RAPPORT DE STAGE

Master 2

Modélisation et intégration de données de capteurs/compteurs du SGE

par
Ines BEN KRAIEM

Tuteurs de stage :
Hervé CROS, Faiza GHOZI, André PENINOU

Juillet 2017

DEDICACES

...A ma chère famille

...A mes chers amis

...A tous ceux

qui comptent pour moi

...A tous ceux pour

qui je compte

Je dédie ce travail

Ines BEN KRAIEM

REMERCIEMENTS

Je tiens à remercier vivement les personnes qui ont contribué à l'aboutissement de ce travail

Mme Faiza GHOZZI, pour ses aides et ses conseils tout le long de ce projet. J'espère être à la hauteur de sa confiance et qu'elle trouve dans ce travail l'expression de ma profonde gratitude.

Mr André PENINOU, pour la qualité de son encadrement, ses critiques et ses conseils enrichissants qui m'ont permis d'atteindre mes ambitions initiales. Je vous remercie pour votre sympathie et votre aide.

Mr Olivier TESTE, qui m'a accueilli dans l'équipe SIG et au sein du laboratoire IRIT. Je tiens à vous remercier pour la confiance que vous m'avez accordée.

Mr Hervé CROS, qui m'a accueilli dans le service SGE et m'a fourni tous les moyens pour que je puisse travailler dans les meilleures conditions, ainsi que toute l'équipe de la société qui m'ont soutenu tout au long de mon stage.

J'adresse aussi mes remerciements aux membres du Jury Mme Lamia FOURATI et Mme Salma JAMMOUSI pour avoir accepté de juger, d'évaluer et d'enrichir ce travail. Votre présence me fait honneur. Je vous témoigne mon grand respect et ma reconnaissance infinie.

Je tiens aussi à remercier tous mes enseignants qui ont contribué à ma formation universitaire. Je vous témoigne mon grand respect et ma reconnaissance infinie.

Enfin je remercie tous mes amis avec qui j'ai partagé des moments d'entraides, de faiblesse, de souffrance, de joie et de courage.

Table des matières

1 Cadre du projet 11
1 Introduction 11
2 Cadre général 11
2.1 Présentation de l'organisme d'accueil 11
2.2 Présentation général du projet 14
2.3 Objectifs préliminaires 14
2.4 Planning du projet 15
3 Analyse de l'existant 15
3.1 Étude de l'existant 15
3.2 Critique de l'existant 19
3.3 Analyse et spécification des besoins 20
4 Systèmes d'intégration pour l'analyse décisionnel 21
4.1 Solution d'intégration à court terme 21
4.2 Solution d'intégration Big Data 22
5 Conclusion 24
1 L'informatique décisionnelle 25
1 Introduction 25
2 Concepts généraux du BI 25
2.1 La Business Intelligence 25
2.2 Les principes des systèmes décisionnels 26
3 La différence entre OLTP et OLAP 28
4 Démarche de construction d'un entrepôt de données 29
4.1 Modélisation et Conception de l'entrepôt 30
4.2 Alimentation de l'entrepôt 30
4.3 Administration 32
4.4 Restitution 32
5 Le Big Data 33
5.1 Définition 33
5.2 Les bases des données NoSQL 33
6 Conclusion 36

1 Conception 37
1 Introduction 37
2 Sources de données 37
2.1 Source METASYS 37
2.2 Source PcVue 38
2.3 Source globale 38
3 Conception du schéma multidimensionnel 39
3.1 Méthode ascendante 40
3.2 Démarche descendante 42
3.3 Démarche mixte 45
4 Schéma logique Big Data 46
4.1 Choix du modèle NoSQL 47
4.2 Choix de l'outil d'analyse 47
4.3 Structure du schéma multidimensionnel dans Hbase 48
5 Conclusion 49
1 Conception et Développement de l'ETL 51
1 Introduction 51
2 Environnement et langage de développement 51
2.1 Environnement logiciel 51
2.2 Langages de développement 53
3 Développement de l'ETL pour la solution à court terme 53
3.1 Problèmes rencontrés et étude de la solution 53
3.2 Diagrammes d'activités 55
3.3 Description des processus ETL 57
4 Développement de l'ETL pour les données à long terme 65
4.1 Intégration de l'historique sous SQL Server 65
4.2 Développement de l'ETL pour le Big Data 69
4.3 Création du schéma multidimensionnel en étoile 70
5 Conclusion 73
1 Restitution 75
1 Introduction 75
2 Modélisation de l'application BI 75
2.1 Type d'application BI 75

2.2 Type d'analyse BI....76
2.3 Type des utilisateurs BI....76
3 Développement de l'application BI....76
3.1 Analyse des données avec PcVue....76
3.2 Analyse de l'historique....77
4 Conclusion....80

CHAPITRE 1 : Cadre du projet

1 Cadre du projet

Modélisation et intégration de données de capteurs/compteurs du SGE

1 Introduction

Dans ce chapitre, nous présentons l'organisme d'accueil au sein duquel s'est déroulé notre projet, situons le présent travail dans son contexte général, spécifions les besoins et mettons en relief l'analyse et critique de l'existant et enfin nous proposons la solution aux problèmes soulevés.

2 Cadre général

Dans le cadre de la préparation du projet de fin d'étude en vue de l'obtention du diplôme National d'Ingénieur en Informatique, Technologie Web et Multimédia, nous avons choisi d'effectuer un stage d'une durée de 4 mois au sein du SGE et en collaboration avec le laboratoire IRIT. Le stage a débuté le lundi 13 Mars 2017 et s'est achevé le lundi 13 Juillet 2017. L'objectif premier était de mettre en place un système décisionnel pour les données énergétiques des bâtiments et réseaux du campus de Rangueil (ou se trouve entre autre l'Université Paul Sabatier) provenant de plusieurs capteurs et récupérées de différents bâtiments dans le but de visualiser [17] et mettre à disposition ces données aux utilisateurs [15] [18].

2.1 Présentation de l'organisme d'accueil

L'équipe SIG (Systèmes d'Informations Généralisés) affilée au laboratoire IRIT, nous a proposé le sujet d'intégration et de modélisation des données de capteurs/compteurs du SGE. Le projet est élaboré au sien du SGE le Service de Gestion et d'Exploitation de la Chancellerie des Universités du Rectorat de l'Académie de Toulouse.

2.1.1 Le Service de Gestion et d'Exploitation

C'est un service du rectorat spécialisé dans la gestion, l'exploitation et l'investissement des réseaux mutualisés des campus de Rangueil. Il gère les données liées aux différentes

installations en termes de fluides (énergie, eau, air comprimé) sur différents campus.

Présentation des activités

Le SGE gère plusieurs pôles d'activités :

• Le pôle maintenance exploitation avec 11 Agents de l'Education Nationale : ce service concerne les activités électromécaniques telles que les réseaux et infrastructures électriques Haute Tension, les réseaux et infrastructures de chauffage urbain, l'éclairage Public, la gestion technique centralisée (GTC), l'air comprimé, les réseaux d'eau et d'assainissement, les réseaux de gaz Arrosage, Eau Usées, Eaux pluviales et Réseau Gaz.

• Le pôle espaces extérieurs avec 11 Agents de l'Education Nationales : ce service gère les espaces verts et Voirie.

• Le pôle administratif avec 6 agents de l'Education Nationales

• Le pôle travaux SIGT avec 4 agents de l'Education Nationales

Organigramme

Le SGE compte 36 personnes occupant différents postes. La figure 1 représente l'organigramme de l'entreprise

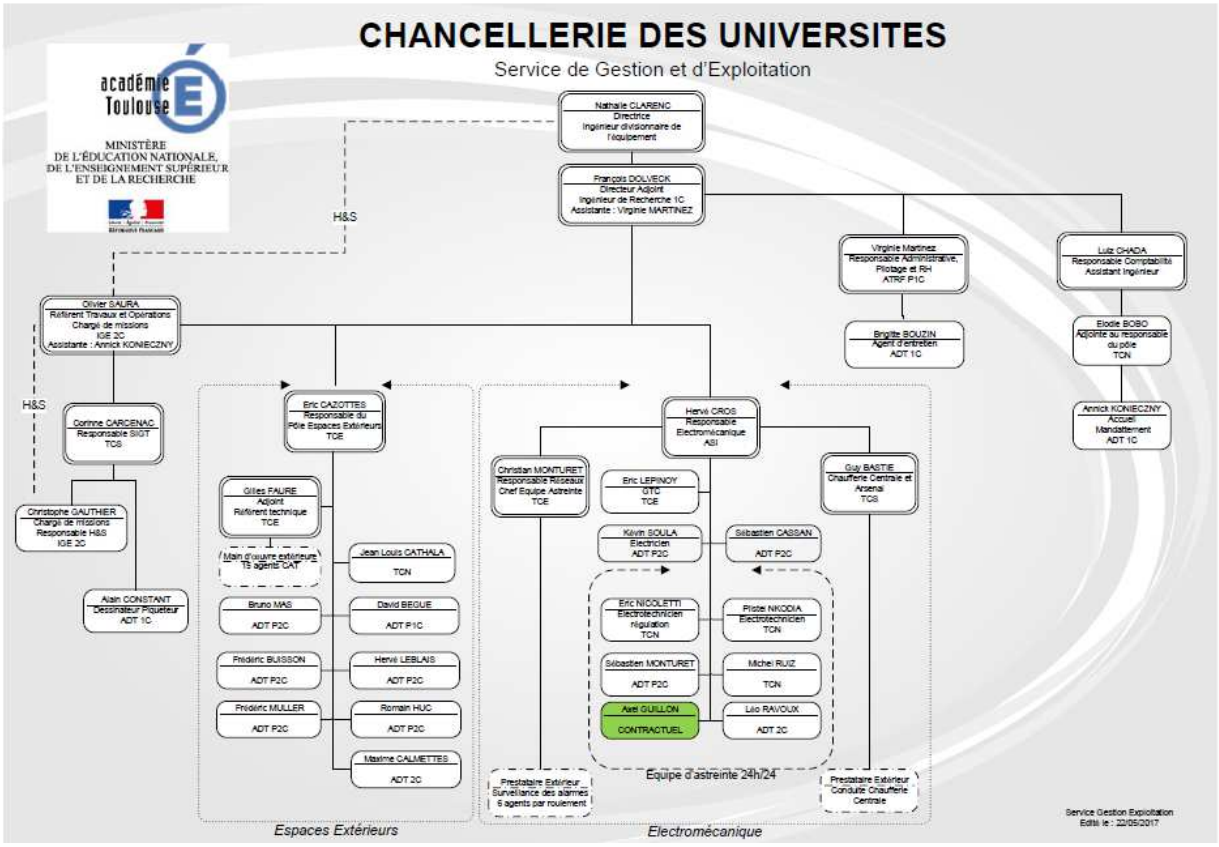


Figure 1: Organigramme du SGE

2.1.2 Le laboratoire IRIT

L'Institut de Recherche en Informatique de Toulouse (IRIT) [1] est l'un des piliers de la recherche en Midi-Pyrénées avec ses 700 membres, permanents et non-permanents. De par son caractère multi-tutelle (CNRS, INPT, Universités toulousaines), son impact scientifique et ses interactions avec les autres domaines, le laboratoire constitue une des forces structurantes du paysage de l'informatique et de ses applications dans le monde du numérique, tant au niveau régional que national. Notre projet était en collaboration avec l'équipe SIG (Systèmes d'Informations Généralisés) qui se compose de 19 permanents, et d'une trentaine d'étudiants post-doctorants, doctorants ou stagiaires, et d'ingénieurs de recherche. Ses recherches s'inscrivent dans le domaine des Systèmes d'Informations (Information Systems) et ont pour objet scientifique la donnée et l'utilisateur. Elles concernent notamment les problématiques d'accès à

[1] https://www.irit.fr/

la donnée brute et élaborée pour les utilisateurs.

2.2 Présentation général du projet

Le Service de Gestion et d'Exploitation gère deux systèmes de gestion des données de capteurs, demandant des tâches lourdes et complexes de manipulations et d'extractions de ces données de différentes sources (10000 points de comptage). Le système de gestion d'exploitation visé dans ce stage,nommé METASYS, stocke les données sous forme de fichiers plats et indépendants organisés en une hiérarchie de répertoires. Un fichier concerne un capteur (température, vannes, pression, …) ou un compteur (eau chaude, électricité, …) et conserve les différents relevés automatiques réguliers (toutes les 30', 15 minutes, 2 heures, …). Par conséquent, chaque relevé concerne une valeur avec un horodatage. Dans un même temps, ce système est entrain d'être migré vers un autre système de monitoring, appelé PCVUE, qui dans l'avenir contiendra la majorité des données des capteurs. Le besoin était de mettre en place une solution d'intégration des données de ces systèmes de supervision METASYS et PCVUE dans une base de données afin de faciliter l'extraction et la manipulation de ces données entre les deux systèmes par l'intermédiaire d'outils disponibles sur PCVUE. La totalité des données d'historique ne pouvant pas être traitée par PCVUE, il nous a été demandé de mettre en place une base pour l'ensemble de l'historique pour faire des analyses et des comparaisons entre les différents fichiers. Une solution simple et facile pour pouvoir explorer et visualiser [17] les données de cette base devra être proposée.

2.3 Objectifs préliminaires

Le « SGE» souhaite apporter des solutions pour la modélisation, le stockage et l'exploration des données générées par les capteurs/compteurs afin de pouvoir répondre à la variété des besoins et exigences d'accès et d'analyses des utilisateurs. Pour cela, le SGE espère:

- Avoir une solution (modèle, architecture, outils) qui permette d'extraire et de stocker les données, pour pouvoir ensuite les explorer et les visualiser suivant différents critères [15] [18].
- Avoir une solution d'intégration de données des fichiers dans une source unique et non volatile.
- Recenser les besoins des utilisateurs en termes d'analyse des données (pour

l'exploitation directe sur un bâtiment par exemple, pour des analyses à long terme et comparer des consommations sur plusieurs années par exemple, pour l'extraction de données, …).

• Avoir des préconisations en termes de SGBD à retenir, de logiciel de visualisation adapté [18], de matériel nécessaire (serveur, …), et de procédures d'exploitation à mettre en place.

• Avoir à disposition des décideurs les différents outils d'exploration des données en toute sécurité afin de favoriser une meilleure prise de décision.

• Avoir une solution pour piloter la performance durable.

Notre ambition est de répondre aux besoins du SGE en utilisant les technologies liées aux bases de données, aux entreposages de données et à l'informatique décisionnelle [1] [8] [12].

2.4 Planning du projet

La planification est une étape importante dans le déroulement du projet parce qu'elle traduit l'organisation du projet et l'estimation du temps nécessaire à la réalisation des différentes tâches. Pour cela nous décomposons le projet en plusieurs tâches et nous prévoyons à chaque tâche le temps nécessaire pour sa finalisation. La figure 2 présente les différentes étapes à suivre pour la réalisation de notre projet (provisoire).

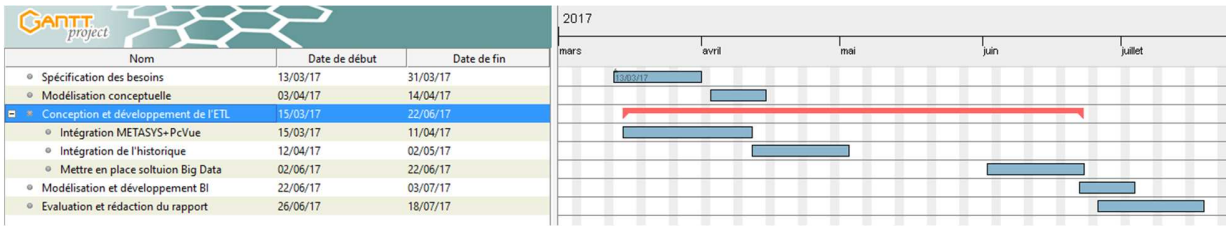


Figure 2: Planning prévisionnel

3 Analyse de l'existant

3.1 Étude de l'existant

Plus de 5000 de capteurs sont répartis dans tous les bâtiments. Les capteurs permettent de suivre et gérer l'ensemble des équipements exploités par le SGE, comme ,par exemple, des sondes de températures, des commandes de vannes de régulation et des compteurs. Au sein du

SGE, il y a deux systèmes de gestion de données qui cohabitent:

• METASYS, système propriétaire de supervision, gère essentiellement les équipements de CVC (Chauffage traitement d'air). Cette supervision communique exclusivement avec des automates régulateurs de marque Johnson Controls.

• PCVUE, système de supervision de type ouvert, intègre les équipements de CVC, de haute tension, d'air comprimé, d'éclairage public... Cette supervision communique avec des automates de toute marque.

Le SGE dispose d'une base de données Access qui représente le patrimoine des équipements en gestion dans le sens où il y a l'ensemble des informations sur les bâtiments, les clients, les équipements, les visites et les maintenances des équipements et des capteurs.

3.1.1 Système METASYS

Description

METASYS est un système de supervision pour l'ensemble des fonctions techniques du bâtiment. Il gère essentiellement les équipements de CVC mais il ne permet pas le suivi de la performance énergétique et l'édition des rapports pour surveiller les consommations d'un bâtiment.

Architecture

METASYS stocke les données sous forme de fichiers plats d'extension .DBF. Ces fichiers représentent 252187 fichiers comme historique, soit 151 Go de données brutes, et 2635 fichiers comme données opérationnelles de chauffage sur 6 mois d'une taille de 4 Go. Le SGE est entrain de migrer du système METASYS vers le système PcVue. En effet, tous les nouveaux automates sont maintenant branchés directement sur PcVue et non pas sur METASYS. La base de données de METASYS est organisée en une hiérarchie de répertoires comme le montre la figure 3.

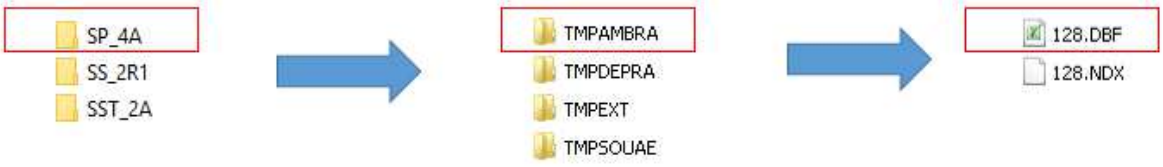
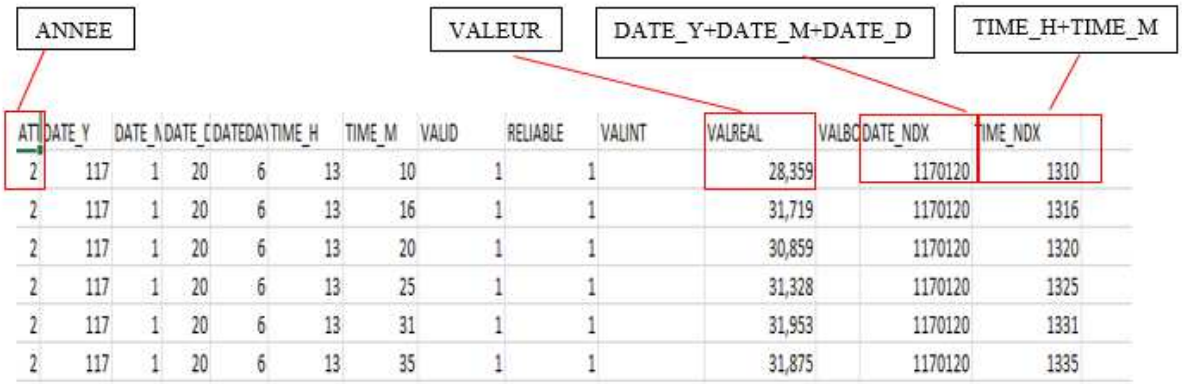


Figure 3: Hiérarchie des répertoires de données METASYS

Par exemple, le nom du répertoire SP_4A indique le nom du local technique SP suivi d'une référence du bâtiment 4A. En effet, le SGE dispose de plusieurs locaux techniques qui décrivent les Sous-Stations de chauffage. Parmi lesquels nous trouvons :

- SSP : Sous Station Primaire.
- SS : Sous Station secondaire.
- LT : Local Technique.
- CTA : Central de traitement d'air.
- PHT : Poste Haute Tension.
- TGBT : Tableau Général Basse Tension.

Sous chaque répertoire de local technique, nous trouvons des répertoires qui représentent la liste des capteurs. Finalement, nous arrivons aux fichiers .dbf qui représentent les données de capteurs(4).



ATT	DATE_Y	DATE_M	DATE_D	DATEDAY	TIME_H	TIME_M	VALID	RELIABLE	VALINT	VALREAL	VALBOOL	DATE_NDX	TIME_NDX
2	117	1	20	6	13	10	1	1		28,359		1170120	1310
2	117	1	20	6	13	16	1	1		31,719		1170120	1316
2	117	1	20	6	13	20	1	1		30,859		1170120	1320
2	117	1	20	6	13	25	1	1		31,328		1170120	1325
2	117	1	20	6	13	31	1	1		31,953		1170120	1331
2	117	1	20	6	13	35	1	1		31,875		1170120	1335

Figure 4: Extrait d'un fichier.DBF

Ces fichiers contiennent des informations importantes concernant la date et la valeur de la mesure. En revanche pour avoir une date complète qui désigne une mesure, il faut concaténer les champs Att, Date_NDX et TIME_NDX.

3.1.2 Système PCVUE

Description

PcVue est un système de supervision temps-réel permettant à l'opérateur de visualiser

et de piloter la production, la gestion d'alarmes, l'affichage de tendances, l'archivage de données et l'acquisition de données. Il est utilisé dans le contrôle industriel, la gestion de bâtiments, la gestion d'énergie, la distribution électrique, le chauffage, l'automatisation des sous-stations, la sécurité, la détection incendie, le transport, les énergies renouvelables et les infrastructures. Ce logiciel, qui est donc un outil de supervision, fonctionne en interaction avec les automates qui modélisent les différentes informations envoyées par les capteurs et permet de générer des courbes en fonction des informations de la base.

