

Document d'Architecture Générale TNSI

Document présentant l'architecture générale du futur SI CNAF.

Historique du document

Version	Date	Description de la modification	Auteur
0.x	17/05/2021	Initialisation	JFN
	29/06/2021	Ajout d'un bloc « évolution » sur les outils de développement suite au travail sur la fiche objectif du CC « outils de dev »	JFN
	05/07/2021	§3.4 précisions sur les conditions de livraison via l'usine logicielle	JFN
	05/07/2021	Le §3.5 est renommé afin de couvrir plus largement la maîtrise de la qualité des livrables applicatifs ; ajustements réalisés pour tenir compte des décisions du comité de suivi TNSI du 02/10/21	JFN
	05/07/2021	Intégration des commentaires de Fabrice C. : importance du DEX, travaux du chantier EDSI n°4 à intégrer, OPA en SaaS	JFN
	06/07/2021	Prise en compte des remarques de J. Ben-Soussan : suppression du mot boutique dans les cibles du Cloud Azure, ajout de la notion de PCA	JFN
1.x	16/07/2021	Validation en comité de pilotage TNSI Ajout des précisions demandées : <ul style="list-style-type: none"> • Les briques techniques pour le paiement • L'analyse des sujets de sécurité avec la dimension fonctionnelle. 	JFN
	20/07/2021	Intégration des commentaires MCIS	JFN

Détail des sous-versions : voir le versionning SharePoint

Sommaire

1. Introduction.....	4
1.1. Philosophie du TNSI	4
1.2. Objectif du document.....	4
1.3. Procédure de mise à jour du DAG	4
2. Principe d'architecture du TNSI.....	5
3. Les briques centrales du TNSI.....	6
3.1. Kafka	7
3.2. Service centralisé de capture des traces	7
3.3. Contrôle d'accès.....	7
3.4. Outils de développement.....	8
3.5. Maîtrise de la qualité des livrables applicatifs	10
3.6. Serveur d'application.....	10
3.7. Brique ETL.....	11
3.8. API management	11
4. Les briques périphériques	11
4.1. Moteur de calculs.....	11
4.2. Case management.....	12
4.3. GED et Éditeur.....	12
4.4. Chaîne de synchronisation NSI / SIA.....	12
5. La gestion de la donnée.....	12
5.1. Les systèmes de gestion de base de données autorisés	13
5.2. Accès unitaire aux données	14
5.3. Manipulation des données en masse.....	14
5.4. Gestion des configurations des systèmes de base de données.....	14
6. Sécurité	14
7. Les environnements du TNSI.....	15

Bibliographie

Etude SopraStéria « CNAF - TNSI - Architecture générale v1.0 » du 24/03/2021
 Etude SopraStéria « CNAF - TNSI - Cadre d'architecture des référentiels_v1.0 » du 12/03/2021
 Etude Thales « CNAF_Regard sur le TNSI_Rapport d'analyse_V1.0 » du 20/04/2021 et son support de restitution « 20210427_Support restitution_v1.1 »

1. Introduction

1.1. Philosophie du TNSI

La TNSI doit se construire sur les grands principes suivants :

- Centrer le SI sur la personne (données et services), intégrer et gérer ses événements de vie (proactivité)
- Unifier la gestion sur l'ensemble des Caf et mettre en œuvre des bases nationales
- S'appuyer sur des données certifiées et intégrer tous les référentiels nationaux
- Réduire la charge cognitive des agents (18 mois de formation) en augmentant l'automatisation des processus
- Augmenter la connaissance allocataire (V360)
- Développer une architecture technique de services normalisés, modulaires, réutilisables
- Mettre en œuvre les principes de sécurité moderne dès la construction du TNSI

L'architecture technique du TNSI décrite dans ce document répond avant tout aux 6^{ième} et 7^{ième} principes mais propose des briques techniques permettant de répondre aux 5 autres principes.

1.2. Objectif du document

Le document d'architecture générale de la TNSI explique les principes généraux d'architecture du futur SI de la branche famille. Il décrit les briques techniques de base du NSI de façon agnostique afin de rester sur un plan uniquement technique. Pour chacune de ces briques, il décrit la technologie à utiliser et fixe les règles techniques de bon usage de cette technologie (accès, sécurité, paramétrage). Les paramétrages « métier » spécifiques de ces briques sont à décrire usage par usage dans les documents d'architecture des applicatifs qui les mettront en œuvre.

