le sous-graphe de la figure 3, ce respect des contraintes est présenté par une cohérence entre les traitements du processus d'anonymisation et les contraintes intra-attribut et inter-attributs.

La figure 3 présente les exécutions du processus d'anonymisation des données personnelles des clients. Elle montre trois traitements d'anonymisation :

— "tr4" : généralisation des dates de naissance ;
— "tr6" : sélection aléatoire des valeurs parmi une liste prédéfinie pour l'attribut "sexe" ;
— "tr2" : génération aléatoire de numéro de sécurité sociale "NSS".

Une contrainte inter-attributs de type "contient" est définie entre NSS (attribut principal) et le sexe et la date de naissance (attributs inclus) où les arcs entre les nœuds "Attribut" et les nœuds "RelationShip_Attribute" représentent les rôles des attributs dans les relations. La cohérence entre les trois traitements et cette contrainte est présentée via les arcs "has_next" entre ces traitements. Plus précisément, le traitement "tr2" prend en entrée les résultats des traitements "tr4" et "tr6" pour les utiliser dans l'anonymisation de "NSS".

Les contraintes intra-attribut sont définies via les arcs "operation_source" entre les nœuds du type "treatment" et les nœuds du type "attribute". Comme le présente la figure 3, les traitements "tr2" et "tr4" prennent respectivement en entrée (source) les valeurs de NSS et la date de naissance afin de maintenir la cohérence entre les valeurs avant et après les traitements.

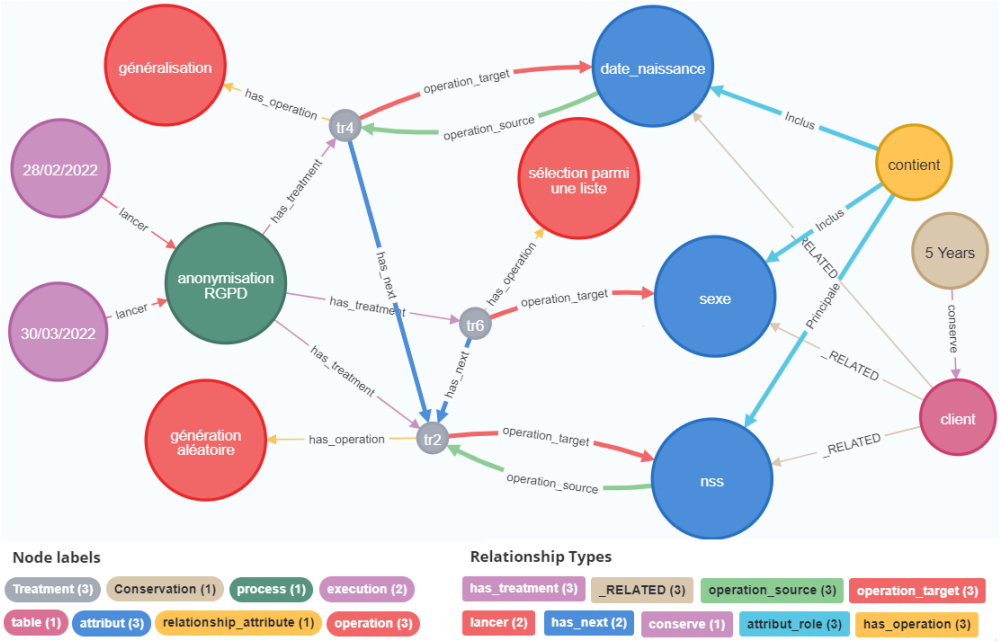

FIG. 3 – *sous-graphe d'anonymisation*

6 Conclusion

Nous avons proposé dans ce papier un schéma conceptuel pour la méta-modélisation des lacs de données. Cette proposition permet de décrire les structures de données et leurs relations, les contraintes de cohérence intra/inter attributs, les différents traitements réalisés dans le lac

et les échanges entre le lac de données et son environnement. Afin d'améliorer la conformité au RGPD, notre proposition intègre les aspects du privacy : la finalité et le consentement aux traitements des données personnelles ainsi que la durée de conservation. L'implémentation de ce modèle qui est faite via une base de données Neo4j permet de (1) fournir un registre de traitements, (2) vérifier la correspondance entre les traitements et les niveaux de sensibilité des données, (3) respecter la durée de conservation et (4) valider la cohérence entre les processus d'anonymisation et les contraintes intra/inter attributs.

Dans la perspective de ce travail, nous avons commencé à développer un système de gestion de métadonnées. Ce système se base sur des APIs qui permettent de construire et modifier la base Neo4j qui représente nos métadonnées. Un tel système devrait être lié à des systèmes d'extraction de structures de données et de détection des relations entre ces structures. Dans la même perspective, nous envisageons d'intégrer nos anciens travaux sur la détection de données personnelles (Mrabet et al., 2019; Mrabet et al., 2019) afin d'automatiser la définition de niveau et de type de sensibilité des données.