Architecture

PcVue est en connexion permanente avec une base propriétaire sous forme de fichiers d'extension .dat, qui permet le stockage de toutes les informations des capteurs. Étant donné que les capteurs envoient une information à chaque seconde, la base de données contient des millions de lignes. Cette base occupe pour une période de 6 mois 10 Go de données brutes et pour son historique qui date depuis 2010, nous avons 80 Go sur un premier lecteur et 15 Go sur un deuxième lecteur. Le principe de PcVue est la création des variables qui représentent des points. Ces points représentent des chemins pour les fichiers qui contiennent les valeurs des capteurs. Chaque fichier contient les informations de plusieurs capteurs comme le montre la figure 5.

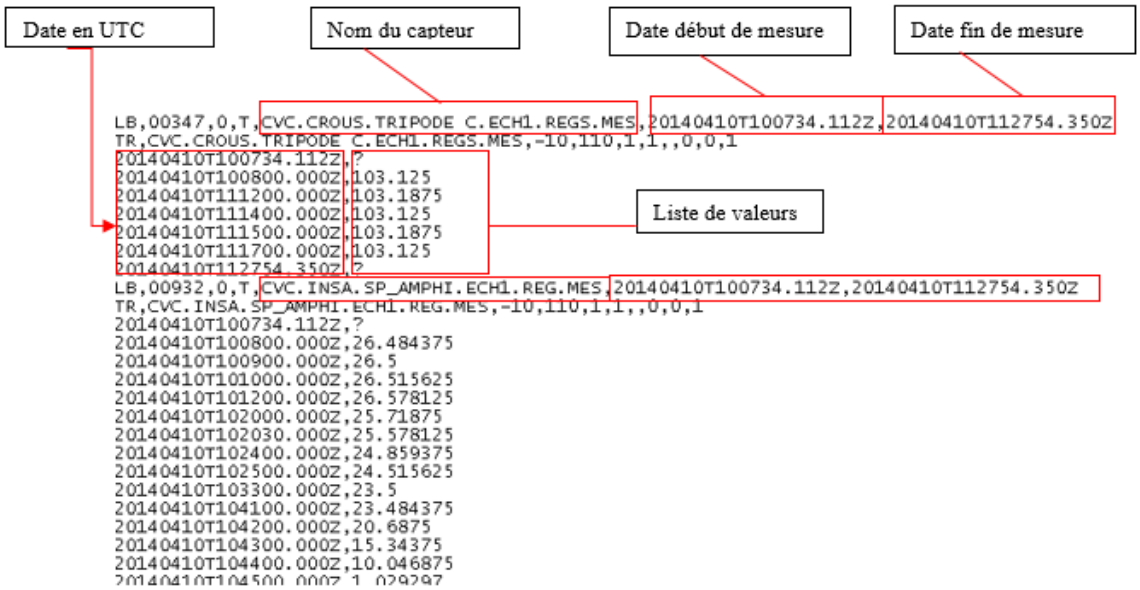


Figure 5: Extrait des données d'un fichier.dat

Ces fichiers sont une longue liste de chiffres et de lettres comportant des informations précises. Le bandeau principal indique le nom du capteur qui est sous forme de chemin indiquant le nom du client 'CROUS', le nom du batiment 'TRIPODE C', l'équipement 'ECH1' et le type de mesure 'REGS'. Egalement, nous distinguons deux dates dont l'une est la date de début de la mesure et l'autre la date de fin de la mesure. Finalement, nous trouvons une liste de dates en UTC (Coordinated Universal Time) suivie de valeurs générées par le capteur.

3.1.3 Base de données patrimoine ACCESS

Le SGE dispose d'une base de données Access qui représente le patrimoine de la société. Elle contient les informations sur les clients du SGE, les bâtiments, les locaux techniques, les équipements et les capteurs.

3.2 Critique de l'existant

Un des principaux problèmes du SGE est l'accès à un historique des données pour effectuer des comparaisons et des analyses. A ce jour, le système METASYS n'offre pas une gestion de l'historique supérieure à 6 mois et ne permet pas de croiser facilement des données de plusieurs capteurs/compteurs. Egalement, il ne permet ni de faire une comparaison entre deux fichiers qui ont un horodatage différent ni de faire une comparaison avec des fichiers provenant d'autres systèmes différents. En ce qui concerne les courbes offertes par METASYS, le système ne permet pas l'affinage des courbes à travers un calendrier pour choisir les dates voulues et remonter dans le temps pour faciliter la lecture par les analystes. Et finalement ce système ne suit pas une charte bien spécifique pour nommer les répertoires ce qui a engendré par conséquent une énorme hétérogénéité au niveau des noms d'un même répertoire. De l'autre côté, PcVue est limité par la complexité de fichiers.dat et la lourdeur du système. En effet, ce système permet d'afficher des courbes lisibles pour surveiller les consommations d'un bâtiment mais il n'est pas assez performant en terme temps de réponse. Egalement, actuellement, PcVue enregistre les données sur deux machines dont une principale et l'autre de secours pour assurer la pérennité des données en cas de coupure. Mais cette solution n'est pas fiable en terme d'analyse étant donné que PcVue permette d'afficher les courbes d'une seule machine à la fois. Par conséquent, lorsqu'il s'agit d'une coupure au niveau d'une machine, les données sont

enregistrées automatiquement dans l'autre machine mais elles ne sont pas remontées dans les courbes ce qui engendre un affichage avec des valeurs manquantes. Finalement, ce système fait une purge chaque jour et n'arrive pas à garder des traces de données à cause du grand volume de ses bases propriétaire. Et enfin, le schéma de la base de données Access n'est pas créé convenablement en termes de relations entre les tables, d'optimisations et de nombre de tables crées. Et en plus, il ne contient aucun lien avec les données de METASYS ou PcVue.

3.3 Analyse et spécification des besoins

3.3.1 Besoins fonctionnels et non fonctionnels

Dans notre processus de définition des besoins, nous avons débuté par un entretien avec le responsable électromécanique. C'est celui qui est chargé du projet. Il sera la première personne interrogée suivie des cadres supérieurs. Suite à l'analyse des réponses des interrogés, nous avons rassemblé les besoins fonctionnels, qui se classifient en deux niveaux majeurs ; la spécification des caractéristiques de l'entrepôt et la spécification des fonctionnalités de l'application d'interrogation de l'entrepôt [3] [9].

Les spécifications de l'entrepôt

L'entrepôt doit être conçu de manière qu'il permette de:

- Construire une source de données unique et non volatile.
- Mieux contrôler les données : nous pouvons organiser les données selon nos besoins et donc rendre la base de données plus optimale.
- Rendre la base plus évolutive et plus adaptée à nos futurs besoins.
- Garder la traçabilité de chaque donnée : source d'extraction de cette information.
- Avoir une modélisation multidimensionnelle des données et hiérarchisation de granularité d'analyse d'un niveau global à un niveau plus détaillé.
- Garantir une évolution et un rafraîchissement périodique et régulier des données.
- Gérer un grand volume de données variées.

Les spécifications de l'application

L'application doit être conçue de manière à permettre :

• La haute disponibilité des données à n'importe quel instant.

• La formulation des requêtes assez complexes permettant l'interrogation des données de l'entrepôt à n'importe quel niveau hiérarchique.

• La génération facile des rapports ad-hoc ou autre types de rapport.

• La consultation aisée des tableaux de bord fournis.

• L'application facile des différentes opérations sur les données.

3.3.2 Besoins non fonctionnels

• Définir une résolution et une quantité graphique adéquate.

• Fournir des interfaces interactives et compréhensibles manipulées facilement par les décideurs afin d'explorer leurs données.

• Fournir des tableaux de bord clairs et facilement analysables.

• Fournir des rapports lisibles tout en respectant la charte graphique.

• Assurer la sécurité et la performance de l'entrepôt.

4 Systèmes d'intégration pour l'analyse décisionnel

Face aux problèmes soulevés des systèmes de supervision du SGE et suite aux besoins déduits par la direction, nous avons opté pour la réalisation du projet en deux phases.

4.1 Solution d'intégration à court terme

4.1.1 Présentation

La première phase du projet consiste à modifier la sauvegarde du PcVue de sa base propriétaire vers SQL Server, à intégrer l'ensemble des données de METASYS sur une période de 6 mois dans la même base de PcVue. Finalement, nous construisons une source commune qui englobe l'ensemble des données des deux systèmes comme le montre la figure 6.

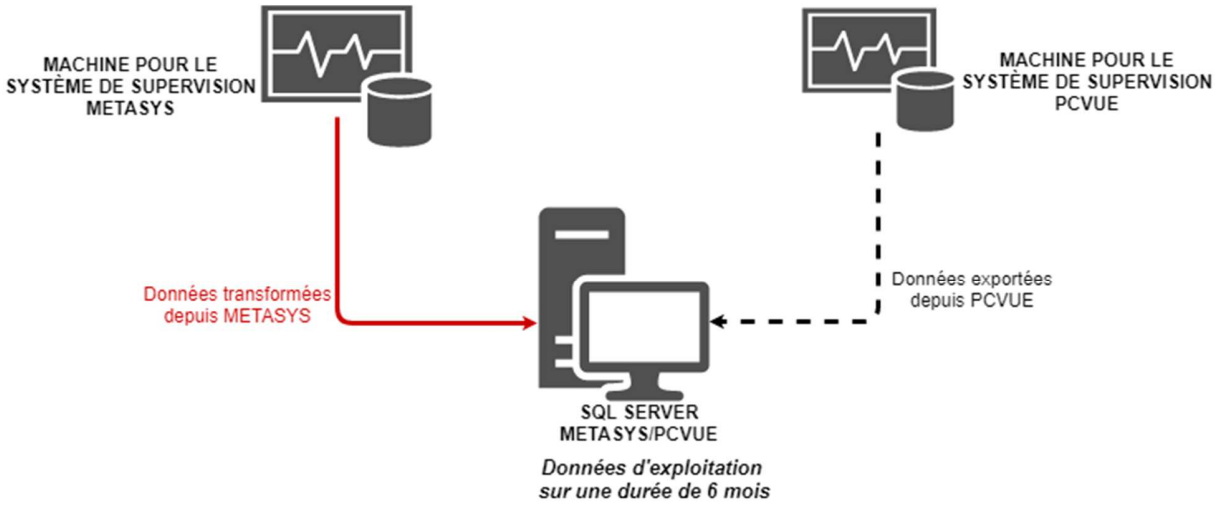


Figure 6: Schéma descriptif de la première partie du projet

4.1.2 Avantages

Cette solution présente plusieurs avantages présentés ci-dessous:

• Avoir une seule base de données au lieu de deux bases propriétaires assez complexes.

• Assurer la pérennité et la récupération de l'ensemble des données.

• Pouvoir appliquer les courbes et les analyses de PcVue sur les données de METASYS.

• Comparer les données de METASYS et PCVUE.

• Résoudre le problème des valeurs manquantes dans les courbes de PcVue causée par l'enregistrement des données sur deux machines distantes à travers la récupération des données depuis SQL Server.

• Faire des comparaisons et des analyses entre des sous systèmes différents de METASYS.

4.2 Solution d'intégration Big Data

4.2.1 Présentation

La deuxième phase du projet consiste à gérer l'historique des systèmes du SGE. Pour cette raison, nous avons décidé de créer un entrepôt de données orienté Big Data [4] [10] [11]qui peut supporter le grand volume de données. Cet entrepôt va contenir l'historique de METASYS, l'historique de PcVue, la base de données Access et finalement, les données de 6 mois enregistrées lors de la première phase sous SQL Server [7]. Cette intégration va nous permettre de brancher un outil d'analyse sur l'entrepôt afin d'obtenir des courbes et des tableaux de bord qui aident la direction à piloter et analyser son système énergétique. La figure 7 montre la nouvelle architecture de l'entrepôt Big Data [2].

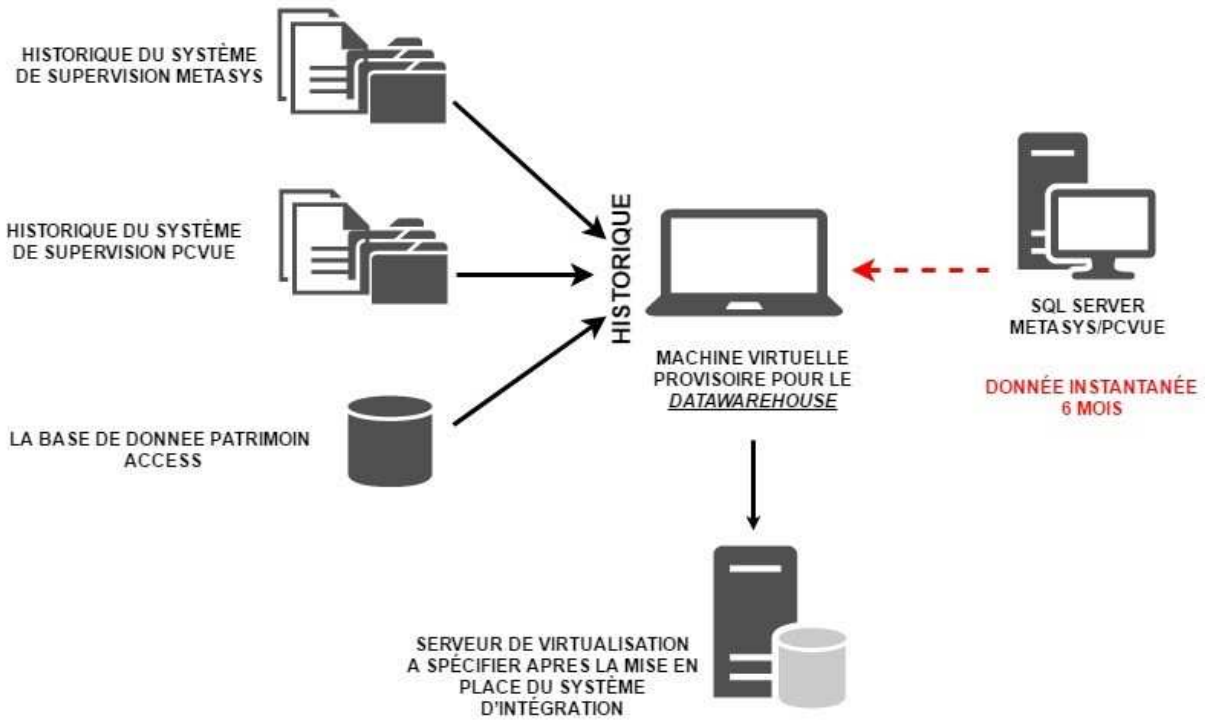


Figure 7: Schéma descriptif de la deuxième partie du projet

4.2.2 Avantages

Cette deuxième partie de la solution présente les avantages suivant:

- Se débarrasser des archives des bases propriétaires.
- Avoir un accès à un historique des données pour effectuer des comparaisons et des analyses.

- Avoir une source fiable, pertinente et solide.
- Stocker d'importants volumes de données de toute nature en temps réel.

5 Conclusion

Dans ce premier chapitre, nous avons défini le champ de notre étude suivi d'une étude de l'existant afin de préciser les objectifs à atteindre. En effet, l'étude de l'existant nous a permis de préparer une bonne conception pour les améliorations que nous allons ajouter dans la solution proposée afin de répondre à nos besoins. Dans le chapitre qui suit, nous présentons les démarches de développement et de conception de notre solution.

CHAPITRE 2 : L'informatique décisionnelle

1 L'informatique décisionnelle

Modélisation et intégration de données de capteurs/compteurs du SGE

1 Introduction

Ce chapitre sera réservé pour définir l'informatique décisionnelle. Nous présentons dans un premier temps ses avantages et ses limites. Nous abordons, ensuite, ses termes et les concepts clés en détaillant la notion d'ETL et d'entrepôt de données. Puis, nous développons les notions de Big Data et de bases de données NoSQL.

2 Concepts généraux du BI

2.1 La Business Intelligence

L'informatique décisionnelle, également Business Intelligence ou BI en anglais, désigne les moyens, les méthodes et les outils qui apportent des solutions en vue d'offrir une aide à la décision aux professionnels afin de leurs permettre d'avoir une vue d'ensemble sur l'activité de l'entreprise et de leurs permettre de prendre des décisions plus avisées à travers des tableaux de bord de suivi et des analyses.

2.1.1 Avantage du BI

Déployer une solution BI apporte de nombreux avantages :

- Améliorer la visibilité sur les chiffres, les écarts et les anomalies.
- La combinaison de plusieurs sources de données (ERP, systèmes comptable, feuilles de calcul, des budgets …).
- La présentation uniforme d'informations fiables.
- L'automatisation permettant l'accélération de la collecte et de la diffusion de l'information.
- La performance dans le calcul d'agrégats sur de gros volume de données.

• La prise de décision grâce à des indicateurs pertinents et à une structure cohérente des informations.

• L'aide à nettoyer les données présentes dans différents logiciels.

• L'anticipation des événements et la projection dans l'avenir.

2.1.2 Limites du BI

Parmi les limites de la Business Intelligence :

• La mise en place d'une solution de BI prend beaucoup du temps : de nombreuses entreprises dans le scénario industriel rapide ne sont pas assez patientes pour attendre la mise en place du système décisionnel dans leur organisation.

• Complexité : un autre inconvénient de BI pourrait être sa complexité dans la mise en œuvre des données.

• Erreur : les résultats produits par les systèmes décisionnels sont le résultat de conceptions informatiques et mathématiques complexes, qui peuvent révéler des erreurs, par ailleurs les résultats sont souvent statistiques, donc non déterministes. La possibilité d'une erreur ou d'une approximation inadaptée devra toujours être prise en compte dans les décisions.

2.2 Les principes des systèmes décisionnels

Le système décisionnel est architecturé de la façon suivante :

• Plusieurs sources de données en lecture.

• Un entrepôt de données fusionnant les données requises.

• Un ETL permettant d'alimenter l'entrepôt de données à partir des données existantes.

• Des magasins de données permettant de simplifier l'entrepôt de données.

• Des applications d'exploitation de données pour présenter l'étude aux utilisateurs finaux et décideurs.

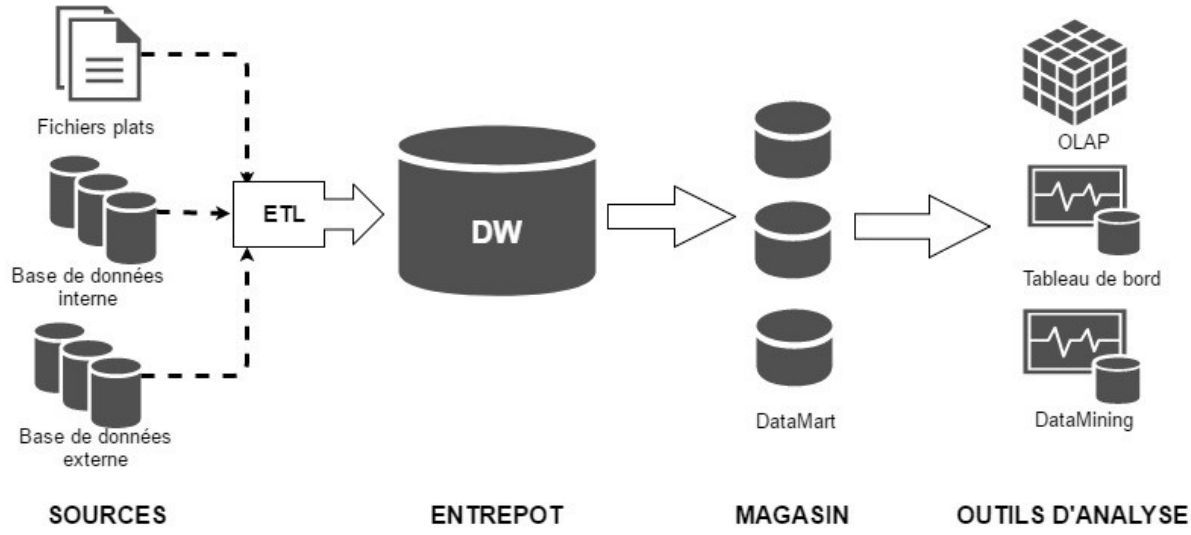


Figure 1: Architecture générale d'un système décisionnel

2.2.1 Sources de données

Afin d'alimenter l'entrepôt, les informations doivent être identifiées et extraites de leurs emplacements originels. Il s'agit des sources de données hétérogènes qui peuvent comporter des données internes à l'entreprise, stockées dans les bases de données de production des différents services. Elles peuvent être aussi des sources externes, récupérées via des services distants et des web services ou des sources qui peuvent être sous format de fichiers plats.

2.2.2 Entrepôt de données

D'après BILL Inmon: "Un entrepôt de données est une collection de données thématiques, intégrées, non volatiles et historiées, organisées pour la prise de décision.". D'après cette définition nous distinguons les caractéristiques suivantes :

• Données Orientées sujet : les données des entrepôts sont organisées par sujet et donc triées par thème.

• Données intégrées : les données provenant des différentes sources doivent être intégrées avant leur stockage dans l'entrepôt de données. Un nettoyage préalable des données est nécessaire afin d'avoir une cohérence et une normalisation de l'information.

• Données non-volatiles : à la différence des données opérationnelles, celles de l'entrepôt sont permanentes et ne peuvent pas être modifiées. Le rafraîchissement de

l'entrepôt, consiste à ajouter de nouvelles données sans perdre celles qui existent.

• Historiées : les données doivent être datées.