La mise en place du NSI va se construire selon une trajectoire de 4 à 5 années, il est donc crucial de ne pas inscrire dans ce document des technologies qui seront obsolètes avant la fin de la TNSI (par exemple Angular 9 fin du Long Term Support – LTS – le 6 août 2021).

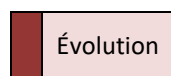
Les exigences liées à l'exploitation des services (ops) ne sont pas traitées dans ce document mais devront l'être dans les documents d'exploitation à fournir lors de la mise en œuvre des différents services. Toutes les briques de la TNSI doivent faire l'objet d'un document d'exploitation (DEX) conformément aux travaux du chantier EDSI n°4.

Sur le sujet ops, prévoir d'intégrer dans la V2 les principales recommandations du chantier EDSI n°4.

1.3. Procédure de mise à jour du DAG

Le DAG TNSI est géré par la cellule TNSI qui récupère les différentes demandes d'évolutions, les instruit et les intègre ou non dans le document. Les thématiques de la version courante qui font

L'objet d'étude pour d'éventuelles évolutions dans une version future sont indiquées par les encarts suivants :



Chaque nouvelle version du DAG est approuvée en comité de suivi TNSI.

2. Principe d'architecture du TNSI

L'architecture globale et les briques techniques décrites dans ce document répondent aux principes de conception suivants :

Modularité : le SI est découpé sous-ensembles les plus simples possibles et rattaché à un seul bloc fonctionnel du POS

Cohérence : Cohérence forte entre les sous-ensembles à l'intérieur des blocs fonctionnels.

Sécurité de la donnée et des informations sur les personnes selon les meilleurs standards du moment

Couplage faible : réduire les adhérences entre les blocs fonctionnels :

- Unicité de la source des données
- Unicité de l'orchestration de traitement
- Asynchronisme des traitements
- Mise en place de formats d'échange pivot
- Standardisation des interfaces et des flux – interopérabilité de la solution

Responsabilité : Chaque module est responsable du traitement des données. Il y a donc unicité du lieu de traitement de l'information.

Mutualisation : Réutilisation des composants, priorité aux briques mutualisées et privilégier l'intégration de solutions sur étagère (éditeur ou open source) à la construction de briques spécifiques

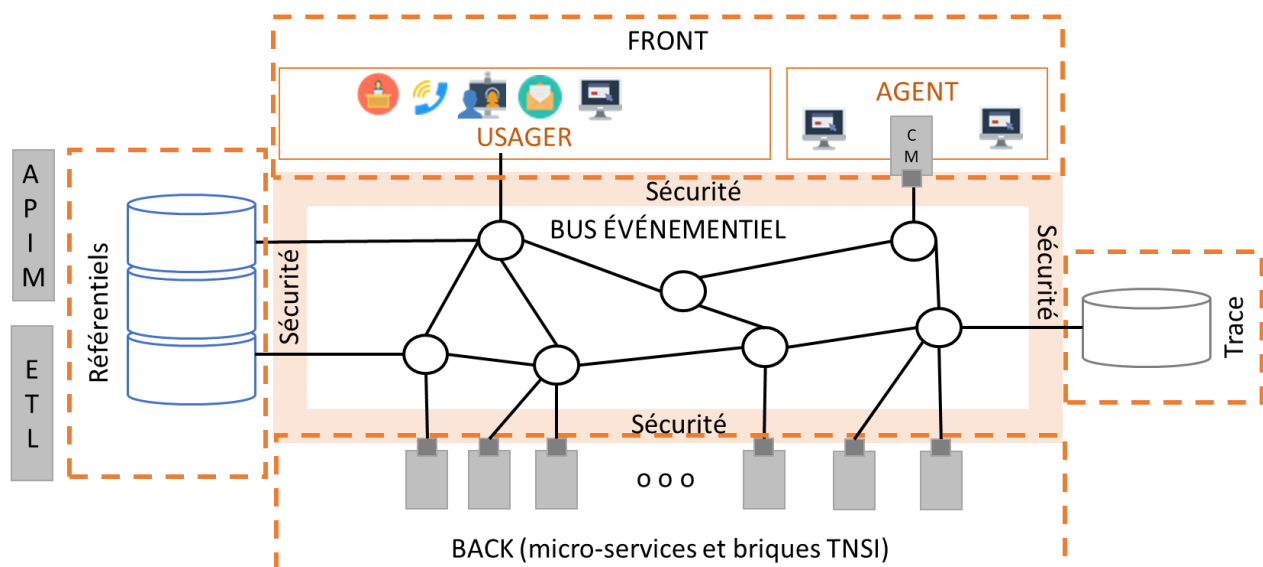
L'approche en micro-services a pour objectif de rendre le SI fortement modulaire et les briques périphériques sont bien rattachées qu'à un seul bloc fonctionnel. Les briques centrales sont, elles, très transverses et permettent une forte mutualisation des traitements élémentaires (bus Middle, sécurité, exposition des données). L'usage quasi-systématique du bus événementiel Kafka apporte un couplage extrêmement faible entre les briques (asynchronisme, orchestration, format pivot, interopérabilité). Les technologies choisis sont essentiellement des solutions sur étagère : kafka, talend, gravitee, postgresql, **Nosql**, sailpoint, OPA ...