Références

Diamantini, C., P. L. Giudice, L. Musarella, D. Potena, E. Storti, et D. Ursino (2018). A new metadata model to uniformly handle heterogeneous data lake sources. In *New Trends in Databases and Information Systems*, pp. 165–177.

Dixon, J. (2010). Pentaho, hadoop, and data lakes. `https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/`. Accessed : 2022-04.

Fang, H. (2015). Managing data lakes in big data era : What's a data lake and why has it became popular in data management ecosystem. In *CYBER 2015*, pp. 820–824.

Hai, R., S. Geisler, et C. Quix (2016). Constance : An intelligent data lake system. In *Intl. Conf. on Management of Data*, SIGMOD '16, pp. 2097—2100.

Hai, R., C. Quix, et M. Jarke (2021). Data lake concept and systems : a survey. *ArXiv abs/2106.09592*.

Hassan, A., A. Mrabet, et P. Darmon (2021). Implémentation des plugins logstash de généralisation et de chiffrement pour l'anonymisation et la pseudonymisation. In *EGC 2021*.

Inmon, B. (2016). *Data Lake Architecture : Designing the Data Lake and Avoiding the Garbage Dump* (1st ed.). Denville, NJ, USA : Technics Publications, LLC.

John, T. et P. Misra (2017). *Data Lake for Enterprises : Lambda Architecture for building enterprise data systems*. Packt Publishing.

Kafando, R., R. Decoupes, L. Sautot, et M. Teisseire (2020). Spatial data lake for smart cities : From design to implementation. *AGILE : GIScience Series 1*, 8.

LaPlante, A. (2016). *Architecting data lakes*. O'Reilly Media.

Madera, C. et A. Laurent (2016). The next information architecture evolution : The data lake wave. In *Intl. Conf. on Management of Digital EcoSystems*, pp. 174—180.

Megdiche, I., F. Ravat, et Y. Zhao (2021). Metadata management on data processing in data lakes. In *Intl. Conf. on Current Trends in Theory and Practice of Informatics*, pp. 553–562.

Mrabet, A., M. Bentounsi, et P. Darmon (2019). Secp2i a secure multi-party discovery of personally identifiable information (pii) in structured and semi-structured datasets. In *IEEE Intl. Conf. on Big Data (Big Data)*, pp. 5028–5033.

Mrabet, A., A. Hassan, et P. Darmon (2019). Détection des données à caractère personnel dans les bases multidimensionnelles. In *EDA 2019*, Volume B-15 of *RNTI*, pp. 31–44.

Oram, A. (2015). Managing the data lake. sebastopol, ca, usa, 2015.

Quix, C., R. Hai, et I. Vatov (2016). Metadata extraction and management in data lakes with gemms. *Complex Systems Informatics and Modeling Quarterly* (9), 67–83.

Ravat, F. et Y. Zhao (2019a). Data lakes : Trends and perspectives. In *Database and Expert Systems Applications*, pp. 304–313.

Ravat, F. et Y. Zhao (2019b). Metadata management for data lakes. In *European Conference on Advances in Databases and Information Systems*, pp. 37–44. Springer.

Sawadogo, P. et J. Darmont (2021). On data lake architectures and metadata management. *Journal of Intelligent Information Systems 56*(1), 97–120.

Sawadogo, P. N., T. Kibata, et J. Darmont (2019). Metadata Management for Textual Documents in Data Lakes. In *ICEIS 2019*, Volume 1, pp. 72–83.

Scholly, E., P. Sawadogo, P. Liu, J. A. Espinosa-Oviedo, C. Favre, S. Loudcher, J. Darmont, et C. Noûs (2021). Coining goldmedal : a new contribution to data lake generic metadata modeling. *arXiv preprint arXiv :2103.13155*.

Spiekermann, M., D. Tebernum, S. Wenzel, et B. Otto (2018). A metadata model for data goods. In *Multikonferenz Wirtschaftsinformatik*, Volume 2018, pp. 326–337.

Theodorou, V., R. Hai, et C. Quix (2019). A metadata framework for data lagoons. In *European Conference on Advances in Databases and Information Systems*, pp. 452–462. Springer.

Walker, C. et H. Alrehamy (2015). Personal data lake with data gravity pull. In *2015 IEEE Fifth Intl. Conf. on Big Data and Cloud Computing*, pp. 160–167. IEEE.

Zhao, Y. (2021). *Metadata Management for Data Lake Governance*. Ph. D. thesis, Toulouse 1. Thèse de doctorat dirigée par Ravat, Franck. Informatique et télécommunications.

Summary

In order to improve the protection of personal data in data lakes, we propose a meta-model that includes several aspects of privacy. Our meta-model describe all necessary constraints for the implementation of personal data protection procedures (pseudonymization/anonymization) in data lakes. This meta-model also enhances GDPR compliance in data lakes by having, for example, a personal data processing log and the identification of the finality of each data integration. Our meta-model is presented via a conceptual schema (UML) and implemented via a graph database (Neo4j). The validation of our proposal is illustrated by modeling and discussing several personal data protection scenarios in data lakes.