2.2.3 Magasin de données

Les magasins de données ou Datamarts sont un sous-ensemble complet et naturel de l'entrepôt de données. Ils sont structurés pour répondre rapidement aux sollicitations des utilisateurs. À l'inverse d'un entrepôt de données qui peut être très volumineux et qui ne permet pas une utilisation appropriée, ils ont en effet moins de données à gérer ce qui permet d'améliorer considérablement les temps de réponse.

2.2.4 Extract-Transform-Load

ETL, acronyme d'Extraction Transformation Loading, est un processus d'intégration des données [13] [14]. Il permet de transférer des données brutes d'un système source, de les préparer pour une utilisation en aval et de les envoyer vers l'entrepôt de données. Ce système doit faire passer les données par un tas de processus pour les dé-normaliser, les nettoyer, les contextualiser, puis de les charger de la façon adéquate. Cependant, la réalisation de l'ETL est une étape très importante et très complexe parce qu'il constitue 70% d'un projet décisionnel en moyenne.

3 La différence entre OLTP et OLAP

Les systèmes informatiques peuvent se subdiviser en deux catégories: les systèmes transactionnels OLTP (Online Transaction Processing) et les systèmes analytiques OLAP (OnLine Analytical Processing).

• Les systèmes OLTP sont dédiés aux métiers de l'entreprise pour les assister dans leurs tâches de gestion quotidiennes et donc directement opérationnels. Le mode de travail est transactionnel. L'objectif est de pouvoir insérer, modifier et interroger rapidement et en sécurité la base. Ces actions doivent pourvoir être effectuées très rapidement par de nombreux utilisateurs simultanément. Il est proposé essentiellement pour les application gérant des opérations commerciales comme les opérations bancaires, ou l'achat de bien divers.

• Les systèmes OLAP sont dédiés au management de l'entreprise pour l'aider au pilotage de l'activité. C'est un outil de reporting dont la couche d'analyse permet de générer les

des résultats en fonction du contenu d'un entrepôt de données. Les programmes consultent une quantité importante de données pour procéder à des analyses. Les objectifs principaux sont :: regrouper, organiser des informations provenant de sources diverses, les intégrer et les stocker pour permettre à l'utilisateur de retrouver et analyser l'information facilement et rapidement.

Bien que les systèmes d'informations OLTP et OLAP aient le point commun de regrouper les données de l'entreprise dans un SGBD et d'en fournir l'accès aux utilisateurs, ils présentent de profondes différences, présentées dans le tableau 0:

Caractéristiques	**OLTP**	**OLAP**
Utilisation	SGBD base de production	Entrepôt de données
Opération typique	Mise à jour	Analyse
Type d'accès	Lecture écriture	Lecture
Taille BD	Faible (max quelque GB)	Importante (pouvant aller à plusieurs TB)
Requête	Simples, régulières, répétitives, nombreuses	Complexes, irrégulières, peu nombreuses, non prévisibles
Quantité d'informations échangées	Faible	Important
Orientation	Ligne	Multi dimension
Structure de données	Beaucoup de tables	Peu de table mais de grande taille
Ancienneté des données	Récente	Historique

Table 1: Les différences entre OLTP et OLAP

4 Démarche de construction d'un entrepôt de données

L'entreposage de données se déroule en quatre phases principales [5] :

- Modélisation et conception de l'entrepôt.
- Alimentation de l'entrepôt.
- Mise en œuvre de l'entrepôt.
- Administration et maintenance de l'entrepôt.

4.1 Modélisation et Conception de l'entrepôt

Les approches les plus connues dans la conception des entrepôts sont :

- L'approche descendante qui est basée sur les besoins d'analyse.
- L'approche ascendante qui est basée sur les sources de données.
- L'approche mixte qui est une combinaison des deux approches.

4.1.1 Approche descendante

Dans cette approche le contenu de l'entrepôt sera déterminé selon les besoins de l'utilisateur final. Ainsi, les instructions sont données en amont et les objectifs du projet sont fixés par la direction. Toutes les attentes du chef de projet sont communiquées clairement à chaque participant au projet.

4.1.2 Approche ascendante

C'est une approche dont le contenu de l'entrepôt est déterminé selon les sources de données. C'est un processus analytique qui examine des données de base pour en tirer un schéma multidimensionnel offrant une vision analytique des données.

4.1.3 Approche mixte

C'est une approche hybride qui combine les approches ascendante et descendante. Elle prend en considération les sources de données et les besoins des utilisateurs.

4.2 Alimentation de l'entrepôt

Une fois l'entrepôt est conçu, il faut l'alimenter et le charger en données. Cette alimentation s'effectue à travers le processus ETL et se déroule en trois phases:

4.2.1 L'extraction des données

Il s'agit de la première étape de récupération des informations dans l'environnement de l'entrepôt de données. L'extraction comprend la lecture et la compréhension de la source de

données, ainsi que la copie des parties nécessaires à une exploitation ultérieure dans la zone de préparation. Ainsi, nous avons deux types d'extraction :

• Extraction complète : il s'agit d'une capture de données à un certain temps. Elle est employée dans deux situations à savoir le chargement initial des données ou le rafraîchissement complet des données en cas d'une modification de source par exemple.

• Extraction incrémentale : il s'agit de capturer uniquement les données qui ont changé ou ont été ajoutées depuis la dernière extraction. Nous distinguons alors deux manières pour faire l'extraction incrémental.

- Extraction temps-réel qui s'effectue au moment où les transactions surviennent dans les systèmes sources.

- Extraction différée qui extrait tous les changements survenus durant une période donnée (ex: heure, jour, semaine, mois) à posteriori.

4.2.2 La transformation des données

Une fois que les données sont extraites dans la zone de préparation nous appliquons plusieurs étapes de transformations qui ont pour but de rendre les données cibles homogènes afin qu'elles puissent être traitées de façon cohérente :

• Résolution des cas d'informations manquantes et conversion en format standard.

• Combinaison des sources de données par mise en correspondance exacte avec des valeurs clé ou par mise en correspondance approximative d'attributs hors clé et y compris la recherche d'équivalents textuels des codes des systèmes sources.

• Construction d'agrégats pour optimiser les performances des requêtes les plus courantes.

• Application de filtres.

4.2.3 Le chargement des données

C'est la dernière phase de l'alimentation d'un entrepôt de données, le chargement est une étape indispensable. Elle reste toutefois très délicate et exige une certaine connaissance des structures du système de gestion de la base de données afin d'optimiser au mieux le processus. Nous distinguons trois types de chargement :

• Chargement initial : se fait une seule fois lors de l'activation de l'entrepôt de données.

• Chargement incrémental : se fait une fois le chargement initial complété et peut se faire en temps réel ou en lot.

• Chargement complet : est employé lorsque le nombre de changements rend le chargement incrémental trop complexe.

4.3 Administration

Cette étape est constituée de plusieurs tâches pour assurer :

• La qualité et la pérennité des données aux différents applicatifs.

• La maintenance et le suivi.

• La gestion de configuration.

• La gestion de l'évolution et les demandes d'expansion.

• L'organisation et l'optimisation du SI.

• La documentation et les formations.

4.4 Restitution

C'est la dernière étape d'un projet d'entreposage de données, soit son exploitation. L'exploitation de l'entrepôt se fait par le biais d'un ensemble d'outils analytiques développés autour de ce dernier. Il s'agit de regrouper tout ce qui a attrait à la représentation et la transmission des résultats d'analyse de données. Le principe de la restitution, donc, est d'agréger et de synthétiser des données nombreuses et complexes sous forme d'indicateurs, de tableaux, de graphiques permettant d'en avoir une appréhension globale et simplifiée pour faire toutes les analyses nécessaires. 1.2 Nous avons défini dans cette partie les notions de bases des systèmes décisionnels. Dans la partie suivante nous allons introduire les notions du Big Data et les bases de données NoSQL.

5 Le Big Data

5.1 Définition

Littéralement, le Big Data signifie méga données, masses de données ou encore données massives. Ces termes désignent un ensemble très volumineux de données qu'aucun outil classique de gestion de base de données ou de gestion de l'information ne peut vraiment traiter. Pour faire face à ces énormes volumes de données, de nouvelles technologies sont apparues comme Hadoop, MapReduce ou les bases de données NoSQL (Not only SQL). Le Big Data couvre quatre caractéristiques de base : volume, vélocité, variété et véracité.

• Volume : Les entreprises sont submergées de volumes de données croissants de tous types, qui se comptent en téraoctets, pétaoctets voire en zettaoctets.

• Vélocité : La vélocité, ou vitesse, fait référence à l'énorme rapidité avec laquelle les données sont générées et doivent être traitées.

• Variété : Le Big Data se présente sous la forme de données structurées ou non structurées (texte, données de capteurs, son, vidéo, fichiers journaux, etc.).

• Véracité : La notion de véracité met en avant la dimension qualitative des données nécessaire au fonctionnement des outils de Big Data.

5.2 Les bases des données NoSQL

Le nombre de bases NoSQL est important. Il est impératif de connaître leurs différences pour adopter la bonne technologie pour notre besoin.

5.2.1 Définition

Le NoSQL (Not only SQL) désigne une catégorie de base de données qui s'oppose à la notion relationnelle des SGBDR et qui a pour objectif de proposer une alternative au langage SQL et au modèle relationnel de bases traditionnelles. Le NoSQL englobe une gamme étendue de technologies et d'architectures afin de résoudre les problèmes de performances en matière d'évolutivité et de Big Data que les bases de données relationnelles ne sont pas conçues pour affronter.

5.2.2 La différence entre SQL et NoSQL

Le tableau 1 représente la différence entre les bases de données SQL (Structured Query Language) et les bases de données NoSQL :

	SQL	**NoSQL**
Stockage de données	Les données sont stockées dans un modèle relationnel, représentées dans des tables avec des lignes et des colonnes.	Les données sont stockées sous forme de document, graphe, clé-valeur ou colonne.
Schémas et flexibilité	Le schéma des données doit être prédéfini explicitement au niveau du serveur c'est-à-dire les colonnes doivent être décidées et définies avant la saisie des données et chaque enregistrement doit être conforme au schéma fixé.	Les schémas sont dynamiques. Une base NoSQL peut en effet croître sans contrainte, sa structure organisationnelle n'est pas liée à un schéma relationnel difficile à modifier.
Évolutivité	Les bases de données SQL sont verticalement évolutives. La mise à l'échelle nécessite l'augmentation de la charge en modifiant la CPU et la RAM.	Les bases de données NoSQL sont évolutives horizontalement. La mise à l'échelle se déroule entre les serveurs. Il s'agit d'ajouter simplement des serveurs de plus dans l'infrastructure de base de données.
Application	Utilisé dans les projets qui exigent des données discrètes qui peuvent être identifiées à l'avance et dont l'intégrité des données est	Utilisé dans les projets qui exigent des données indépendantes, indéterminées ou évolutives.

	essentielle.	
Relation	Les requêtes SQL offrent une puissante clause JOIN. Nous pouvons obtenir des données connexes dans plusieurs tables. Les données ne doivent pas être redondantes et elles doivent respecter l'intégrité référentielle.	NoSQL n'a pas d'équivalent de JOIN. Il est possible d'avoir des données redondantes.

Table 2: La différence entre SQL et NoSQL

5.2.3 Schémas de données dans les bases NoSQL

Les solutions NoSQL existantes peuvent être regroupées en quatre grandes familles:

• Clé-valeur : Les données sont simplement stockées sous la forme d'un couple clé/valeur. Les solutions les plus connues sont Redis, Riak et Voldemort créé par LinkedIn. Ces moteurs offrent des fonctionnalités simplifiées avec une moins grande richesse fonctionnelle en termes de requêtes et d'excellente performance grâce à leur modèle d'accès simplifié.

• Les moteurs orientés colonnes [10] : Ce modèle implémente une structure proche d'une table. Les données sont représentées en lignes et séparées par des colonnes qui sont variables. En effet, le nombre de colonnes peut varier d'un enregistrement à un autre. Ce modèle est plus utilisé pour des volumétries importantes. Il existe comme solution Hbase ainsi que Casandra.

• Les moteurs orientés documents [11] : Cette famille est une extension de la famille clé/valeur en associant une clé à un document hiérarchique comme le XML, le JSON (JavaScript Object Notation). Pour ce modèle, les implémentations les plus populaires sont CouchDB d'Apache et MongoDB.

• Orienté graphe : Ce modèle de représentation des données se base sur la théorie des graphes. Il s'appuie sur la notion de nœuds, de relations et de propriétés qui leur sont rattachées. Ce modèle facilite la représentation du monde réel, ce qui le rend adapté au

traitement des données des réseaux sociaux par exemple. La principale solution est Neo4J.

6 Conclusion

Dans ce chapitre, nous avons détaillé toutes les notions relatives aux systèmes décisionnels, au Big Data et aux base de données NoSQL pour les maîtriser afin de favoriser le bon déroulement du projet.

CHAPITRE 3 : Conception

1 Conception

Modélisation et intégration de données de capteurs/compteurs du SGE

1 Introduction

Au cours de ce chapitre, nous présentons dans un premier temps la structure de la base de donnée SQL Server contenant les données de chaufferie sur 6 mois. Puis nous présentons la conception multidimensionnelle de l'entrepôt de données Big Data.

2 Sources de données

Dans cette partie, nous présentons les diagrammes de classe de nos sources de données afin de construire un digramme de classe finale qui défini la source globale. Ces schémas ont été définis après application d'une phase en retro ingénierie (reverse engineering). En effet, nous avons dégagé trois schémas pour nos source METASYS, PcVue et Access.

2.1 Source METASYS

Notre source METASYS est présentée sous forme de hiérarchie de répertoires. Ainsi, elle est sous forme de répertoires qui représente les bâtiments et les locaux_techniques. Ces derniers sont composés de sous-répertoires indiquant les capteurs. Finalement, ces capteurs sont composés de mesures englobant les valeurs de capteurs. La figure 1 représente le schéma de METASYS.

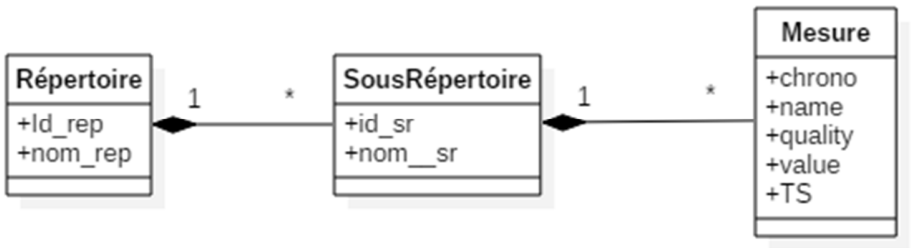


Figure 1: Schéma source du système METASYS

2.2 Source PcVue

Notre source de PcVue est représentée ,sous le système, sous la forme de hiérarchie de répertoires. Ainsi, elle est composée des points qui sont représentés par la classe variable. Cette classe est composée de répertoires qui indiquent les bâtiments et les locaux_techniques. La classe SousRepertoire indique dans notre contexte les équipements qui sont composés de sousSousrépertoires présentant les capteurs. Finalement, ces capteurs sont composés de mesures englobant les valeurs de capteurs. La figure 2 représente le schéma du PcVue.

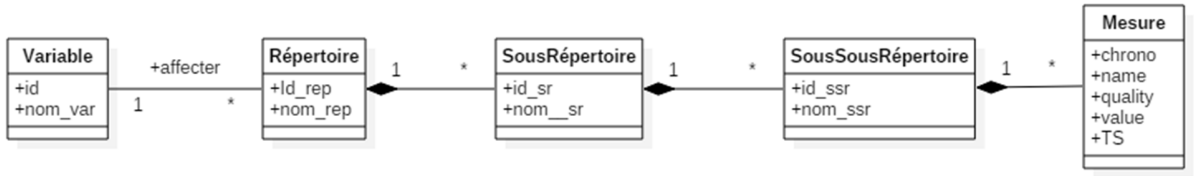


Figure 2: Schéma source du système PcVue

2.3 Source globale

Avant de commencer la conception de notre schéma multidimensionnel, nous présentons le diagramme de classe de notre source globale. Une analyse de ces tables a permis d'établir le diagramme de classe suivant afin de pouvoir appliquer notre démarche ascendante. Notre source est composée de six tables qui sont reliées entre elles par un lien de composition comme le montre la figure 3.

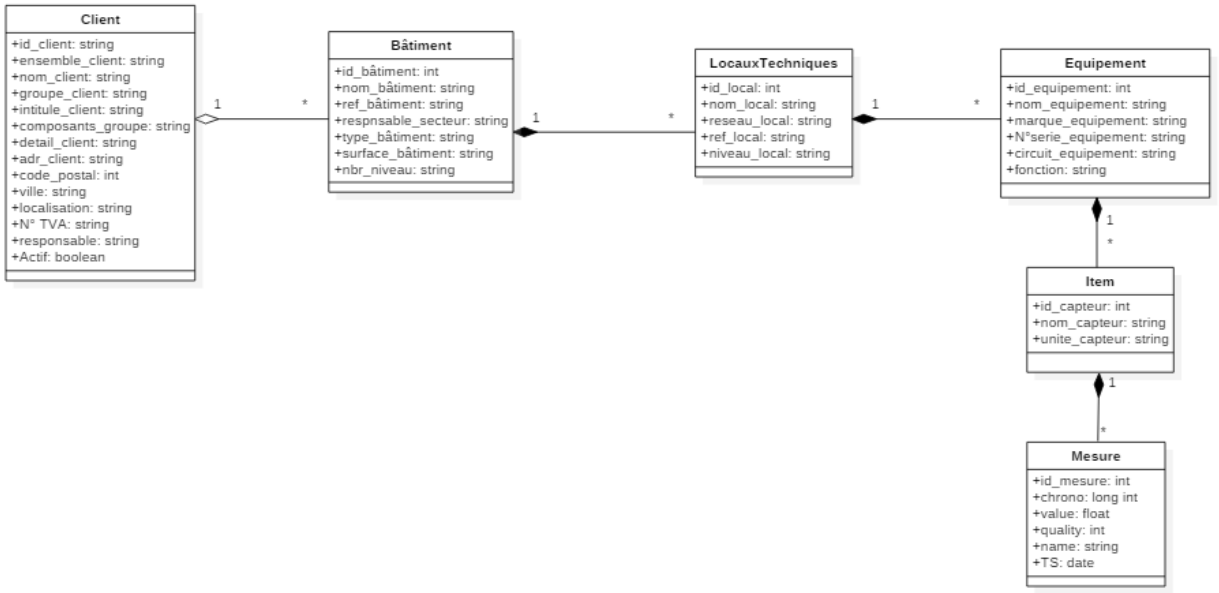


Figure 3: Schéma relationnel de la source de donnée

Ce digramme de classe décrit la source Access du patrimoine du SGE d'une part, et d'autre part les classes des systèmes de supervision METASYS et PCVUE représentés dans ce schéma à travers la table **Mesure**. En effet, elle permet de suivre les différentes mesures générées par les capteurs et les sources d'où quelles proviennent. Ainsi, la classe **Mesure** avec ses attributs id_mesure, chrono, name, value, quality, TS et type_capteur permet d'historiser la liste des valeurs générées par des milliers de capteurs de notre société. Ces capteurs sont modélisés par la classe **Item** qui fixe le nom du capteur et son unité de mesure. Chaque item est installé dans un équipement (Classe **Equipement**) qui représente des composants dans les locaux techniques. Ainsi, la table **LocauxTechniques** qui est caractérisée par les champs nom_local, reseau_local, ref_local et niveau_local décrit les Sous_Stations de chauffage gérées par le SGE. Elles sont différenciées en plusieurs types : Primaire, Secondaire, LT (Local Technique) et CTA (Central de Traitement d'Air). Les locaux techniques appartiennent aux **Bâtiments** desservis par le SGE et liés aux clients. La table **Client** décrit alors les différents clients du SGE, avec un détail sur l'ensemble scientifique auquel ils appartiennent, le sous_ensemble des structures ainsi que leur localisation

3 Conception du schéma multidimensionnel

Les méthodes de conception d'un schéma multidimensionnel sont:

- Méthodes descendantes (Top-down).
- Méthodes ascendantes (Bottom-up).
- Méthodes mixtes.

3.1 Méthode ascendante

Cette méthode propose un ensemble d'étapes pour la définition des faits, des dimensions et des hiérarchies à partir du schéma de la source.

3.1.1 Détermination des faits

Cette étape consiste à détecter les classes représentatives de l'analyse. Une classe représentative (CR) décrit un événement qui se produit à un instant donné et elle contient les mesures d'analyse.

Pour notre application, la classe la plus active et qui représente un événement daté est la classe **Mesure**. Cette classe représente l'activité des capteurs. A partir de cette dernière, nous définissons le fait (F_mesure) englobant les mesures: "value" permettant de tracer les valeurs générées par les capteurs, "Name" le nom du capteur, "chrono" le nombre en milliseconde de l'événement depuis 1970, "quality" indique la qualité de la mesure, "type_capteur" précisant s'il s'agit d'un compteur, sonde, vanne motorisée ou débitmètre et la dernière mesure "consommation" c'est une valeur à calculer à partir de deux valeurs de value en fonction du type du capteur.

F1: **F_Mesure** = { value=S.Mesure.value (sum, avg, max, min), consommation = (s.Mesure.value1-S.mesure.value2) }.

3.1.2 Détermination des dimensions

Détection des classes déterminantes des classes représentatives

Une classe Ci est déterminante d'une classe CR si et seulement si Ci=Cr ou Cr est relié à Ci par un lien de dépendance fonctionnelle directement ou indirectement. Le graphe suivant(Fig 4) représente les liens de dépendances fonctionnelles entre les classes représentatives et les autres classes de la source de données.