La donnée est aussi au cœur de la TNSI avec des bases de données essentiellement organisées en référentiel et occupant donc une place très transverse dans l'architecture globale. Les procédures d'accès à ces données font l'objet de règles spécifiques pour garantir la cohérence avec les règles d'architecture générale.

Globalement les choix d'architecture de la TNSI visent à respecter la liste des exigences générales non fonctionnelles d'un SI est basé sur la norme ISO/IEC 25010.

Efficacité des performances	Compatibilité	Utilisabilité	Fiabilité	Sécurité	Maintenabilité	Portabilité
<ul style="list-style-type: none"> Comportement dans la durée Utilisation des ressources Capacité 	<ul style="list-style-type: none"> Co-existence Interopérabilité 	<ul style="list-style-type: none"> Opérabilité Protection contre les erreurs utilisateurs Accessibilité 	<ul style="list-style-type: none"> Maturité Disponibilité Tolérance aux pannes Résilience Survivabilité 	<ul style="list-style-type: none"> Confidentialité Intégrité Non-répudiation Responsabilité Authenticité 	<ul style="list-style-type: none"> Modularité Réutilisabilité Analysabilité Observabilité Facilité de modification Testabilité 	<ul style="list-style-type: none"> Facilité d'adaptation Facilité d'installation Facilité de remplacement

Le schéma synthétique de l'architecture de la TNSI est le suivant :



3. Les briques centrales du TNSI

Les briques centrales de la TNSI apportent avant tout les éléments transversaux indispensables :

- Bus événementiel pour piloter les flux d'échanges entre les composants techniques (briques périphériques, base de données, micro-services)
- Éléments de sécurité : traçabilité et authentification

Les briques centrales embarquent aussi tous les outils de développement permettant d'avoir une suite logicielle homogène pour produire les composants à développer en propre : micro-services « métier » et applications web. Cette homogénéisation va permettre de garantir une meilleure maîtrise de la dette technique de ces composants.

3.1. Kafka

Kafka est un bus événementiel capable de publier, stocker et traiter des flux très importants de messages. Il est conçu pour gérer des flux de données provenant de plusieurs sources et les fournir à plusieurs utilisateurs de façon asynchrone mais avec une latence très faible.

Tous les échanges entre les briques techniques TNSI doivent se faire via le bus **Kafka** que ce soit dans les liaisons Front <-> Back, les liaisons Back <-> Back et, quand cela est possible, pour consommer les référentiels de données.

La version kafka mise en œuvre dans la TNSI est la V2.8.0 (Released April 19, 2021) sous réserve de compatibilité avec la plateforme IaaS Oracle.

Toutes les modifications sur le paramétrage du service Kafka (topics, sécurisation, partition, replica, stream, mise à l'échelle...) doivent être vues avec le centre de compétence "Kafka" (cf. Annexe I pour avoir les contacts).

Les règles d'usage de Kafka seront affinées par le centre de compétence au fil des implémentations à venir de ce service.