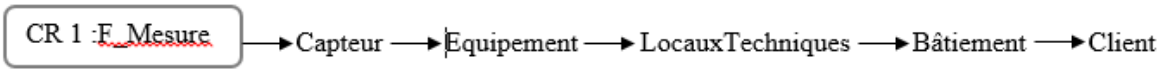


Figure 4: Graphe des liens de dépendances fonctionnelles

Après une analyse sémantique, nous avons dégagé la dimension "D_Item" qui regroupe les paramètres d'analyse et la dimension "D_temps" présentées ci-dessous :

D_Item { id_capteur=S.Item.id_capteur, nom_capteur = S.Item.nom_capteur, unité = S.Item.unite_capteur, id_client = S.Client.id_ client, ensemble_client = S.Client.ensemble_ client, nom_ client = S.Client.nom_ client, groupe = S.Client.groupe_ client, intitulé = S.Client.intitule_ client, Comp_groupe = S.Client.composant_groupe, détail = S.Client.détail_ client, adresse = S.Client.adr_ client, CodePostal = S.Client.code_postal, ville = S.Client.ville, localisation = S.Client.localisation, NTVA = S.Client.NTVA, responsable = S.Client.responsable, actif = S.Client.Actif, id_bâtiment = S.Bâtiment.id_ bâtiment, ref_b = S.Bâtiment.Ref_ bâtiment, nom_ b = S.Bâtiment.nom_ bâtiment, resp_secteur = S.Bâtiment.responsable_secteur, type = S.Bâtiment.type_ bâtiment, surface = S.Bâtiment.surface_ bâtiment, nbr_niveau = S.Bâtiment.nbr_niveau, id_LT = S.LocauxTechniques.id_local, nom_local = S.LocauxTechniques.nom_local, réseau = S.LocauxTechniques.reseau_local, num_ss = S.LocauxTechniques.num_ss, type = S.LocauxTechnique.type, id_equipement = S.Equipement.id_equipement, nom_equipement = S.Equipement.nom_equipement, num=S.Equipement.num_equipement, marque = S.Equipement.marque_equipement, numSérie = S.Equipement.Nsérie, fonction = S.Equipement.fonction_equipement, circuit = S.Equipement.circuit_equipement }

D_Temps {Timestamp=S.Mesure.TS}

Définition de la dimension temporelle

En suivant la démarche ascendante, nous proposons d'adopter à ce niveau-là la granularité définie au niveau de la source. La dimension temps est basée sur l'attribut source TS stockant la date de chaque mesure du capteur et qui a le format 'DateTime'. Dans notre contexte d'entrepôt de données, nous n'avons pas besoin de créer une dimension temps destinée à historiser un enregistrement pour chaque minute ou chaque seconde d'une période entière. Nous

avons donc besoin de la granularité heure, c'est pourquoi nous l'avons définie comme la granularité d'analyse.

DTemps : { codeDate = S. Mesure.codeDate, Heure = S. Mesure.Heure, Jour = S. Mesure.jour, Mois = S.Mesure.Mois, Année = S. Mesure.annee }

3.1.3 Hiérarchisation des dimensions et définition de la granularité de l'analyse

A ce stade-là, nous définissons les hiérarchies complètes en fonction du schéma de la source ainsi que le niveau le plus détaillé d'analyse.

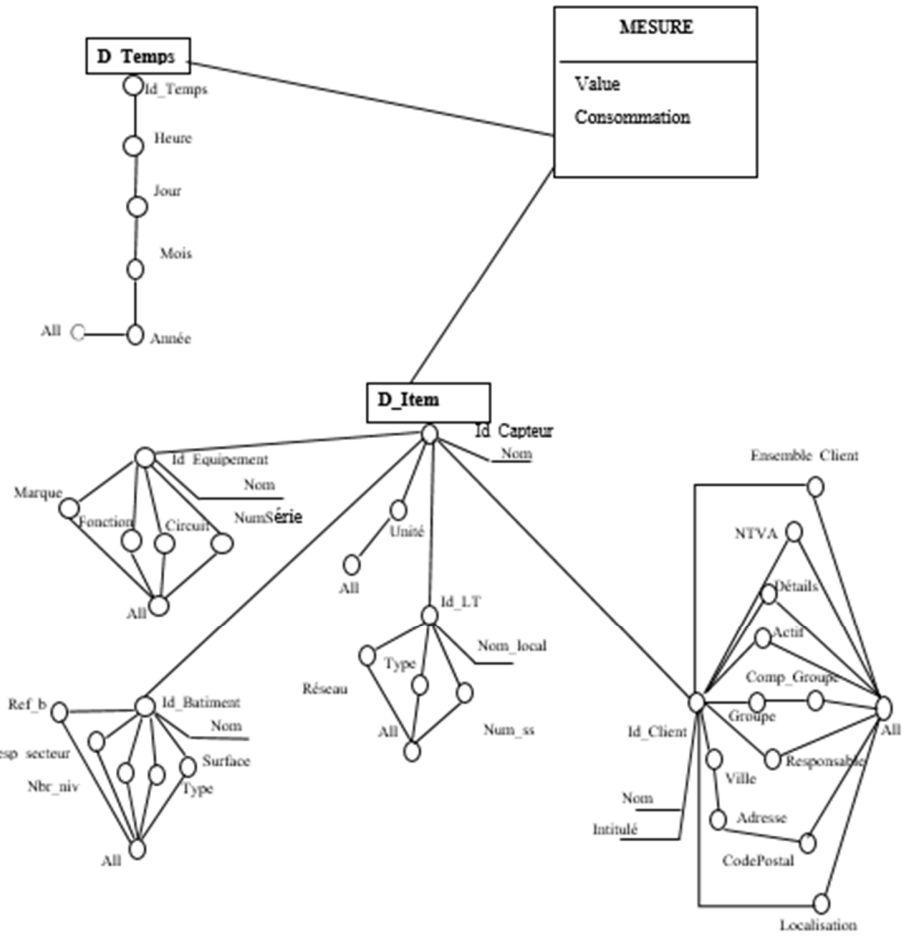


Figure 5: Schéma multidimensionnel en constellation (Démarche ascendante)

3.2 Démarche descendante

La démarche descendante commence par la collecte des données ensuite la spécification des besoins à travers la matrice des besoins, et enfin, la formalisation des besoins en créant le schéma multidimensionnel.

3.2.1 Collecte des données

Dans cette étape, nous collectons les requêtes-types pertinentes en interviewant les décideurs et nous élaborons un questionnaire permettant de mieux caractériser et identifier les besoins des décideurs.

Requêtes-types

- **R1** : **Analyser** la consommation **en fonction** des équipements, des capteurs et des heures.
- **R2** : **Analyser** la consommation **en fonction** des bâtiments, des capteurs et des mois.
- **R3** : **Analyser** la consommation **en fonction** des clients, des capteurs et des années.
- **R4** : **Analyser** la consommation **en fonction** locaux techniques, des capteurs et des mois.
- **R5** : **Analyser** les valeurs de mesures **en fonction** des clients, des locaux techniques, des bâtiments, des équipements, des capteurs et du temps.

3.2.2 Spécification des besoins

Il s'agit de créer la matrice des besoins en fonction de nos requêtes crées auparavant afin d'en tirer les dimensions et les faits.

Paramètres/ Indicateurs	Id_equip	Id_capt	heure	Id_bat	mois	Id_clt	année	Id_LT
Val_cons	X	X	X	X	X	X	X	X
val_mesure	X	X	X	X	X	X	X	X

Table 1: Matrice des besoins

3.2.3 Formulation des besoins

Dans cette partie, nous commençons tout d'abord, par la définition des faits, puis la définition des dimensions et l'enrichissement, ensuite, nous finissons par la création du schéma multidimensionnel.

Définition des faits

F1 : F_Consommation { Val_cons (sum, avg, max, min), Val_mesure (sum, avg, max, min) }

Définition des dimensions et enrichissement

D1 : D_Equipement { Id_Equipement,nom }

D2 : D_Capteur { Id_Capteur,nom }

D3 : D_Temps { année,mois,jour,heure }

D4 : D_Batiment { Id_Batiment,nom }

D5 : D_Client { Id_Client,nom }

D6 : D_LocauxTechniques { Id_LocauxTechniques,nom }

Définition des dimensions et enrichissement

3.2.4 Construction du schéma multidimensionnel

La figure 6 représente le schéma multidimensionnel en suivant la démarche descendante.

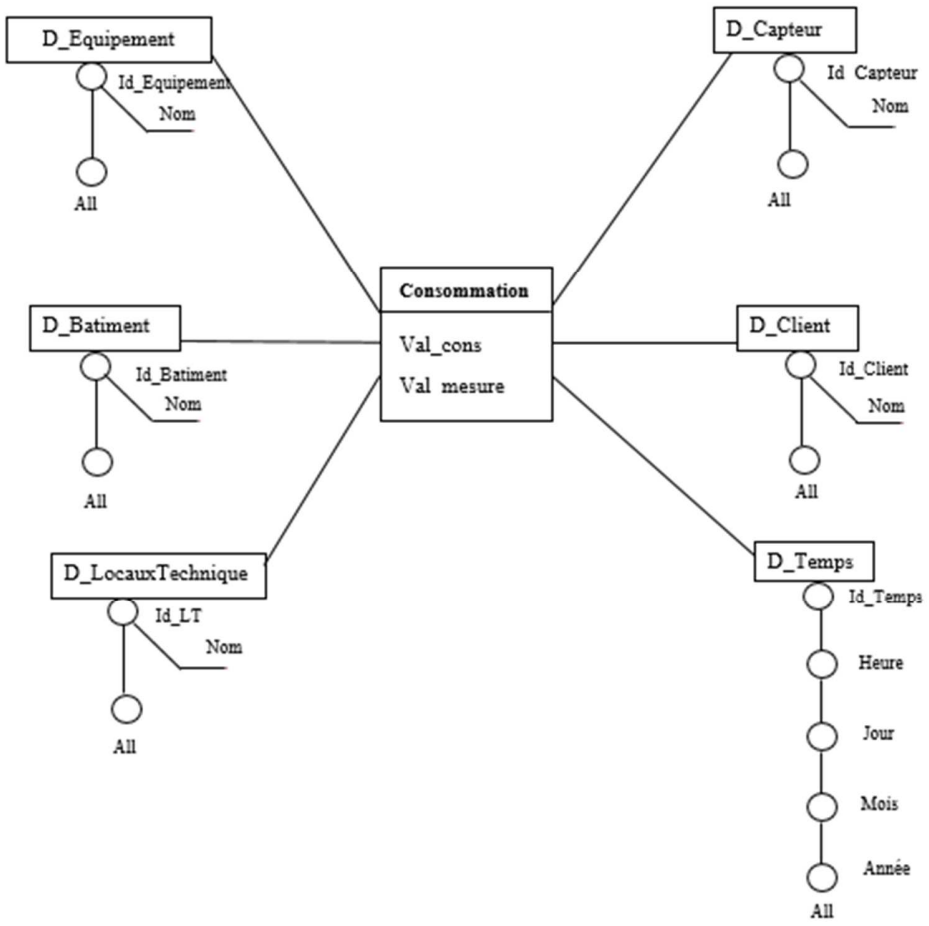


Figure 6: Schéma multidimensionnel en constellation (Démarche descendante)

3.3 Démarche mixte

Nous procédons maintenant à la combinaison des deux approches ascendante et descendante ; c'est là où nous dégageons les différences des besoins exprimées par les décideurs de celles extraites de la source de données. A ce stade, nous attribuons des ajustements, la correction de la granularité de l'analyse et la suppression des dimensions comme montre la figure ci-dessous. Après confrontation des deux schémas, nous jugeons que les dimensions client, bâtiment, Locaux Techniques, équipements doivent être supprimées parce que toutes ces tables sont reliées par une clé étrangère avec la table capteur.

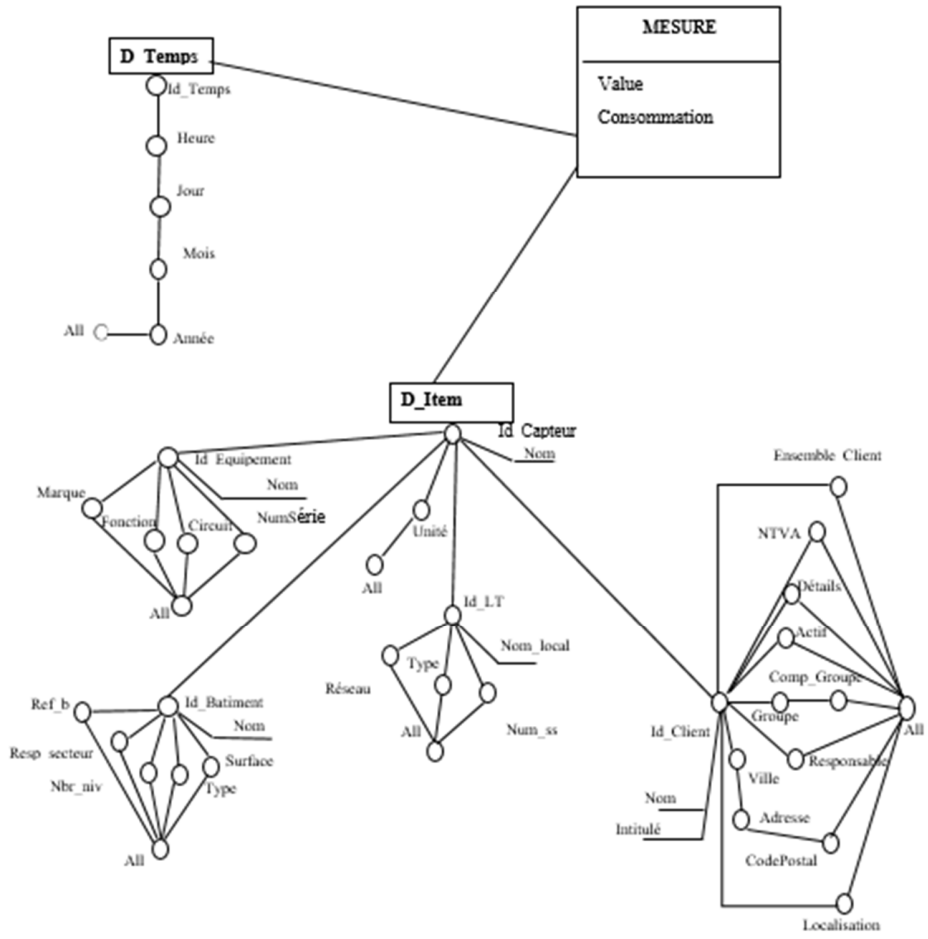


Figure 7: Schéma multidimensionnel en constellation (Démarche mixte)

4 Schéma logique Big Data

Étant donné que les historiques des systèmes de supervision METASYS et PcVue ainsi que la base de données Sql Server qui contient les données d'exploitation d'une durée de 6 mois soient très volumineux, nous avons opté de concevoir une solution Big Data pour remédier à ce problème de volumétrie de données.

Dans le contexte du Big Data une panoplie d'outils est offerte présentant chacune un ensemble d'avantages et de limites. Notre objectif est d'offrir une architecture décisionnel qui intègre ces outil afin d'offrir une solution répondant d'un côté, aux besoins d'analyse OLAP et de l'autre les contraintes de volumétrie, de variété et de vélocité.

4.1 Choix du modèle NoSQL

Traditionnellement, les données des systèmes d'analyse OLAP sont gérées par des bases de données relationnelles, mais avec le problème de gros volume de donnée il devient difficile de les gérer. Pour cette raison, nous avons choisi comme alternative d'utiliser les systèmes NoSQL (Not-OnlySQL) pour créer notre cube OLAP et plus précisément nous avons choisi les bases de données orientées-colonnes parce qu'elles correspondent d'une part à des besoins Big Data où il y a un volume très important de données et d'autre part elles sont assez proche conceptuellement des tables relationnelles.

La structure des bases orientées colonnes est modélisée selon BigTable, la base de données de Google. Une table comporte des clés, souvent appelées rowkeys, les clés de lignes. À l'intérieur de la table, des familles de colonnes sont définies et regroupent des colonnes. La famille de colonnes est prédéfinie, et nous lui attribuons souvent des options, alors que les colonnes qui s'y trouvent ne sont pas prédéfinies, c'est-à-dire qu'il n'y a pas de description de schéma à l'intérieur d'une famille de colonnes. D'une ligne à l'autre, les colonnes présentes dans une famille peuvent varier selon les données à stocker.

Dans cette partie, nous définissions les règles pour convertir notre schéma en étoile en modèle logique NoSQL orienté colonne et nous fixons les outils ainsi que l'architecture pour exploiter notre schéma multidimensionnel en Big Data avec le moteur d'analyse Kylin sur Hadoop.

4.2 Choix de l'outil d'analyse

4.2.1 Apache Kylin

C'est un moteur d'analyse distribué open source conçu pour fournir une interface SQL et une analyse multidimensionnelle (OLAP) sur Hadoop prenant en charge des jeux de données extrêmement volumineux. Kylin [2] s'associe étroitement avec Hadoop MapReduce comme outil de calcul des agrégations, Hive comme source de données et HBase pour le stockage.

4.2.2 Principe et Architecture

Le principe est de créer un modèle logique (dimensions / mesures) à partir d'un schéma en étoile dans Hive. Kylin créera ensuite des agrégats de cubes en utilisant MapReduce et mettra

[2] http://kylin.apache.org/

les agrégats et les métadonnées de cube dans HBase. Nous pouvons ensuite interroger les données du cube via l'interface utilisateur Kylin ou un outil de BI. L'architecture du moteur Kylin se base sur trois axes comme le montre la figure 8:

• Identification d'un schéma en étoile sur Hadoop(Hive)

• Création d'un Cube à partir des tables Hive et stockage des métadonnées et des agrégations du cube dans Hbase

• L'analyse des résultats avec des outils de visualisation via ODBC, JDBC ou RESTful API.

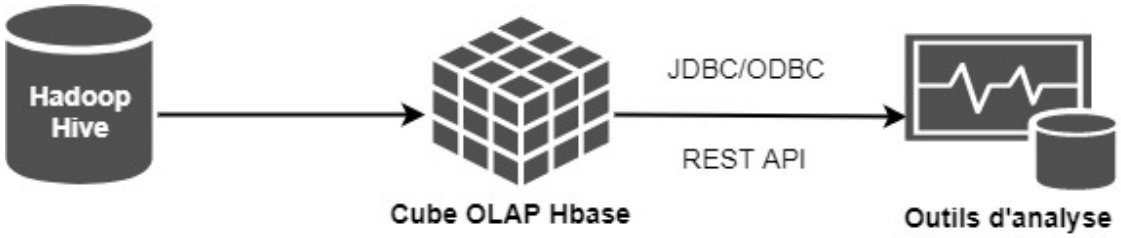


Figure 8: Architecture du Big Data

4.3 Structure du schéma multidimensionnel dans Hbase

Nous allons présenter dans cette partie les règles de correspondance avec le niveau conceptuel et la structure du schéma multidimensionnel dans Hbase. Les éléments (faits, dimensions) du modèle conceptuel multidimensionnel doivent être transformés en éléments du modèle NoSQL orienté colonnes de la façon suivante]:

• Le schéma en étoile conceptuel est transformé en une table.

• Le fait Fi est transformé en une famille de colonnes de cette table dans laquelle chaque mesure mi est une colonne.

• Chaque dimension Di est transformée en une famille de colonnes où chaque attribut de dimension Ai (paramètres et attributs faibles) est transformé en une colonne.

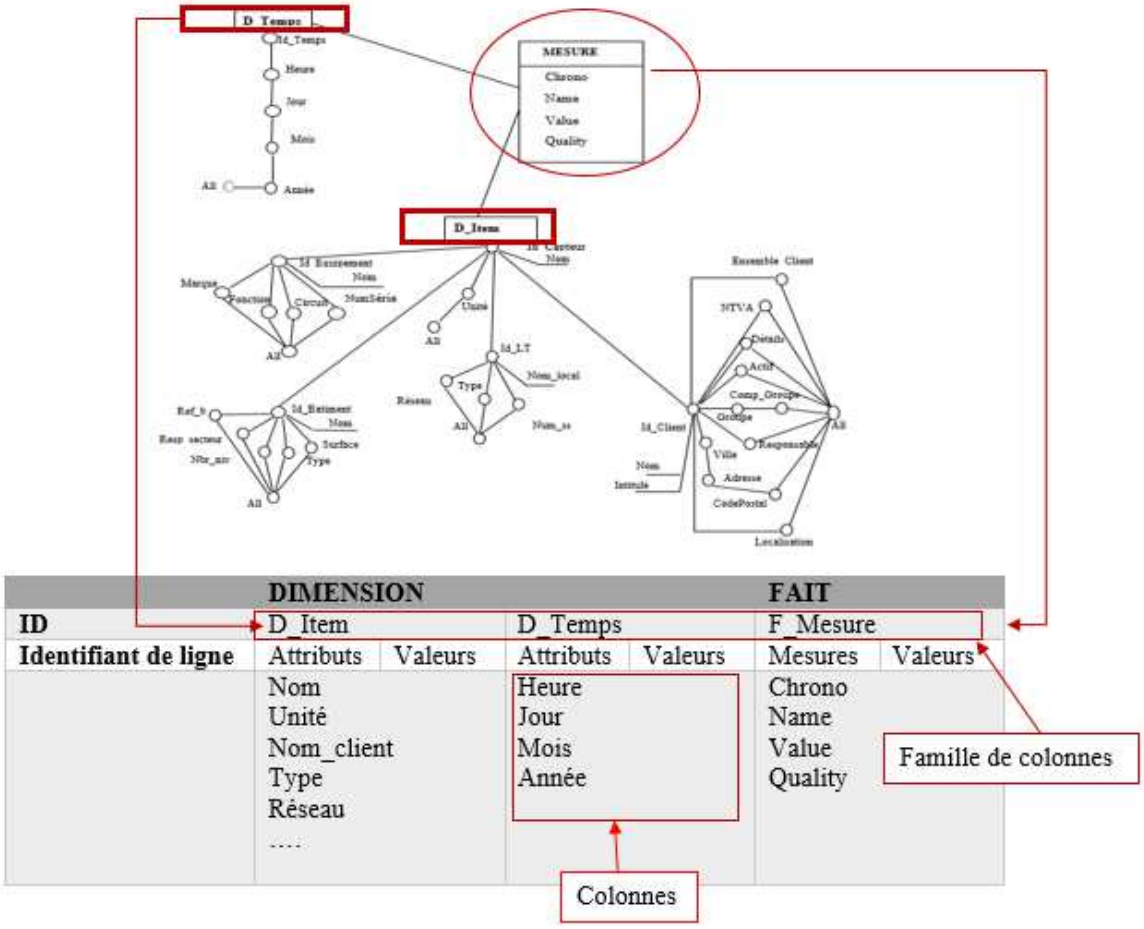


Figure 9: Transformation du modèle au modèle logique NoSQL orienté colonnes

La figure 9 représente notre schéma multidimensionnel conceptuel implanté dans une table Hbase. Le fait "F_Mesure" et ses dimensions " D_item" "D_temps" sont implantés dans trois familles de colonnes. Chaque famille de colonne contient un ensemble de colonnes correspondent soit à des attributs de dimensions, soit à des mesures du fait.