Prochaines implémentations :

- Chantier 1 (améliorations AL) - Réécriture des injecteurs OPA
- GRC : chaîne de traitement entre captation des interactions et écriture dans la base interaction

3.2. Service centralisé de capture des traces

Toutes les traces, applicatives comme techniques, doivent être adressées au service centralisé de capture.

Les modalités de production et de gestion des traces seront définies dans la version V2 du DAG sur les aspects suivants :

- Contenu et mode de publication des traces applicatives ; étudier la mise à disposition d'une librairie dédiée pour intégration dans les micro-services (composition des traces et dépôt via Kafka)
- Identification du contenu et du mode de récupération des traces techniques des logiciels TNSI
- Identification du contenu et du mode de récupération des traces des référentiels de données (usage de Kafka connect)
- Exploitation des traces (traitement dans le SID, purge, sécurité -SOC...)

3.3. Contrôle d'accès

Les accès aux micro-services TNSI et aux briques périphériques doivent être contrôlés. Un serveur de jeton doit être mis à disposition des différentes briques techniques pour pouvoir obtenir les droits d'accès nécessaires.

L'outil de gestion des authentifications et ses règles de gestion des accès seront définis dans la V2 du DAG et étudiant notamment les points suivants :

- Outil de production des jetons interfacé avec l'IAM
- La gestion des secrets (autorité de certification, coffre-fort de clés)
- Règles de composition et d'usage des jetons (claims JWT par exemple)

3.4. Outils de développement

Les outils à utiliser pour tous les développements de la TNSI (micro-services, application web, ...) sont obligatoirement les suivants :

- JDK : **java 11.0.1**
- Framework de développement micro-services : **SpringBoot 2.5.0** et ses dépendances notamment :
 - Web (Tomcat Embedded)
 - Spring Kafka
 - Spring Data + SpringBatch + HikariCP
- Normalisation des API TNSI : **OAS 3.0** (OpenAPI 3.0 Specification www.openapis.org) et exposition **swagger**
- Gestion des projets Java : **Maven**
- Containerisation : **Docker 3.x**
- Framework de développement application web : **Angular 12**

La liste des outils et leur version seront affinées par le centre de compétence.

Points spécifiques à traiter :

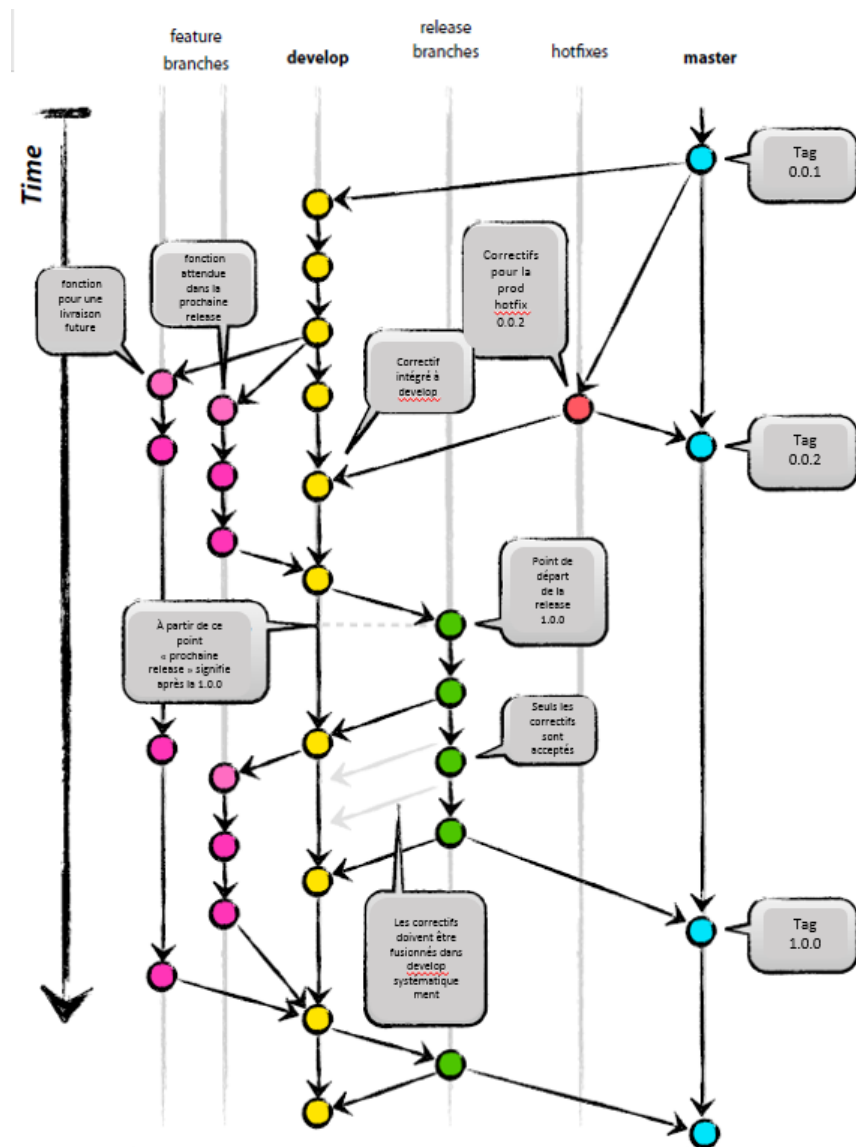
- Prise en compte des travaux du chantier EDSI n°4
- Modalités d'échange avec l'usine logicielle et prise en compte des recommandations du chantier EDSI n°9 notamment dans la gestion des livraisons sous forme de container
- Version JVM liée au JDK retenu
- Gestion du Responsive design (Bootstrap, Springboot material)

Le centre de compétences « outils de développement » fournit une « devbox TNSI » pour les développements d'applications. La « devbox TNSI » est donc un environnement local de développement avec les composants techniques décrits ci-dessus, elle s'appuie sur une machine virtuelle (VM) permettant la rapidité de mise œuvre pour les développements.

L'outil de gestion de version doit être GitLab.

La gestion de version doit obligatoirement s'appuyer sur le modèle GitFlow, multi-branches, standard du marché sur les processus de gestion de version et qui permet :

- Le développement de fonctionnalité en parallèle,
- De concevoir de façon plus modulaire des composants logiques,
- De maîtriser dans le détail le déploiement des fonctionnalités et de réagir plus rapidement en cas de problème à fixer.



Les branches master et develop sont protégées et les autres branches ont une durée de vie limitée dans le temps :

- La branche de feature qui permet le développement d'une fonctionnalité,
- La branche de release qui permet de définir le contenu d'une nouvelle version du produit,
- La branche de hotfix qui permet de livrer rapidement une version corrigée du produit,

Ces branches sont créées à partir des branches ad-hoc sous contrôle du centre de compétences « outils de développement ». Des rôles et responsabilités sont associés au cycle de vie de chaque branche :

- Le « release manager » qui gère les branches master, release et hotfix,
- L'« intégrateur composant » qui gère la branche develop,
- Le développeur qui gère ses branches de feature,

Toutes les livraisons de composants TNSI doivent se faire **via l'usine logicielle** et sous forme de **conteneur Docker**.

Les demandes de dérogation ou d'évolution relatives aux outils de développement sont à adresser au centre de compétences « Outils de développement » (cf. Annexe I pour avoir les contacts). Tous les sujets concernant la livraison des composants développés sont à traiter avec l'usine logicielle.

3.5. Maîtrise de la qualité des livrables applicatifs

Pour assurer le contrôle de la qualité des livrables applicatifs réalisés dans la TNSI, les outils et les procédures seront adaptés sur les aspects suivants :

- Contrôle du code
- Exigences non fonctionnelles : elles sont contrôlées lors des réunions de lancement de projet et lors des comités d'architecture.

Le DAG V2 doit préciser les modalités et les outils de contrôle qualité des développements :

- Intégrer les recommandations du chantier EDSI n°3
- Revue de code à organiser,
- Audit à une fréquence à fixer,
- Les outils de test et leur mode d'usage : SonarQube, CAST, LoadRunner.

Un outil de gestion des cas de test est à l'étude.

3.6. Serveur d'application

Le serveur d'application à utiliser pour les micro-services Middle / Back et pour les services Front doit être **Tomcat 9.0.x** (compatibilité Springboot 2.5)

L'outil de gestion de la persistance de type cache doit être **Redis 6.2**.