5 Conclusion

Après avoir appliqué les différentes démarches de la modélisation conceptuelle de l'entrepôt il ne reste qu'à le construire. Les étapes de cette construction feront l'objet du chapitre suivant.

CHAPITRE 4 : Conception et Développement de l'ETL

1 Conception et Développement de l'ETL

Modélisation et intégration de données de capteurs/compteurs du SGE

1 Introduction

Dans ce chapitre, nous présentons tout d'abord la conception et le développement de l'ETL pour construire l'entrepôt de données pour les données d'exploitation d'une durée de 6 mois puis nous définissons les différentes phases de la construction de l'entrepôt de données Big Data en commençant par la modélisation logique, physique pour aboutir à la phase d'intégration des données où nous présentons la conception et le développement de l'ETL. Afin de réaliser la phase d'intégration de données des deux systèmes de supervision METASYS et PcVue, nous avons exploité les outils de développement présentés ci-dessous.

2 Environnement et langage de développement

2.1 Environnement logiciel

Avant de commencer notre travail, nous avons mené une étude sur quelques solutions open source pour choisir l'outil qui convient à notre besoin. Plusieurs solutions ETL open source existent sur le marché. Nous avons choisi de faire une comparaison entre les deux outils les plus utilisés à savoir; Talend Open Studio (TOS) et Pentaho Data Integration (PDI). Pentaho Data Integration est l'ETL de la suite décisionnelle Open Source Pentaho. La principale différence est que Pentaho est utilisé pour les tâches ELT (Extract Load Transform) tandis que Talend fait partie d'un écosystème complet de gestion de données. Pentaho est plus facile à utiliser mais aussi moins complet par rapport à Talend. Contrairement à Talend Open Studio,qui est un générateur de code java ou perl. Pentaho Data Integration est un « moteur de transformation » : les données

traitées et les traitements à effectuer sont parfaitement séparés. L'avantage de Talend est que le code généré par Java peut être modifié pour obtenir plus de contrôle et de flexibilité. Également, Talend comporte plusieurs composants de SGBDR, NoSQL et Big Data nous pouvons alors se connecter aux principaux SGBD (Oracle, DB2, MS SQL Server, PostgreSQL, MySQL,...) et traiter tous les types de fichiers plats (CSV, Excel, XML).

Les deux outils sont fiables, performants et conviviaux mais étant donné que Talend est une plate-forme de gestion des données plus générale et complète, nous avons opté de l'utiliser dans la partie intégration.

Dans ce qui suit nous décrivons la suite d'outil utilisés dans notre architecture.

2.1.1 Talend Open Studio (TOS)

Talend Open Studio [3] est un ETL du type « générateur de code ». Pour chaque traitement d'intégration de données, un code spécifique est généré, ce dernier pouvant être en Java ou en Perl. Talend Open Studio utilise une interface graphique, le « Job Designer » (basée sur Eclipse RCP) qui permet la création des processus de manipulation de données. Il permet également l'intégration dans les suites décisionnelles Open Source (SpagoBI et JasperIntelligence).

2.1.2 SQL Server 2014

PcVue utilise un moteur d'archivage de données pour enregistrer des valeurs (évènements, tendances) en utilisant divers mécanismes et formats :

- Données sauvegardées dans un format propriétaire dans un fichier texte.
- Données sauvegardées dans une base de données telle que SQL Server en utilisant un composant natif (Historical Data Server).

D'habitude, PcVue utilise les fichiers complexes d'extension .dat pour stocker les données. Afin d'avoir une base de données temps réel, nous avons choisi de sauvegarder les données dans une base de données SQL Server.

Microsoft SQL Server 2014 est un système de gestion de base de données relationnelle (SGBDR) en langage SQL développé par Microsoft. Il existe en différentes éditions : CE (Compact Edition - solution embarquée pour les smartphones), Express (version gratuite), Developper,

[3] https://www.talend.com/products/talend-open-studio/

Standard, BI et Enterprise. Nous avons opté pour l'utilisation de l'édition Standard qui offre des fonctionnalités élémentaires en base de données, rapports et analyses.

2.2 Langages de développement

Afin de faire des transformations spécifiques et nécessaires à l'intégration de données, nous avons utilisé IDLE comme environnement de développement intégré pour le langage Python et Talend Open Studio pour la création des routines à travers un code java.

2.2.1 Java

Java est un langage de programmation informatique orienté objet et open source développé par Sun Microsystems. Une de ses plus grandes forces est son excellente portabilité.

2.2.2 Python

Python [4] est un langage de programmation interprété qui peut s'utiliser dans de nombreux contextes et s'adapter à tout type d'utilisation grâce à des bibliothèques spécialisées. Le langage Python est placé sous une licence libre et fonctionne sur la plupart des plates-formes informatiques. Parmi les avantages de Python est qu'on peut créer des petits programmes très simples, appelés scripts, chargés d'une mission très précise ou bien créer des programmes complets, comme des jeux, des suites bureautiques, des logiciels multimédias, des clients de messagerie ou des projets très complexes, comme des progiciels.

3 Développement de l'ETL pour la solution à court terme

3.1 Problèmes rencontrés et étude de la solution

Le système PcVue fonctionne de la manière suivante: Des automates (DX, SAIA), représentés par des API et qui récupèrent les données des capteurs, sont reliée à un switcher et puis à un ordinateur contenant le système PCVUE. Ce dernier récupère les points représentatifs des capteurs, comme par exemple le point suivant : CVC.SSP.E4.DEBIT.MES, et applique un processus bien spécifique pour les enregistrer. En effet, le principe est de créer des variables dans le système puis de créer des répertoires et des sous répertoires pour chaque point relevé afin d'avoir un chemin précis de chaque capteur pour accéder finalement à la mesure de ce capteur.

[4] https://www.python.org/

Nous allons expliquer davantage à travers la figure 1 l'architecture du système Pcvue et la communication entre PcVue et METASYS.

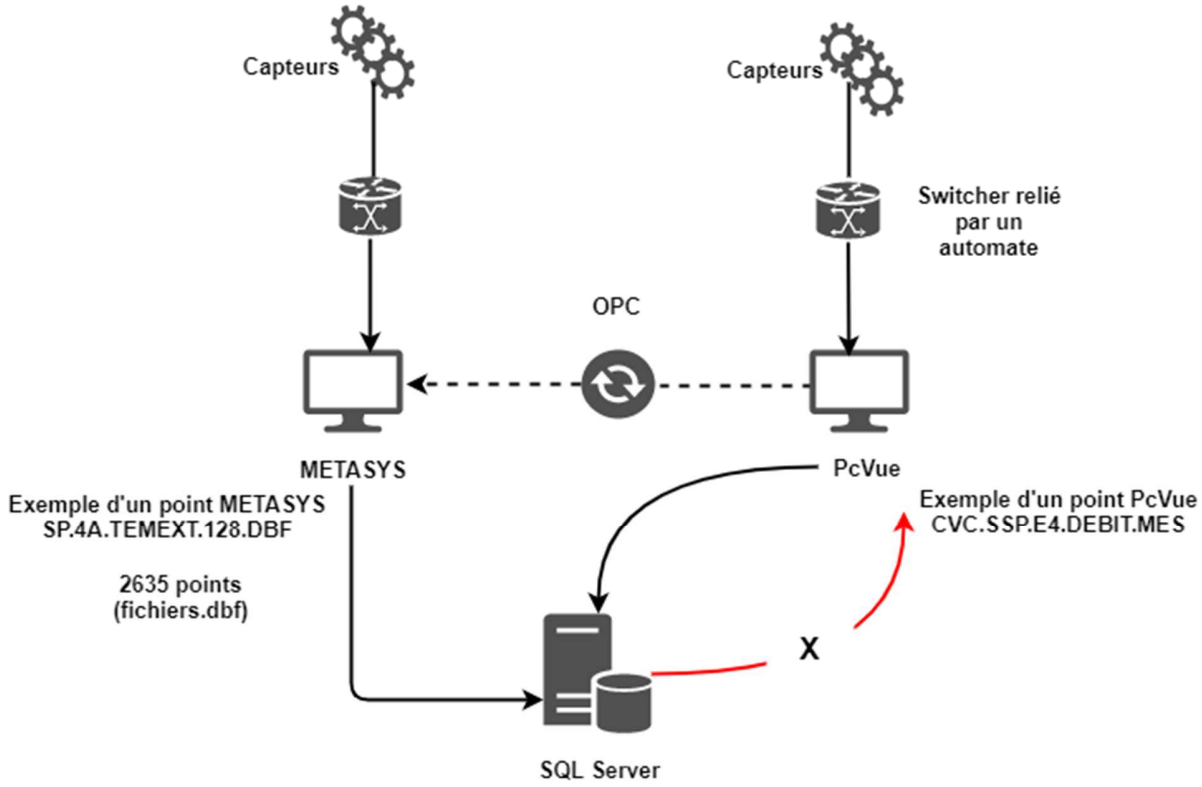


Figure 1: Communication entre PcVue et METASYS

Notre idée principale était d'exporter les données de PCVUE qui sont enregistrées dans une base propriétaire à lui vers une base de données SQL Server et au même temps y intégrer les données de METASYS afin de les remonter et les consolider par PCVUE. Néanmoins, à cause du principe des points indiqué au-dessus, le système PcVue n'arrive pas à récupérer les données sans avoir ces points définis dans le système et donc sans la création d'un chemin pour chaque point relevé du METASYS qui englobe pour le moment 2635 fichiers (points). Nous avons comme alternative, la communication via un lien virtuel OPC (OLE for Process Control). Ainsi, le système PcVue récupère les données de METASYS à travers le lien OPC, qui est un standard de communication dans le domaine de control de processus « automate ». Face à ces problèmes, nous avons étudié les avantages et les inconvénients de chaque solution.

• Création des 2635 points en suivant l'arborescence et la charte de PCVUE «

SGE.UPS.4A.SSP.CVC.ECH1. RECV2V.MES » : dans ce cas de figure, la source sera interne, donc, nous pouvons récupérer les données à la source avec une arborescence correcte qui suit la charte PcVue. Nous n'avons pas besoin d'une liaison OPC. EN revanche, le grand inconvénient de cette solution est qu'il faut créer à la main l'arborescence de chaque point dans le système PcVue qui va prendre beaucoup de temps.

• Création d'une arborescence selon notre choix et qui correspond à METASYS dans le système PCVUE. Dans ce cas, l'accès établi sera via une liaison OPC : « METASYS.SP-4A.RECV2V.MES ». Cette solution est simple et rapide. Par contre, elle manque de fiabilité de communication et il y a une limite au niveau du nombres de points que le lien peut remonter.

• Création d'une arborescence selon notre choix et qui correspond à METASYS dans le système PCVUE et l'accès sera interne et via SQL Server. Cette solution consiste à créer l'ensemble des points automatiquement à travers un fichier de configuration contenant les chemins de fichiers. Cette dernière solution ne suit pas la charte de PcVue mais nous pouvons récupérer les données à la source et c'est rapide à mettre en œuvre.

A travers cette étude, nous avons opté pour la troisième solution parce qu'elle est la plus fiable et la plus rapide à mettre en place. Pour cette raison, nous avons développé un script avec le langage de programmation Python qui permet de créer un fichier Excel contenant l'ensemble des points de METASYS. Ce script permet de parcourir les répertoires et les sous répertoires et de générer un nom spécifique à notre charte comme par exemple le point suivant: **"METASYS.SP-4A.RECV2V.MES"**. Nous avons choisi de commencer le chemin par "METASYS" pour indiquer qu'il s'agit des points remontés depuis le système METASYS. Puis le chemin est augmenté par le nom du répertoire SP-4A qui indique "le bâtiment 4A" et le local technique "sous station primaire". Nous rajoutons par la suite, "RECV2V" qui représente le capteur. Et enfin, nous terminons le chemin par "MES" pour indiquer qu'il s'agit des mesures du capteur au lieu d'avoir des noms non significatifs comme 128.DBF. Nous pouvons maintenant créer tous les 2635 points METASYS à travers ce fichier de configuration d'une manière automatique dans le système PcVue.

3.2 Diagrammes d'activités

Nous avons opté, pour le développement de cette partie vers l'utilisation du diagramme d'activités puisqu'il modélise un flux et décrit la logique d'une opération. Le diagramme 2 présente la phase d'intégration générale qui consiste à récupérer les données à partir de la source et à les charger dans la base de destination, cette fonctionnalité sera assurée par les activités citées ci-dessous :

- Vérification de la connexion à la base de données.
- Extraction des données.
- Transformation des données.
- Alimentation de l'entrepôt de données.

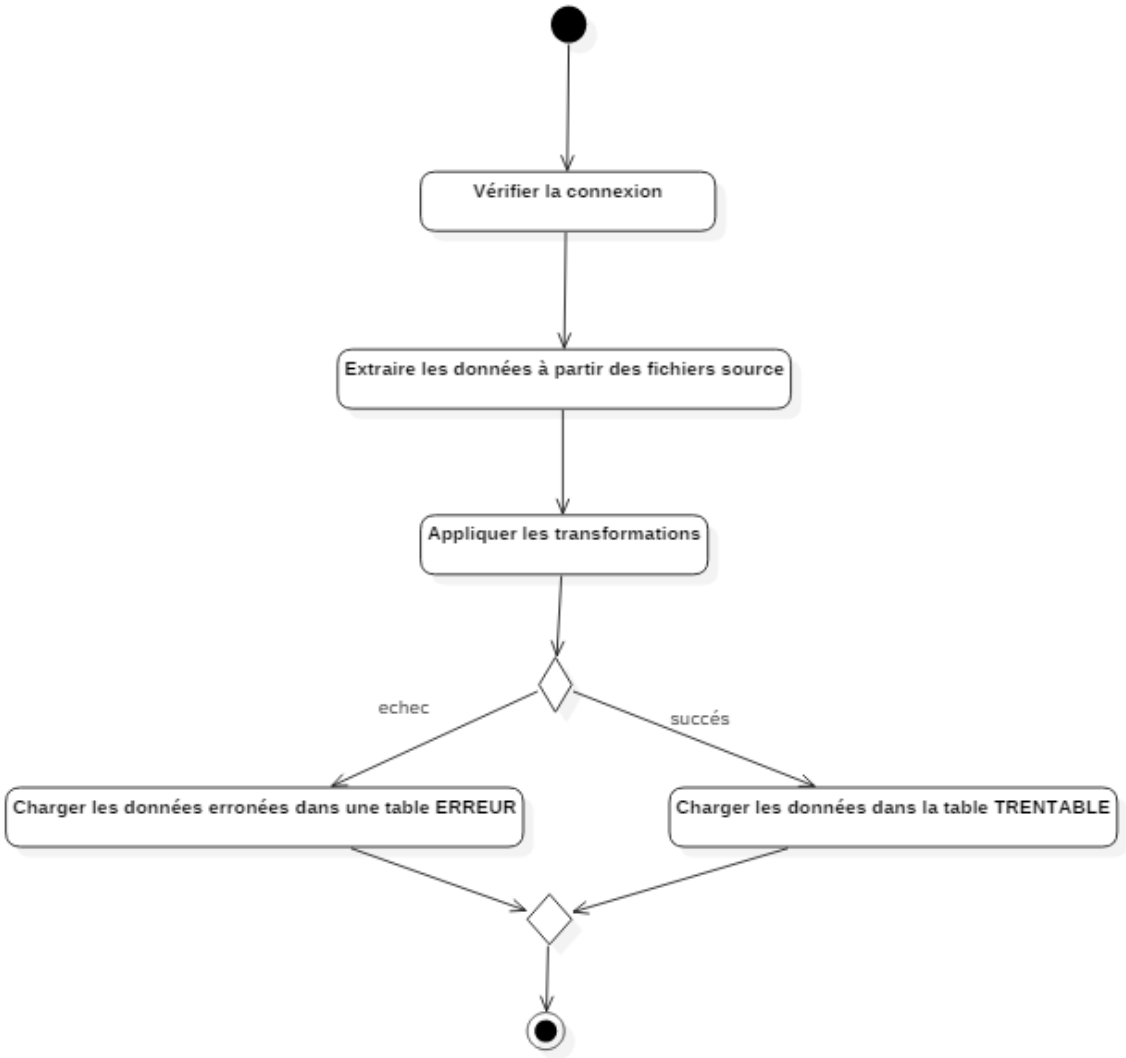


Figure 2: Diagramme d'activités pour l'alimentation de la table TRENDTABLE

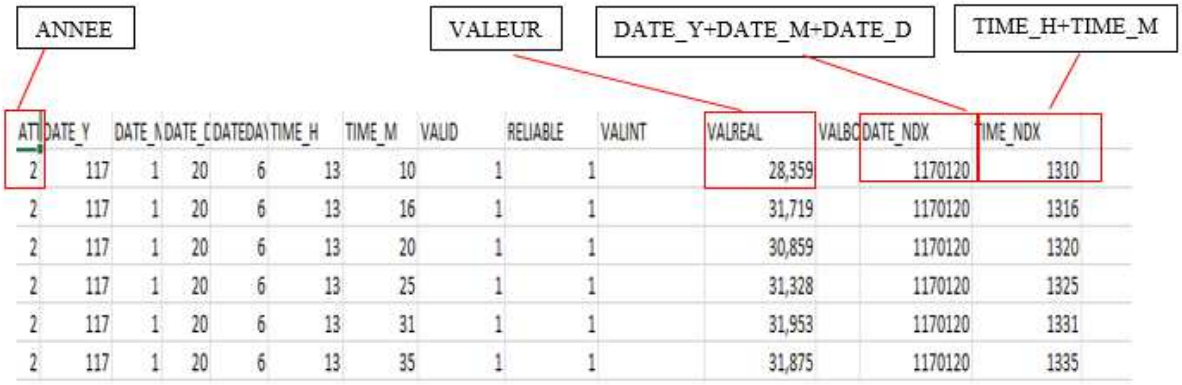
3.3 Description des processus ETL

Nous présentons les étapes de création du job sous Talend Open Studio qui a comme but de migrer les données de chaufferie des fichiers.dbf provenant du système METASYS vers la base de données SQL Server qui contient les données déjà exportées de PcVue.

3.3.1 La source de données

Le SGE dispose de 2635 fchiers .dbf qui représentent les données de capteurs de chaufferie. Il est habitué à faire une sauvegarde chaque soir de ces fichiers sur une machine

distantes et à faire une purge de ces données chaque 6 mois. Dans cette partie nous allons décrire le format des fichiers et lister les difficultés rencontrées dans le traitement de la source. Le contenu de fichier.dbf a la forme suivante (3):



ATT	DATE_Y	DATE_M	DATE_D	DATEDAY	TIME_H	TIME_M	VALID	RELIABLE	VALINT	VALREAL	VALBO	DATE_NDX	TIME_NDX
2	117	1	20	6	13	10	1	1		28,359		1170120	1310
2	117	1	20	6	13	16	1	1		31,719		1170120	1316
2	117	1	20	6	13	20	1	1		30,859		1170120	1320
2	117	1	20	6	13	25	1	1		31,328		1170120	1325
2	117	1	20	6	13	31	1	1		31,953		1170120	1331
2	117	1	20	6	13	35	1	1		31,875		1170120	1335

Figure 3: Extrait d'un fichier.DBF

La difficulté étant que les fichiers n'ont pas un nom significatif qui suit une charte, ils se représentent sous la forme de 128.DBF ou 8322.DBF. Également, nous n'avons pas une colonne qui indique le nom du fichier ou son emplacement et ceci est indispensable pour pouvoir créer les variables dans le système PcVue avant l'intégration des données. En effet, PcVue n'arrive pas à lire les données de METASYS sans la création de ces points qui permettent de fixer le capteur source des informations liées à cette variable. En plus de ça, la date de la mesure se présentent sur plusieurs colonnes sans avoir un format spécifique. Et finalement, il nous manque un champ chrono qui indique le temps écoulé depuis 1970 et que PcVue utilise pour préciser le temps de prélèvement de chaque valeur des capteurs.

3.3.2 La destination des données

Notre destination est une table appelée 'TRANDTABLE' sous SQL Server contenant les données exportées depuis PcVue. Nous allons représenter à travers la figure 4 un extrait des données de PcVue.