La compatibilité Redis 6.2 avec Springboot 2.5 sera confirmée dans le DAG version 2

Toutes les modifications sur le paramétrage des serveurs d'application (paramètre spécifique, mise à l'échelle...) doivent être vues avec le centre de compétence " Outils de développement " (cf. Annexe I pour avoir les contacts).

3.7. Brique ETL

L'outil ETL à utiliser pour toutes les manipulations de données en masse doit être **Talend 7.3**.

Le studio Talend qui permet de générer les artefacts Java de traitement des données n'est accessible que par les membres du centre de compétences « ETL ». Pour toutes les demandes de traitement ETL, il faut donc contacter le centre de compétences « ETL » (cf. Annexe I pour avoir les contacts).

3.8. API management

Toutes les API devant être inscrites au catalogue CNAF et exposées de façon contrôlée (contrôle d'accès, traçabilité) doivent l'être au travers la plateforme de gestion d'API **Gravitee V3.5.12**.

Le paramétrage des API à exposer via la plateforme d'API management est réalisé exclusivement par le centre de compétence « APIM » (cf. Annexe I pour avoir les contacts).

4. Les briques périphériques

Les services périphériques s'appuient fortement sur les briques de base pour interagir avec les autres composants du TNSI mais ils offrent des services à plus haute valeur ajoutée :

- Moteur de calculs pour les traitements essentiels sur les prestations
- Case Management pour organiser les traitements à faire au niveau du poste de travail agent et organiser les flux des différents canaux d'interaction avec les usagers (omnicanalité)
- GED + éditique
- Chaîne SSN

Les briques techniques nécessaires pour la mise en œuvre des fonctionnalités de paiement devront être intégrées au DAG.

4.1. Moteur de calculs

Tous les traitements TNSI qui peuvent être modélisés sous forme de règles de calcul bien déterminés et d'algorithmes de décision simples doivent être implémentés sous le moteur **Oracle**

Policy Automation (OPA) (OPA est mis en œuvre en SaaS la version est celle du proposé par Oracle).

Avant tout développement de traitement de cette nature le centre de service « OPA » doit être contacté pour étudier la faisabilité du traitement à implémenter (cf. annexe I pour avoir les contacts).

Le moteur OPA ne doit être utilisé que sous forme d'appel unitaire (par exemple 1 appel = 1 traitement pour 1 usager), les traitements en mode batch sont donc déconseillés.

Les règles d'accès au service OPA seront affinées (kafka, API, sécurité) par le centre de services au fil des implémentations à venir de ce service. Le devenir du mode batch va dépendre beaucoup des tests de performance à faire sur des injections unitaires notamment en injection directe via kafka.

Prochaine implémentation : Chantier 1 (améliorations AL) - Réécriture des injecteurs OPA

4.2. Case management

Paragraphe à rédiger à l'issu des travaux en cours sur le choix de l'outil : cible version 2 du DAG fin 2021

4.3. GED et Éditique

Paragraphe à rédiger à l'issu des travaux en cours sur le choix de l'outil : cible version 2 du DAG fin 2021

4.4. Chaîne de synchronisation NSI / SIA

Pour accompagner le transfert des services depuis le SIA vers le NSI, la TNSI prévoit un service de synchronisation entre ces 2 environnements pour maintenir une cohérence entre les données utilisées par chacun de ces systèmes.