PROJECT	CHRONO	RECTS	STATION	UNITNAME	VARTYPE	NAME	
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C122.DEBIT.MES	De
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C122.DELTA_T.MES	De
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C122.PUISSANCE.MES	Pu
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C122.TEMP_DEP.MES	Te
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C122.TEMP_RET.MES	Te
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C122.VOLUME.MES	Vo
GTE_UPS	1401714000000	2 juin 2014	1	TREND CVC	REG	CVC.ISAE.C123.DEBIT.MES	De

Figure 4: Extrait des données PcVue

• Le chrono : Heure de l'évènement de la mesure du capteur en milliseconde depuis 1970.

• TS : date de la mesure en UTC (Temps Universel Coordonné).

• Value : valeur numérique indiquant la valeur du capteur.

• Name : le nom des variables.

• Quality : valeur par défaut attribué par PcVue qui sert à vérifier s'il y a ou pas des anomalies au niveau des informations récupérées.

3.3.3 Job Talend Open Studio

Après avoir décrit les données d'entrée et présenter le format de sortie, nous allons détailler les étapes de création de notre job TOS dans ce qui suit : Notre job Talend comporte plusieurs étapes :

• Création d'une connexion avec la base de données : C'est la première étape qui sert à créer une connexion avec notre base de données. Il s'agit de choisir le type de la BD et de remplir les différentes informations pour se connecter à la base.

• Création du schéma de notre table cible : Cette deuxième étape sert à récupérer le schéma de la table cible.

• Définition du job : C'est le processus de lecture et de transformation des données puis leur chargement dans la table cible.

Flux ETL pour l'extraction des données METASYS

Nous détaillons dans la figure 5 le processus d'extraction, de transformation et de chargement des données source du système METASYS.

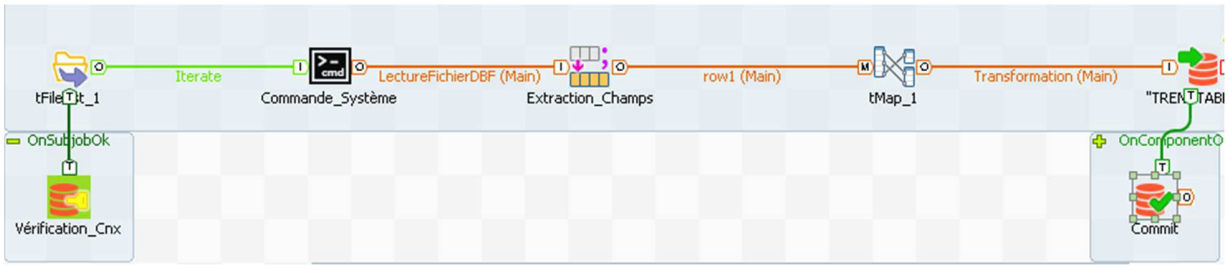


Figure 5: Job d'intégration METASYS dans SQL Server

Notre ETL commence par l'appel de métadonnée crée auparavant pour la connexion aux bases de données avec le composant « Verification_Cnx ». Une fois la connexion établie avec succès, nous pouvons parcourir l'ensemble des répertoires contenant les fichiers.dbf avec le composant tFile_List. A chaque itération, nous récupérons le nom de chaque fichier pour l'utiliser comme entrée dans le composant qui suit « Commande_Système ». Ce composant nous permet d'exécuter un script python qui permet, d'une part, de lire les fichiers.dbf, et d'autre part, de créer un champ « NAME ». Ce champ est la concaténation des répertoires et sous répertoires du fichier.dbf afin d'obtenir une indication sur le chemin et la source des capteurs. Ensuite, nous récupérons le résultat de ce composant dans une variable de sortie à utiliser par le composant suivant « Extraction_Champs » pour la décomposer en plusieurs colonnes. De cette manière-là, nous avons arrivé à lire les fichiers.dbf et il faut maintenant faire les transformations nécessaires et adapter le schéma source au schéma cible. Les composants disponibles dans Talend Open Studio ne puissent pas répondre avec exactitude au besoins exprimés auparavant. Pour cette raison, nous avons créé tout au début une routine afin de pouvoir réaliser notre tâche. Les routines permettent d'exécuter des traitements spécifiques via le langage JAVA. Dans notre cas, notre routine sert à transformer les formats et les types des champs en entrée. Nous avons alors créé trois méthodes :

• La première permet de concaténer les champs Date_NDX et TIME_NDX, faire les tests nécessaires sur les valeurs manquantes et générer comme résultat un seul champ TS qui a

un format de date correct.

• La deuxième permet de calculer la valeur de chrono depuis la date générée dans la méthode précédente.

• La troisième permet de convertir les types de données en entrée au type et format approprié.

Il nous reste alors qu'appeler notre routine via le composant tMap. **TMap** est l'un des composants principaux utilisé pour mapper les données d'entrée vers les données de sortie, c'est-à-dire mapper un schéma à un autre. Il permet aussi d'appliquer des filtres et des jointures. La figure 6 représente l'éditeur de mapping du TMap_1:

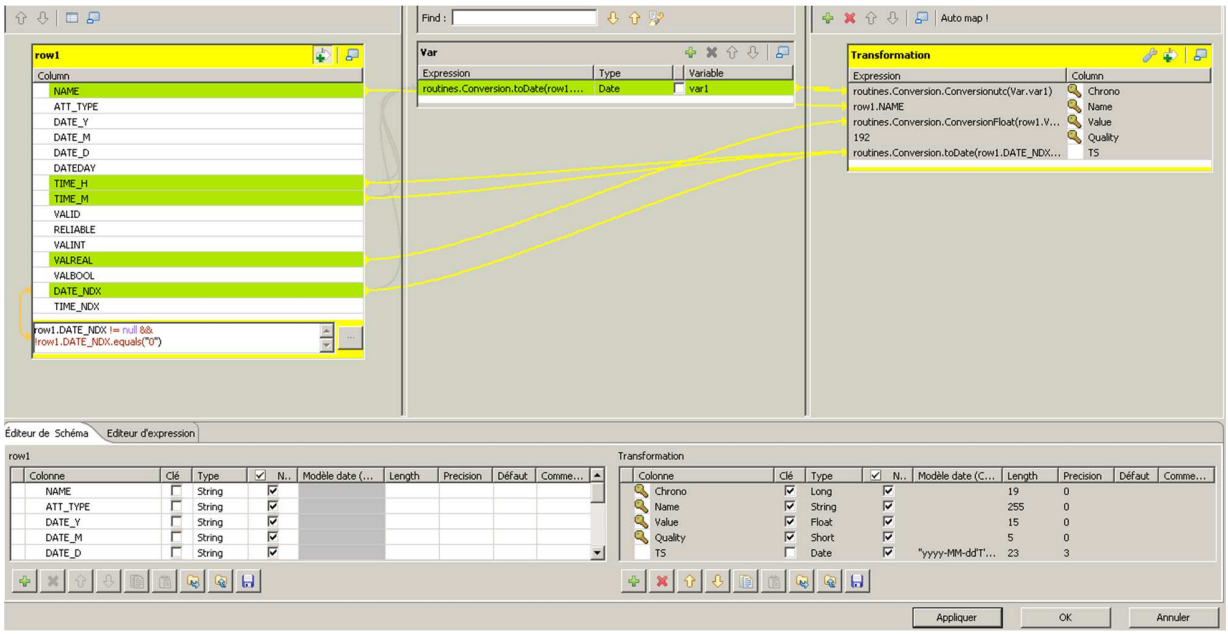


Figure 6: Éditeur de mapping pour faire les transformations

A travers cette fenêtre, nous avons appelé les méthodes de la routine « Conversion » via le constructeur d'expression de chaque colonne. Puis, nous avons appliqué un filtre sur les données en entrée pour ignorer les valeurs nulles. Finalement, nous avons alimenté la table cible par le flux entrant à travers le composant tMSSqlOutput représenté sous le nom « TRENDABLE » et appliquer par la suite un commit via le composant tMSSqlCommit pour valider les données traitées.

Exécution du job

Le SGE a choisi d'intégrer les données dans la base chaque soir après avoir terminé la sauvegarde automatique. Nous avons alors crée un fichier.bat qui permet de faire une copie des fichiers d'un pc à un autre, enregistrer la trace de l'exécution de la sauvegarde dans un fichier log.txt et la date de début de sauvegarde dans un fichier.csv qui va nous servir ensuite pour le chargement incrémental (Fig 7).

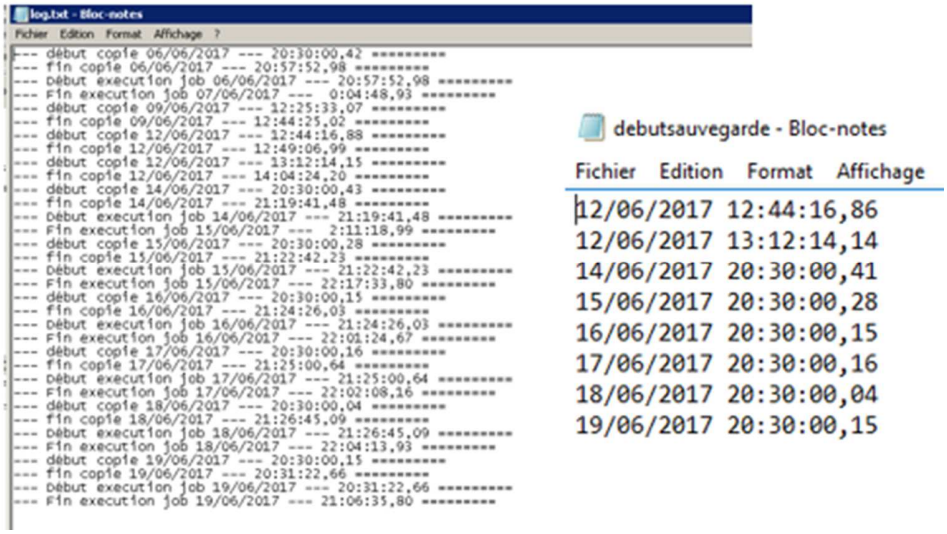


Figure 7: Trace de l'exécution du Job

Etant donné que Talend Open Studio n'offre pas la fonctionnalité de planification du job pour l'exécuter en automatique, nous avons utilisé le planificateur de tâche Windows pour créé une tâche de base qui exécute le job chaque soir.

Extraction Incrémental

Le volume de données et le nombre de lignes à parcourir lors de chaque exécution du job est très important. Pour cette raison, nous avons décidé d'appliquer la méthode d'extraction incrémental en temps différé. En effet, il s'agit de capturer uniquement les données qui ont été ajoutées depuis la dernière extraction à travers la méthode de capture basée sur les timestamps. Avec cette méthode l'extraction se fait uniquement sur les données dont le timestamp est plus récent que la dernière extraction. Cette option malheureusement n'existe que dans la version payante de Talend. Pour contourner ce problème, nous avons utilisé le fichier.csv qui contient la date de début de sauvegarde pour la comparer avec les dates des capteurs en entrées. Par conséquent, nous avons ajouté à notre job un autre composant tMap qui permet d'appliquer un filtre sur les données en entrées pour n'enregistrer que les données qui ont une date supérieur à

la dernière date de début de sauvegarde. De cette façon nous garantissons que nous avons inséré seulement les nouvelles lignes qui ont été ajoutées au fichiers.dbf. La figure 8 représente le processus après la modification :

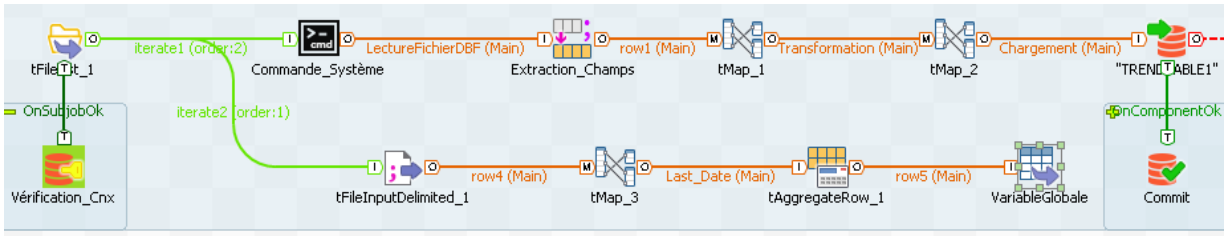


Figure 8: Modification du Job pour appliquer le chargement incrémental

Ainsi, la première étape consiste à lire les dates enregistrées dans le fichier.csv à travers le composant tFileInputDelimited_1. Ensuite, nous avons utilisé tMap_3 pour transformer le format de dates récupérées au format correspondant aux format de la date dans les fichiers sources. Puis, nous avons ajouté le composant tAggegateRow_1 qui permet de recevoir le flux de données depuis tMap et appliquer une opération d'agrégation sur la colonne date pour récupérer la dernière date enregistrés dans le fichier. Enfin, nous avons enregistré le résultat dans une variable globale pour l'utiliser dans l'expression de filtre comme le montre la figure 9.

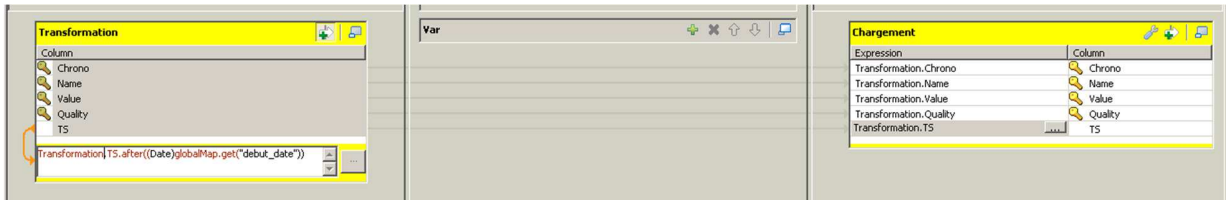


Figure 9: Ajout du filtre dans l'éditeur du mapping

Gestion des Erreurs

Les incidents peuvent toujours y arriver dans le job Talend, pour cette raison il faut mettre en place des procédures qui génèrent un rapport résumant le cycle de vie du job. Pour appliquer la gestion d'erreur dans notre projet, nous avons appliqué les deux méthodes suivantes :

- Ajouter le composant tSatCatcher qui permet de récupérer la totalité des

informations sauvegardées et de les fusionner dans un flux vers la fin de l'exécution du job. Ces informations peuvent aider à analyser le résultat d'exécution et localiser la panne en cas de besoin à travers des champs indiquant le temps, le message (success, failure), l'origine, le nom du fichier etc.

• Ajouter une table « Erreur » dans la base SQL Server qui contient les lignes rejetées de la table principale avec des champs présentant le code d'erreur et le message d'erreur.

Avec toutes ces étapes notre job final est représenté de la façon suivante(Fig 10):

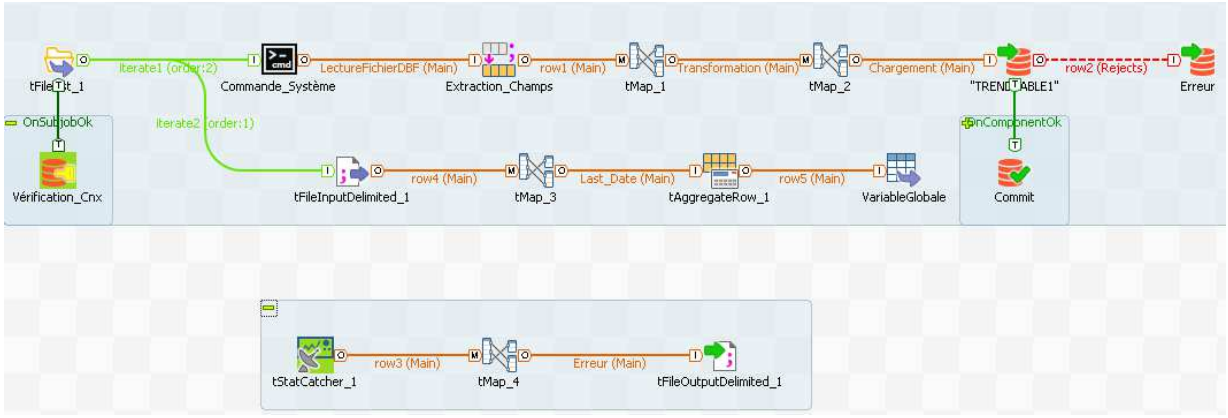


Figure 10: Job final

Ainsi, la table TRENDTABLE contient l'ensemble des données de METASYS et PcVue qui représentent a peu prés 150 000 000 lignes (Fig 11).

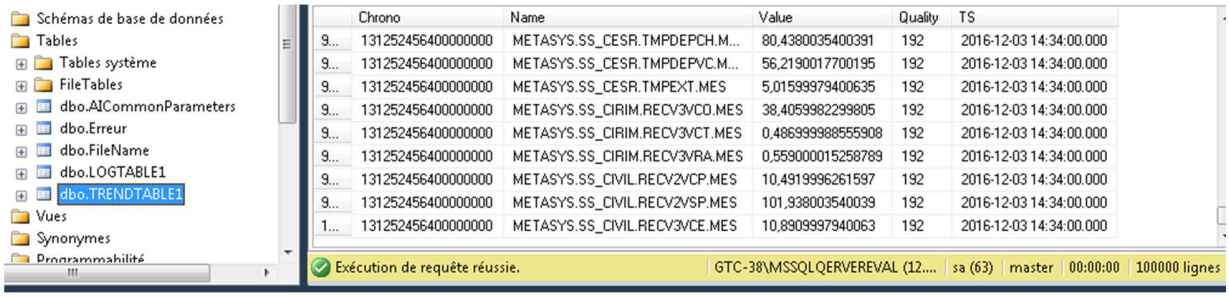


Figure 11: Aperçu des données METASYS dans la Table TRENDTABLE DE PcVue

4 Développement de l'ETL pour les données à long terme

La solution présentée pour analyser les données à court terme ne tient pas compte de l'historique du SGE sur les 10 dernière années. En effet, elle offre une pratique sur 6 mois. Pour intégrer les données d'archive, nous avons amené à proposer une nouvelle solution. Dans ce fait, nous avons développé une solution SQL Server incluant les historiques de METASYS et PcVue ainsi que les dates d'exploitation. Cette solution a été bien accueillie par les responsables de notre projet qui ont pu exploiter et analyser les données du service. Néanmoins, cette solution a montré ses limites notamment au niveau de la gestion de la volumétrie que ne cesse d'augmenter au fil des années. Pour cela, nous avons étudié et proposé une deuxième solution Big Data permettant de palier les limites de la première solution.

4.1 Intégration de l'historique sous SQL Server

4.1.1 Diagramme d'activité

Le diagramme ci-dessous présente le processus d'intégration des données d'exploitation sur une durée de 6 mois depuis la base de données SQL Server vers une nouvelle base de données SQL Server destinées pour les historiques (Fig 12).

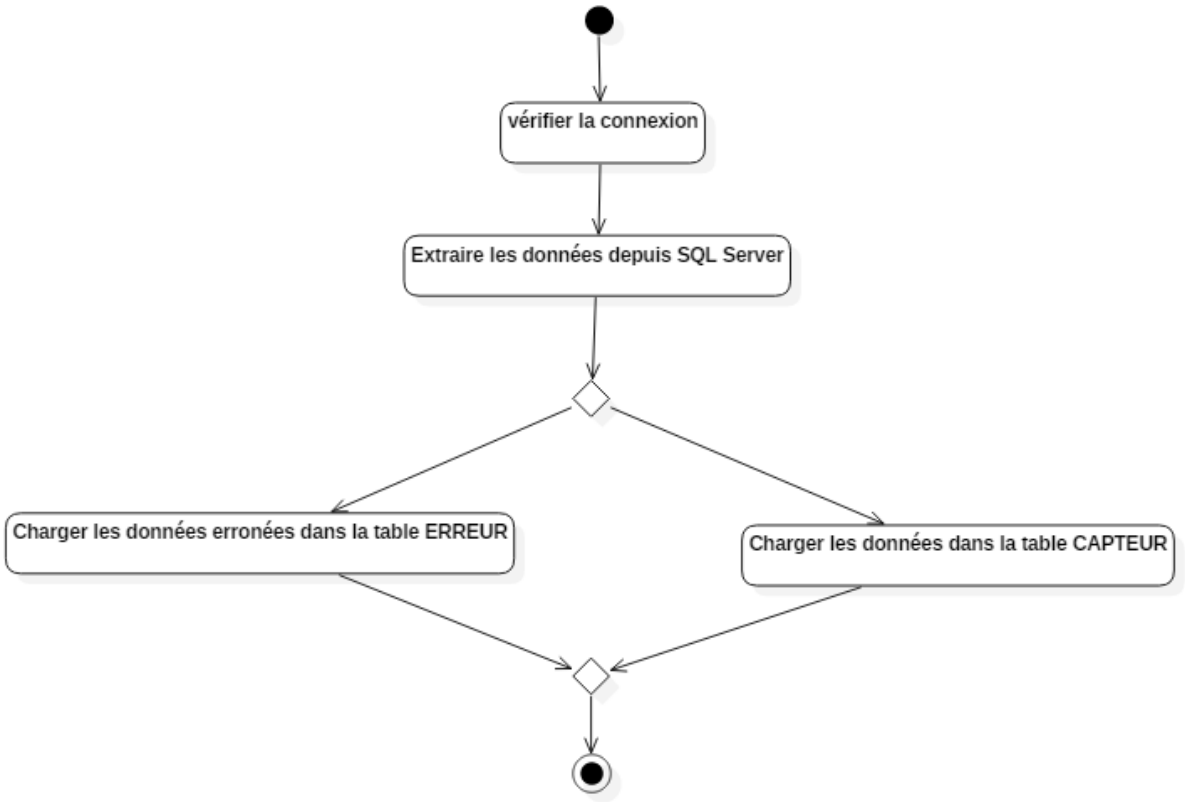


Figure 12: Diagramme d'activité pour l'alimentation de la base de l'historique

4.1.2 Développement et création de l'entrepôt de données

Afin de mettre en place un entrepôt qui englobe les données d'exploitation du SGE, nous avons décidé de créé une deuxième base de données SQL Server destinée pour les historiques des systèmes de supervision METASYS et PCVUE et pour les données journalières provenant de la première base.