A étudier sur la base de la chaîne SSN

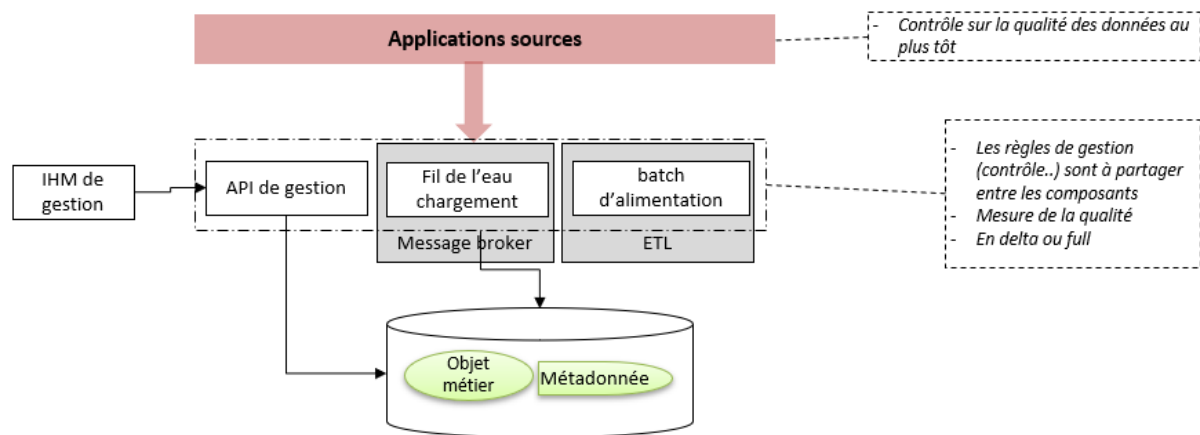
Prochaines implémentations :

- Chantier 1 (améliorations AL) – traitement des objets « personne » et « foyer » de SIA vers NSI

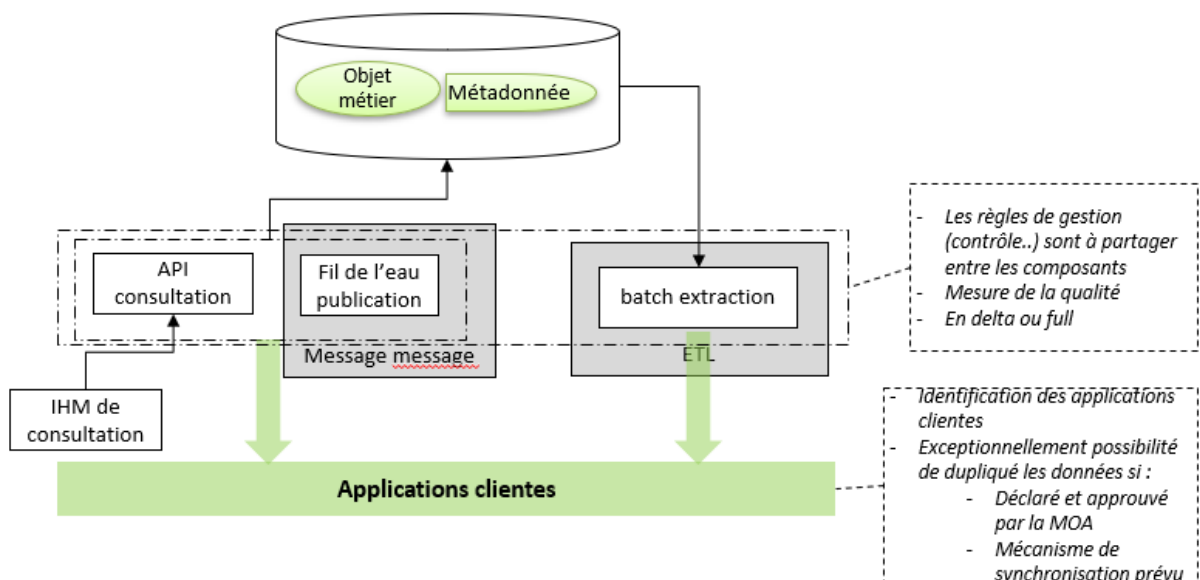
5. La gestion de la donnée

La gestion de la donnée doit répondre aux schémas d'architecture suivants :

Collecte :



Consultation :



La description des objets métier implémentés ainsi que les mécanismes de contrôle de la qualité et de purge à mettre en œuvre ne sont pas abordés dans ce document mais ils doivent être décrits dans les documents d'architecture des différentes bases de données mises en œuvre.

Evolution à prévoir à la suite de la prestation SopraStéria sur la gouvernance de la données

5.1. Les systèmes de gestion de base de données autorisés

3 systèmes de base de données sont proposés :

- **Exadata** (version Cloud Oracle) pour les bases de données Sql existantes
- **Postgresql 13.3** (release 2021-05-13) pour les bases de données Sql en substitution d'Exadata dans une approche Finops
- **Nosql** pour des bases de données Nosql en mode document

L'usage de l'un ou l'autre de ces systèmes est ouvert mais doit être validé par le centre de compétences "SGBD" (cf. Annexe I pour avoir les contacts). En fonction des usages, ce choix doit néanmoins tenir compte des principes suivants :

- SQL : données complexes avec de fortes contraintes d'intégrité sur des modèles très peu évolutifs et pour des volumes de données relativement constants ; usage transactionnel depuis des applications qui peuvent être très variées (référentiel des personnes par exemple).
- NoSQL : données évolutives en termes de modèle et de volumétrie ; usage fortement restreint (une base = 1 application) mais pouvant faire l'objet de traitement très lourd (log ou données brut avant retraitement par exemple).
- Finops : à performance à peu près identique, privilégier le mode d'hébergement qui réduit des coûts de « Run ».

5.2. Accès unitaire aux données

L'accès aux données directement en base de données est interdite, 2 modes d'accès sont autorisés :

- Collecte sous la forme unitaire via une interface API REST (CRUD) standardisée. La sécurisation est portée par l'API d'accès via la présentation d'un jeton d'authentification
- Collecte en mode asynchrone. Les demandes d'accès doivent être postées via Kafka pour être traités via ses connecteurs aux bases de données.

Le mode d'accès en mode asynchrone doit être évalué (performance, sécurisation, paramétrage) par les centres de compétences impactés (Kafka, gestion de la données).

5.3. Manipulation des données en masse

Les manipulations de données en masse pour des opérations de chargement en interne (alimentation SID) ou d'exposition vers des partenaires (via la PEM) doivent obligatoirement faire usage de la brique ETL.

La sécurisation de l'accès aux données est portée par l'ETL.

5.4. Gestion des configurations des systèmes de base de données

Toutes les modifications sur le paramétrage des bases de données ou de leur système de gestion (création de base, mise à l'échelle...) doivent être vues avec le centre de compétence "SGBD" (cf. Annexe I pour avoir les contacts).

6. Sécurité

A rédiger à la suite de l'analyse de risque en cours par Thalès

L'analyse ne doit pas se limiter au seul volet technique et doit être traitée globalement avec la dimension fonctionnelle.

Sécurité :

- D : en fonction besoins + analyse de risque
- I : identification systématique des accès avec jeton
- C : en fonction besoins + analyse de risque
- T : traçabilité de toutes les opérations
- Réglementaire : CNIL et autres

7. Les environnements du TNSI

La logique de répartition des différentes briques TNSI est la suivante :

- Cloud Oracle pour tout ce qui a trait aux allocataires ou aux usagers (prestations individuelles, élaboration des paiements, créances, contentieux, intermédiation financière)
- Cloud Microsoft Azure pour tout ce qui a trait aux agents (poste de travail)
- Datacenter CNAF au legacy qui ne migrerait ni vers le Cloud Oracle, ni vers le Cloud Microsoft Azure
- Cloud Talend pour l'atelier de conception ETL (les exécutables sont déployer sur le cloud Oracle)

Brique TNSI	Cloud	Mode de gestion	Environnements disponibles
Kafka	Oracle	IaaS	A préciser
OPA	Oracle	SaaS	A préciser
Talend Studio	Talend	PaaS	A préciser
Gravitee	Oracle	IaaS	A préciser
Postgres	Oracle	IaaS	A préciser
Nosql	Oracle	IaaS	A préciser
Micro-service Back Script ETL	Oracle	IaaS	A préciser

Front (caf.fr ...)	Oracle	Iaas	A préciser
Devbox TNSI	A préciser		A préciser
Git TNSI	A préciser		A préciser

Le DAG version 2 permettra de fixer en détail les environnements (PROD, PrePROD, QUALIF Métier, QUALIF Tech, PRA, PCA) disponibles pour chaque brique à la suite de l'étude des clauses de nos contrats d'hébergement Cloud.

Les éventuelles contraintes dans les flux réseau seront identifiées.

Annexe I : Centre de compétences - Centre de services

CENTRE DE COMPETENCE	CONTACT
ETL	Frédéric Barale
APIM	Frédéric Barale
KAFKA	Xavier Bottex
OUTILS de DEVELOPPEMENT	Christophe Roussel
SGBD	En cours de nomination
CASE MANAGEMNT	A définir

CENTRE DE SERVICE	CONTACT
OPA	Christophe Bouffay