Job d'intégration de données SQL SERVER

La figure 13 représente le job d'intégration de données journalière d'une base SQL Server vers une autre en utilisant la méthode de chargement incrémental par comparaison de timestamp. Pour cette raison, nous avons comparé par rapport à la date de début de sauvegarde et nous avons planifié l'exécution du job chaque soir après la fin de l'intégration de données METASYS dans la base journalière.

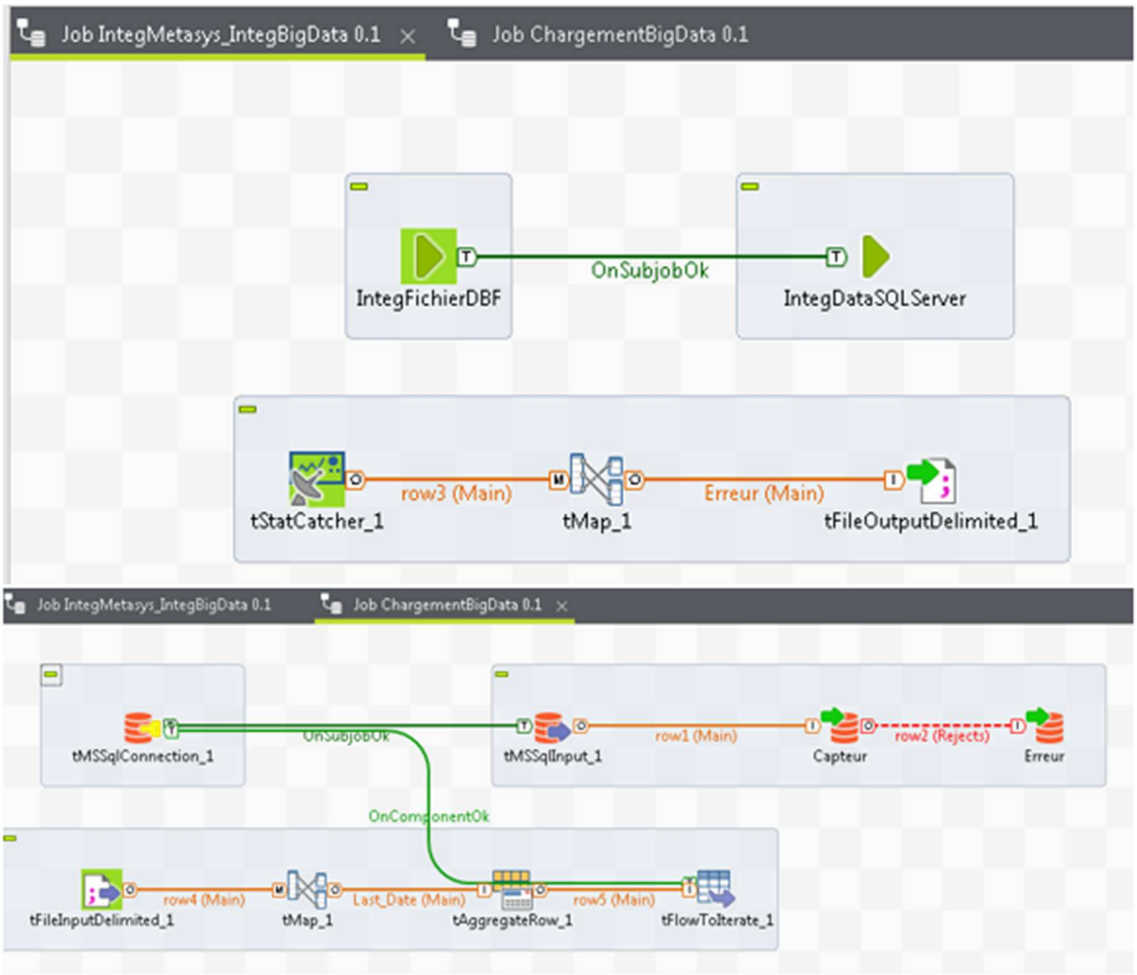


Figure 13: Intégration de la base journalière SQL Server

Job d'intégration de données METASYS

Les données de METASYS, datant depuis 2010, sont très volumineuses. Pour remédier à ce problème, nous avons décidé de créer un job indépendant et d'appliquer le processus d'intégration sur plusieurs tranches en multithread. La figure 14 représente un exemple d'insertion des données des mois octobre, novembre décembre de l'année 2015 à travers l'utilisation du composant tRunJob. Ce composant permet de faire l'appel aux jobs fils destinés à l'intégration des données des fichiers.DBF vers la base de données de l'historique.

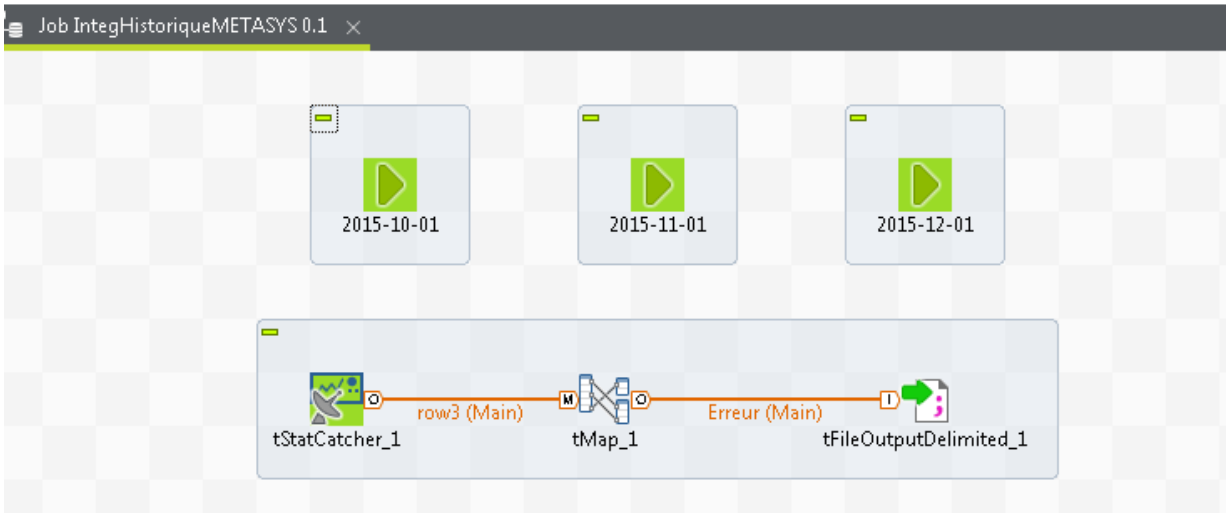


Figure 14: Intégration de l'historique de METASYS

Job d'intégration de données PCVUE

Pour finaliser notre entrepôt, nous avons inséré les historiques de PcVue pour avoir une base de données complète contenant l'ensemble de données énergétiques. Cette intégration ne peut se faire qu'après une étude des fichiers propriétaires de PcVue. Comme nous l'avons décrit auparavant dans l'analyse de l'existant, les fichiers .dat de PcVue sont composés d'un ensemble de valeurs des capteurs/compteurs. La difficulté étant de structurer ces fichiers désorganisés afin de distinguer chaque capteur. Pour remédier à ce problème, nous avons développé un script Python pour décomposer les fichiers à partir des entêtes écrits là-dedans et créer une structure qui correspond à notre chemin cible. Cette structure comporte comme attributs le nom du capteur, la date et la mesure. Puis, nous avons crée un job Talend qui contient les mêmes composants utilisés pour l'intégration des fichiers METASYS mais nous avons rajouté des modifications dans notre routine. Nous avons expliqué, dans la partie d'intégration des données METASYS dans SQL Server, qu'il fallait créer une routine contenant du code java pour développer une fonction qui converti la chrono en date. Maintenant, il faut créer une fonction qui fait l'inverse et qui converti les dates de mesure en chrono : nombre en millisecondes depuis 1970 comme le montre la figure 15 :

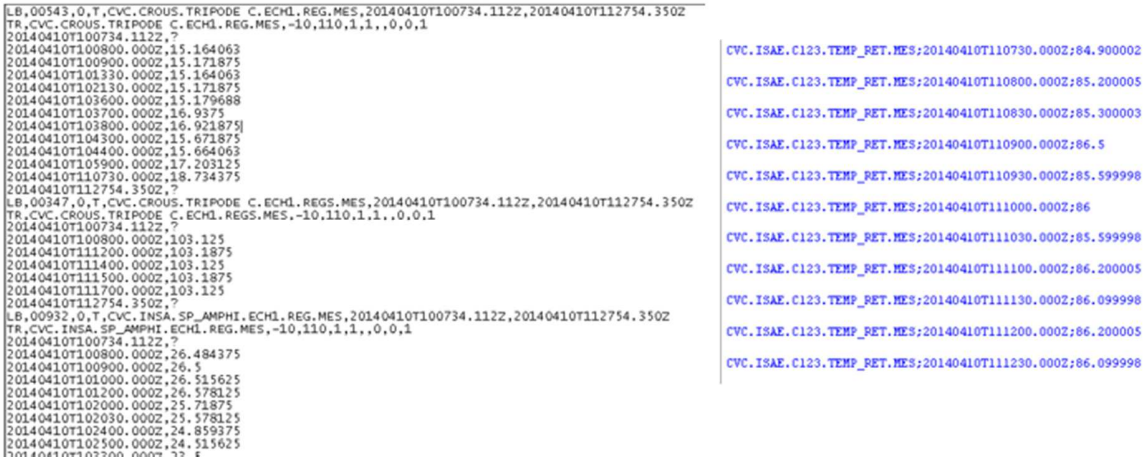


Figure 15: Structuration des fichiers.dat du PcVue

4.2 Développement de l'ETL pour le Big Data

La solution proposée dans la section précédente, permet d'intégrer toutes les données d'exploitation du SGE dans une seule base de données, mais cette solution a deux limites majeurs :

- Le volume important de données qui augmente d'une manière exponentielle.
- L'analyse selon différents axes et critères qui est absente dans cette solution puisque il nous manque la base patrimoine ACCESS.

Pour résoudre ces problèmes, nous avons décidé d'utiliser Hadoop/hdfs,un système de fichiers distribué qui donne un accès haute-performance aux données réparties dans des clusters Hadoop. Nous avons choisis Hive pour créer notre schéma multidimensionnel et interroger nos tables et Hbase pour créer le cube OLAP. Nous avons décidé d'utiliser l'outil Talend Open Studio For Big Data pour réaliser l'intégration. Nous avons réalisé une étude sur les outils existants pour mettre en place facilement un cluster BigData. Par conséquent, nous avons choisi le bac à sable (sandbox) Hadoop de Hortonworks. c'est une application d'un noeud simple Hadoop, basée sur la Data Platform de Hortonworks (HDP). Il s'agit de l'installer sur une machine virtuelle Vmware ou VirtuelBox pour travailler avec l'écosystème Hadoop. Le bac à sable fournit un environnement de développement qui comprend plusieurs composants: Apache Hadoop, Apache Spark, Apache

Hive, Apache Hbase. A travers Data Platform de Hortonworks qui vise à faciliter le déploiement et la gestion des clusters Hadoop nous pouvons gérer et exploiter Hadoop à travers Apache AMbari, gérer les données à travers Hadoop HDFS et intérroger les données via Hive.

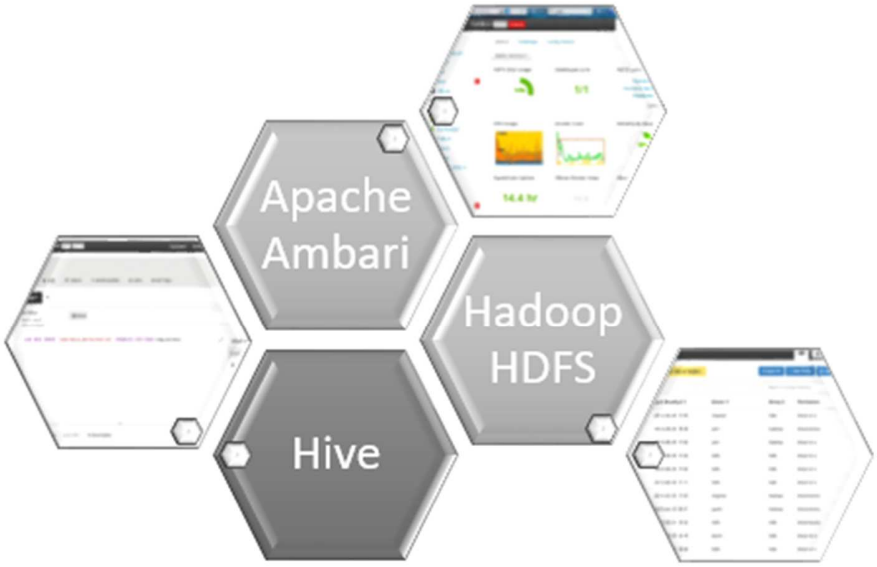


Figure 16: Architecture de la solution Big Data

La figure 16 présente l'architecture de la solution Big Data :

• Hadoop Distributed File System (HDFS): Système de fichiers distribués qui donne un accès hautes performance aux données réparties dans des clusters Hadoop.

• Hive: une surcouche analytique à Hadoop qui offre une couche d'abstration aux données HDFS sous forme d'un modèle tabulaire et relationnel, et un langage de requête SQL nommé HiveQL.

• Ambari: Interface permet la mise à disposition, la gestion et exploitation des clusters Hadoop

4.3 Création du schéma multidimensionnel en étoile

La première étape à réaliser pour mettre en place notre solution Big Data est de créer le schéma multidimensionnel en étoile avec l'outil Talend. Notre ETL commence par extraire,

transformer et charger les tables relatives aux dimensions. Puis traite les tables des faits. Ceci est dû au fait que les tables de faits contiennent des clés étrangères vers les dimensions. Nous détaillons dans la figure 17 le processus d'extraction, de transformation et de chargement des nouvelles données dans les dimensions D_Temps et D_Item.

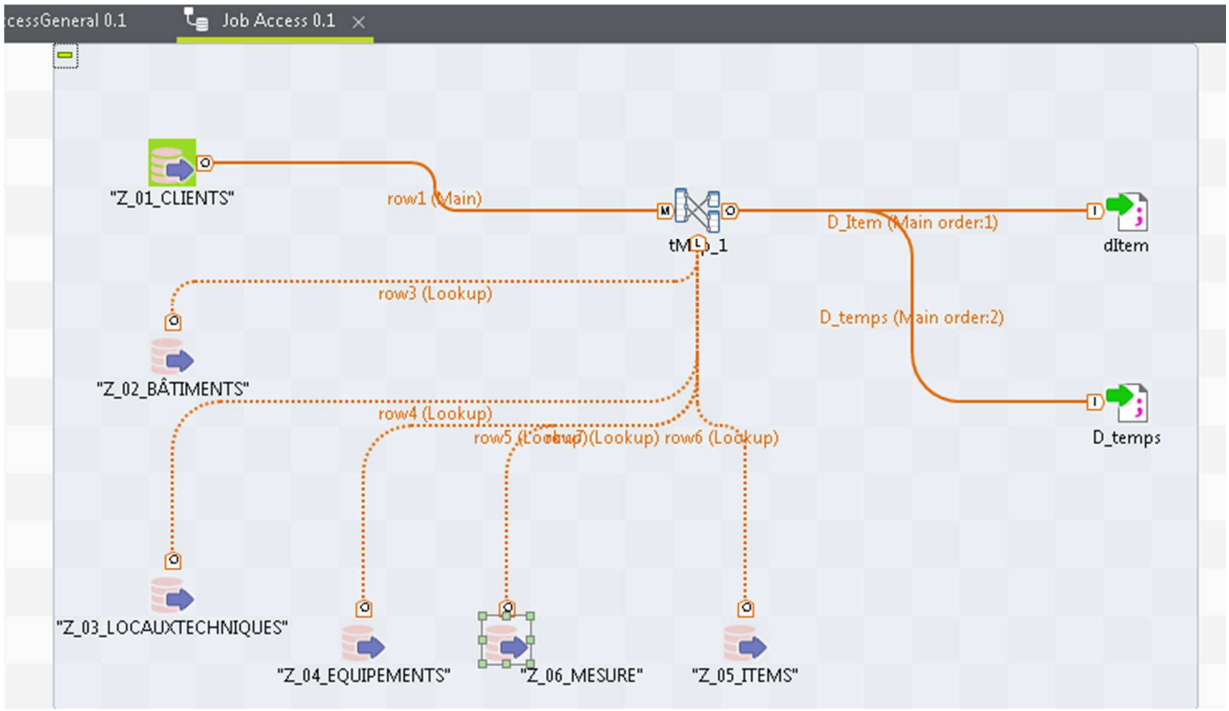
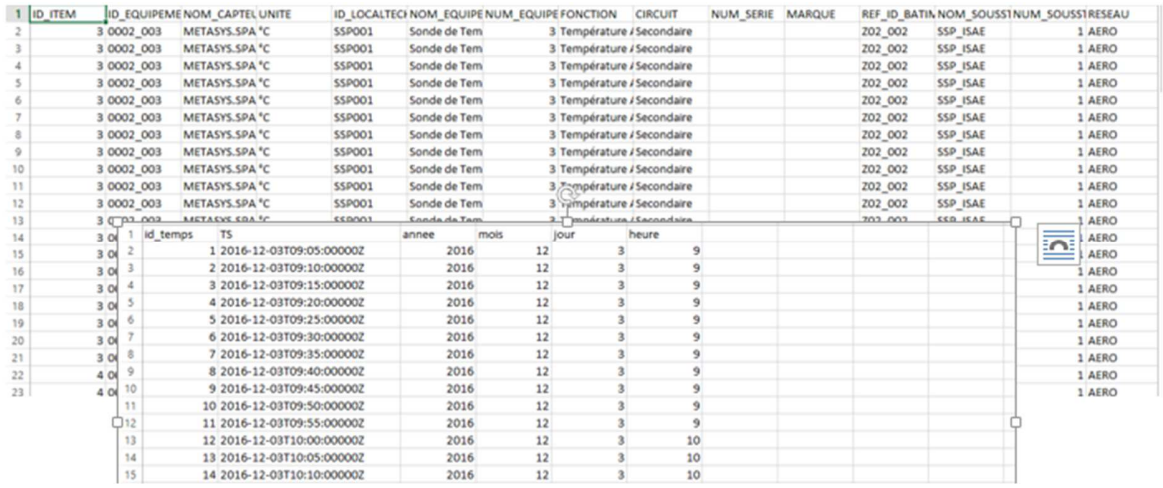


Figure 17: Chargement des tables de dimensions

Tout d'abord, nous définissons notre source Access, la base opérationnelle, pour charger notre dimension. En commençant par la dimension D_Item, nous récupérons le schéma des table Clients, Bâtiments, LocauxTechniques, Equipement, Item à travers le composant tAccessInput. Puis nous le relions avec le composant tmap afin de faire la jointure entre ces tables. Ensuite, nous créons une sortie qui est sous la forme d'un fichier.csv pour l'intégrer dans les prochaine étapes dans Hadoop/HDFS. Nous finissons par matcher les champs de la source avec les champs de la cible. En ce qui concerne la dimension D_temps nous utilisons le champs TS de la table Mesure pour créer notre fichier cible TEMPS.csv tout en faisant les transformations nécessaires avec le composant Tmap. En effet, à partir de ce champ nous devons extraire les l'année, le mois, le jour et l'heure dans des colonnes séparées. En exécutant le job nous avons obtenu les fichiers suivants(Fig 18):



id_temps	TS	annee	mois	jour	heure
1	2016-12-03T09:05:00000Z	2016	12	3	9
2	2016-12-03T09:10:00000Z	2016	12	3	9
3	2016-12-03T09:15:00000Z	2016	12	3	9
4	2016-12-03T09:20:00000Z	2016	12	3	9
5	2016-12-03T09:25:00000Z	2016	12	3	9
6	2016-12-03T09:30:00000Z	2016	12	3	9
7	2016-12-03T09:35:00000Z	2016	12	3	9
8	2016-12-03T09:40:00000Z	2016	12	3	9
9	2016-12-03T09:45:00000Z	2016	12	3	9
10	2016-12-03T09:50:00000Z	2016	12	3	9
11	2016-12-03T09:55:00000Z	2016	12	3	9
12	2016-12-03T10:00:00000Z	2016	12	3	10
13	2016-12-03T10:05:00000Z	2016	12	3	10
14	2016-12-03T10:10:00000Z	2016	12	3	10

Figure 18: Les fichiers.csv représentants les dimensions TEMPS et ITEM

Une fois que les dimensions sont chargées, il faut charger le fait mesure et créer le fichier F_Mesure.csv. Cette étape nécessite la récupération des clés étrangères des dimensions et la récupération des champs de mesures à partir de la table source. Nous devons alors créer un job Talend où nous définissons la source de la table de la classe représentative dans la base opérationnelle ainsi que les dimensions déjà crées.Ensuite nous relions les clé étrangères et les mesures à travers le composants tmap. Et finalement, nous chargeons le fichier.csv (Fig 19).

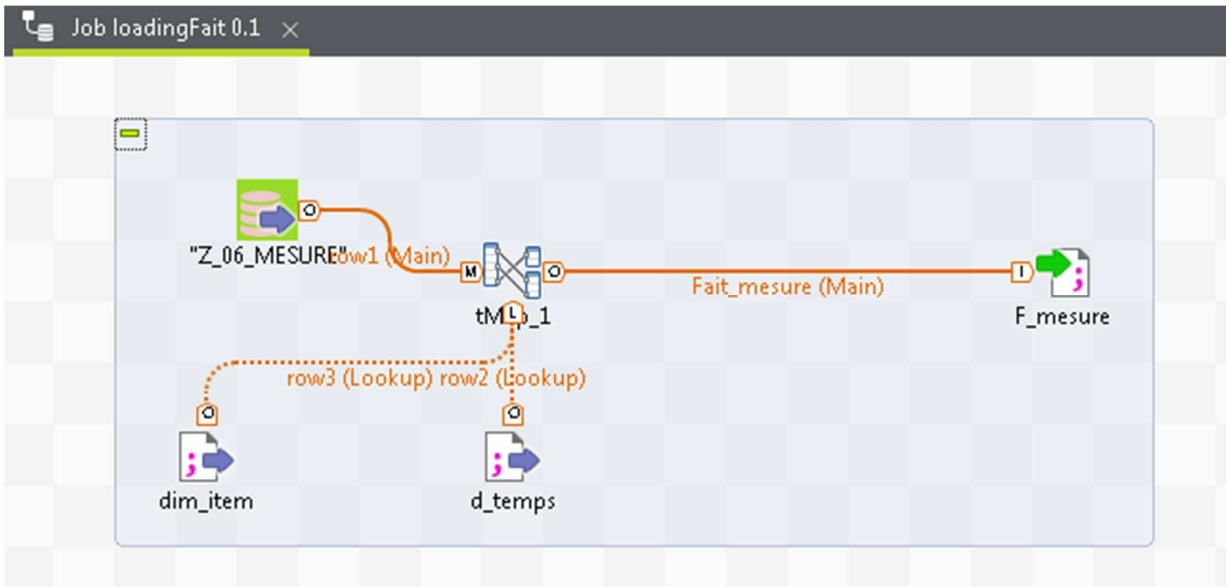


Figure 19: Chargement du fait Mesure

Après avoir créé les fichiers, nous les importons dans HDFS dans la plate-forme Hortonworks. Ensuite nous procédons à la création de notre schéma multidimensionnel en étoile sous Hive. Le principe est de créer des tables de notre schéma dans Hive avec le langage HiveQL où nous allons charger nos dimensions et notre fait. La figure suivante détaille un exemple de requête de création d'une table Hive qui représente la dimension temps et un exemple de la requête qui permet de charger le fichier temps.csv depuis HDFS vers la table Hive.

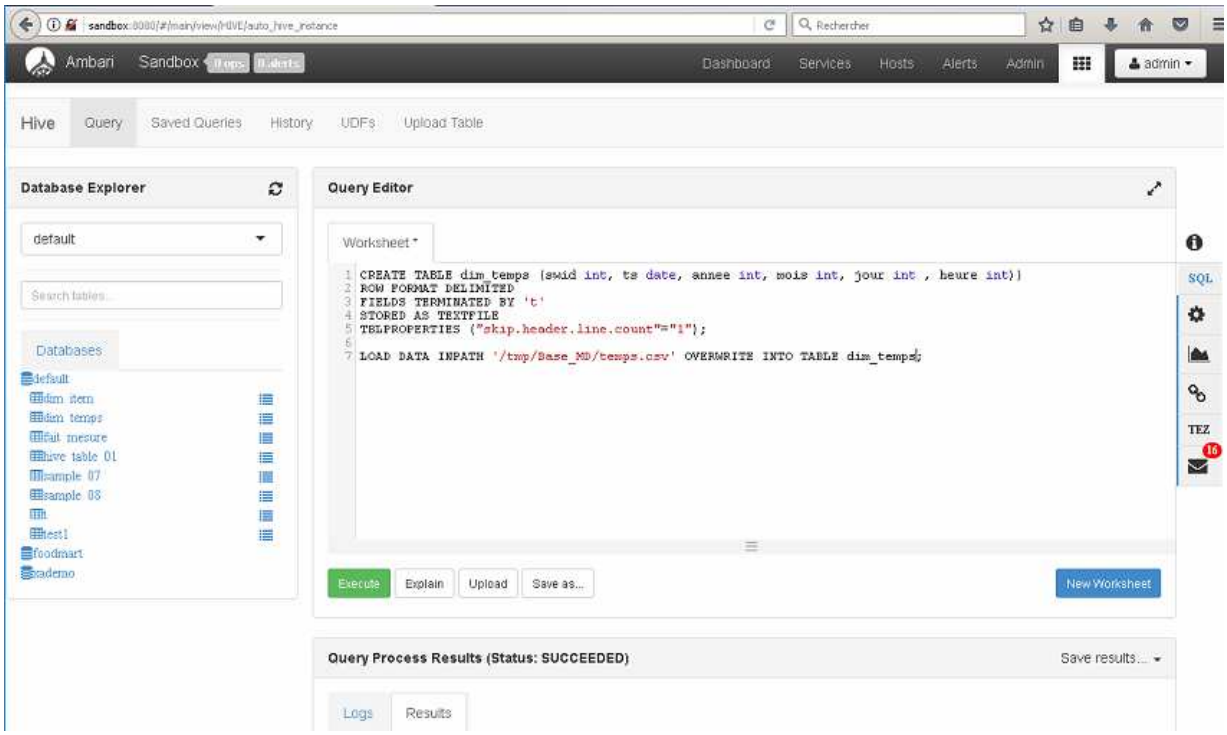


Figure 20: Création du schéma multidimensionnel dans Hive

5 Conclusion

A travers ce chapitre, nous avons détaillé les problèmes rencontrés pour intégrer les deux systèmes de supervision, nous avons proposé les solutions et défini la conception de l'ETL et son développement pour l'exploitation sur une durée de 6 mois, pour les historiques et pour le Big Data. A ce stade, il ne reste qu'à implémenter l'application d'interrogation de l'entrepôt de données.

CHAPITRE 5 : Restitution

1 Restitution

Modélisation et intégration de données de capteurs/compteurs du SGE

1 Introduction

Dans ce chapitre, nous présentons la modélisation de l'application BI en définissant ses différents types, usages et utilisateurs. Dans un second temps, nous entamons le développement de cette application.

2 Modélisation de l'application BI

Lors de la spécification des besoins où nous avons entretenu les membres de la société cliente de ce système décisionnel, nous avons défini le type d'application d'interrogation de l'entrepôt de données attendu, pour quel usage, et pour quel type d'utilisateur. Nous détaillons en ce qui suit le type d'application BI attendue, l'usage de cette dernière et ses utilisateurs.

2.1 Type d'application BI

• Tableaux de bords: C'est un ensemble d'indicateurs de pilotage, construits de façon périodique, à l'intention du responsable, afin de guider ses décisions et ses actions en vue d'atteindre les objectifs de performance.

• Applications analytiques: C'est la représentation des données sous la forme d'un cube multidimensionnel où chaque côté est une dimension d'analyse et chaque case une métrique pour pouvoir appliquer les opérations sur les données, la navigation dans les hiérarchies dimensionnelles et les visualisations des données sous le forme de tableaux croisées.

2.2 Type d'analyse BI

• BI stratégique: Ce type d'analyse est axée sur l'avenir, ce qui permet à une entreprise de prendre des décisions éclairées concernant les conditions futures de sa place dans un marché ou dans un secteur particulier.

• BI tactique: la tactique fournit aux décideurs les informations nécessaires pour surveiller les changements en temps réel de leur environnement et les aide à économiser de l'énergie. L'intelligence tactique aborde les étapes d'actions qui doivent être prises en compte pour atteindre les objectifs stratégiques de l'entreprise.

• BI opérationnel: Il concerne l'état opérationnel de l'entreprise Ce type utilise les données issues en temps réel des applications pour les croiser avec les données historiques de l'entreprise afin de fournir un support informationnel aux points d'affaire de l'entreprise.

2.3 Type des utilisateurs BI

• Visiteurs : ce sont les cadres supérieurs ayant peu de temps à consacrer à l'utilisation de la solution décisionnelle .

• Responsable : ce sont les analystes et les responsables électromécanique .

• Opérateurs : ce sont les techniciens qui exploitent les données journalières.

3 Développement de l'application BI

3.1 Analyse des données avec PcVue

Grace à l'intégration des données des systèmes de METASYS et PcVue dans une même base de données, nous avons résolu les problèmes de communication entre ces deux systèmes hétérogènes tout en bénéficiant des courbes et des analyses que PcVue offert. Egalement, avec cette solution nous pouvons maintenant afficher les tendances et comparer les données METASYS qui n'ont pas le même intervalle de temps de prélèvement de mesure. Ainsi la figure 1 montre un exemple d'affichage de tendance pour le bâtiment TRIPODE A. Les axes d'analyse

représentés dans les courbes désignent :

• Puissance : analyser les valeurs de la puissance instantanée du compteur de calorie.

• Débit: analyser le débit de l'eau.

• Énergie: vérifier le fonctionnement des compteurs à travers les valeurs des indexes.

• TEMP DEP, TEMP RET: il s'agit de la température de départ et la température de retour.

• METASYS.SPAEST.RECVEVRE.MES: des valeurs provenant du système METASYS qui représentent les valeurs des capteurs de vannes de régulation.

A travers ces courbes les utilisateurs peuvent vérifier le démarrage, la bonne ouverture des vannes à chaque appel de puissance, faire le suivi et le contrôle des valeurs des capteurs/compteur des deux systèmes et détecter les problèmes en cas de trous ou de valeurs manquantes sur les courbes (Fig 1).

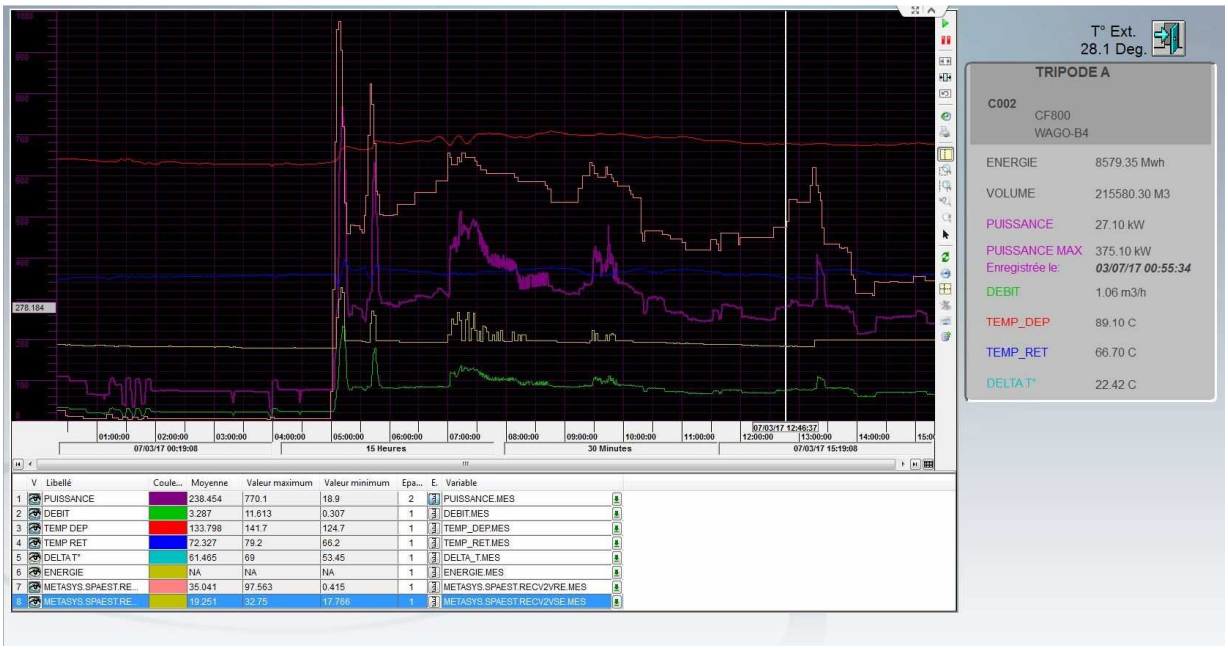


Figure 1: Aperçu des analyses sur le système PcVue

3.2 Analyse de l'historique

L'entrepôt contenant les données journalières et les historiques des systèmes contient un gros volume de données. Pour l'analyser nous avons utilisé la solution open source stack ELK de la société ELASTIC.

3.2.1 Pile ELK

ELK [5] est une pile logicielle composée d'Elasticsearch, Logstash et Kibana. Ces trois outils ont chacun un rôle bien précis dans le workflow permettant de rechercher, d'analyser et de visualiser, en toute fiabilité et sécurité, ainsi qu'en temps réel, des données issues de n'importe quelle source et sous n'importe quel format.

Elasticsearch

Un moteur de recherche et un outil de stockage distribué, utilisant le format JSON, conçu pour une scalabilité horizontale. C'est une base NoSQL qui est orientée Big Data il peut donc gérer un très grand volume de données.

Logstash

Logstash est un pipeline dynamique de collecte de données, doté d'un large écosystème de plugins et d'une forte synergie avec Elasticsearch. Les points d'entrée (input) utilisés pour aller chercher l'information sont définis via un fichier de configuration. Plusieurs types de point d'entrée peuvent être choisis, fichiers logs, bases de données et plusieurs points de sortie peuvent être définis et le plus utilisé c'est ElasticSearch.

Kibana

Kibana est une interface Web qui se connecte au cluster Elasticsearch, et permet de faire des requêtes en mode texte pour générer des graphiques (histogrammes, tableau de bord) ou des statistiques.

3.2.2 Analyse des données avec Kibana

Interface Discover

Grâce à la page DISCOVER de l'interface Kibana nous pouvons explorer les données de façon interactive. Cette interface se décompose en trois parties:

- Un Toolbar pour les recherches.

[5] https://www.elastic.co/fr/products

• Un histogramme pour voir la distribution des indexe dans le temps.

• Un tableau des documents qui comporte l'ensemble des indexes.

Par conséquent, nous pouvons accéder à chaque indexe qui représente un enregistrement de la base de donnée et qui est représenté sous forme de document JSON. Ainsi nous pouvons faire des requêtes de recherche, filtrer les résultats et afficher les données. Nous avons la possibilité finalement de configurer le champ du temps pour changer la distribution des documents au fil du temps dans l'histogramme et paramétrer la durée de rafraîchissement des données pour récupérer les données rajoutées à Elasticsearch comme le montre la figure 2

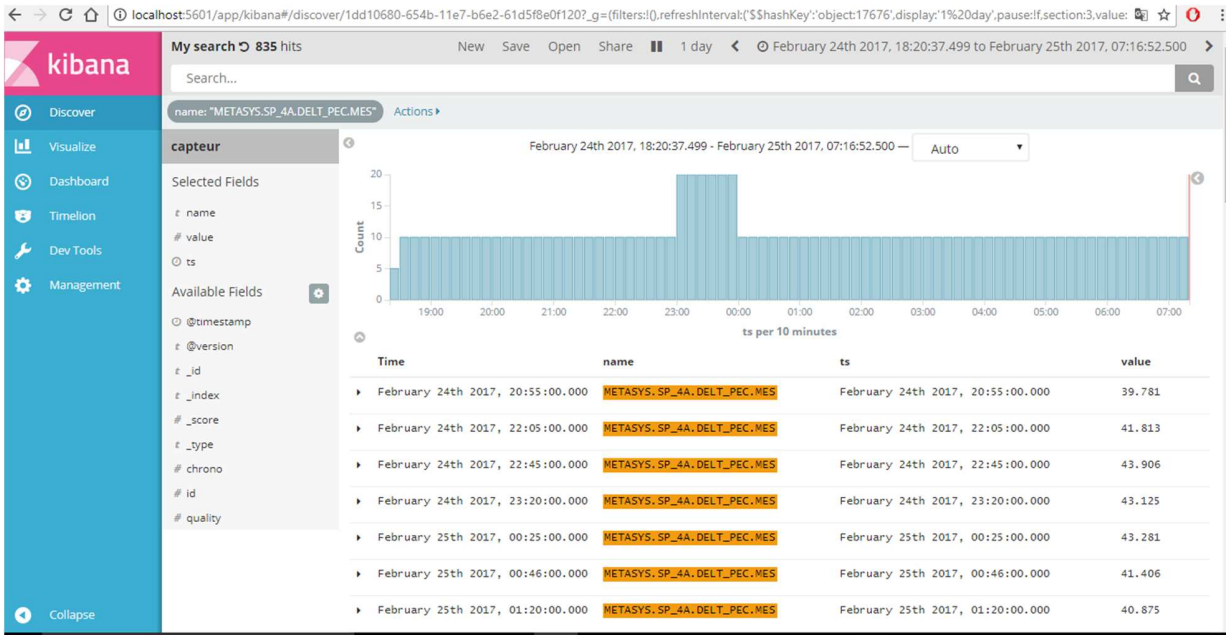


Figure 2: Aperçu de l'interface Discover de Kibana

Interface Tableau de bord

Le tableau de bord kibana affiche une collection de visualisations [17] crées et enregistrées. Nous pouvons ainsi organiser, redimensionner les visualisations aux besoins et enregistrer les tableaux de bord afin de les recharger et de les partager. Le tableau de bord représenté ci-dessous, destiné au responsable électromécanique et aux cadres supérieurs, détaille :

• Les courbes de tendances des capteurs/compteurs en valeurs moyennes en

fonction du temps .

• Le nombre de valeurs récupérées depuis la base de données.

• L'écart types des valeurs de capteurs/compteurs pour analyser la dispersion des valeurs capteur/compteur en fonction du temps.

• Un camembert graphique qui divise les capteurs selon leurs valeurs.

Sa consultation est très facile car il est doté d'une interface interactive, intuitive rapide et ergonomique. A travers cette interface, nous pouvons appliquer des filtres en modifiant les noms de capteurs ou en remontant dans l'intervalle de temps (Fig 3).

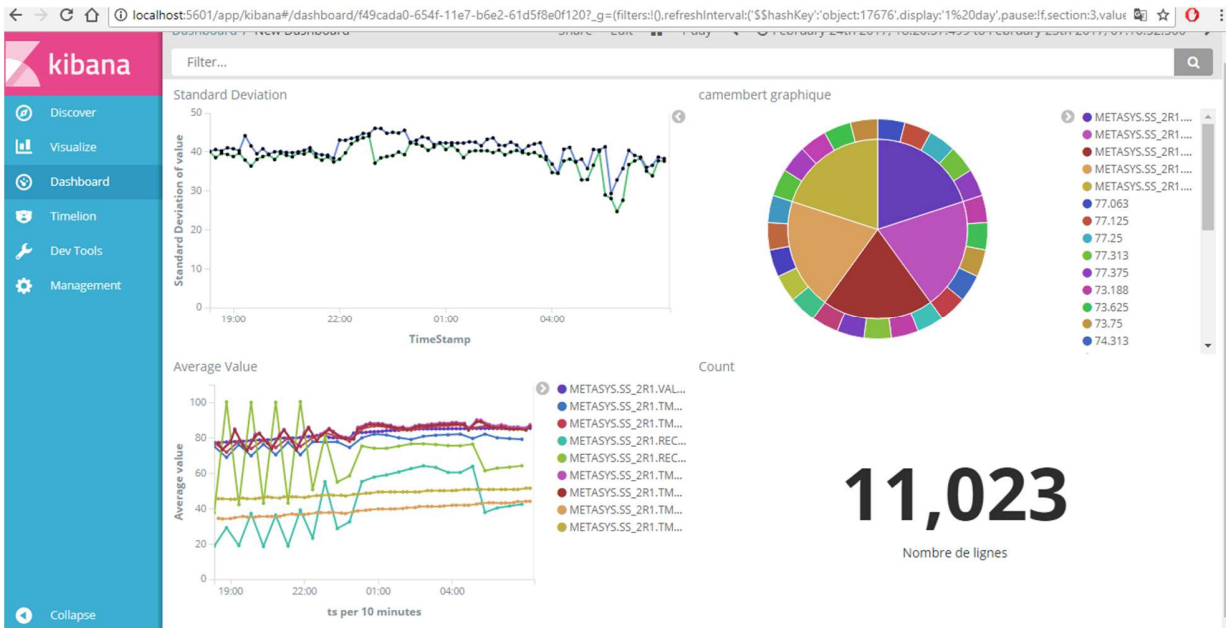


Figure 3: Aperçu d'un tableau de bord avec des données METASYS

4 Conclusion

Dans ce chapitre, nous avons présenté les analyses à l'aide du système de supervision PcVue et nous avons présenté la modélisation de l'application BI ainsi que son développement à travers l'enchaînement des différentes interfaces d'analyses décisionnelles.

CHAPITRE 6 : Conclusion et perspectives

Nous nous sommes intéressés au cours de ce travail à la modélisation et à l'inté- gration des données de capteurs/compteurs du Service de Gestion et d'Exploitation pour faciliter l'analyse et l'exploration des données de chaufferie grâce à un système décisionnel. Ce dernier ne se conçoit pas sans l'intermédiaire d'un entrepôt de données qui regroupe les données de capteurs provenant des systèmes de supervision METASYS et PcVue, c'est pour cette raison que nous avons eu recours aux bases de données décisionnelles.

En termes de réalisation, nous avons répondu amplement aux objectifs établis par le service malgré les difficultés rencontrées qui touchent principalement la complexité de systèmes de supervision. Une de ces difficultés, qui était la plus ardue, concerne la compréhension de la structure des sources de données et le choix de la structure pour intégrer les deux systèmes dans une seule base de données. Il nous a fallu en conséquence un temps assez important pour comprendre la structure des données et les systèmes énergétiques.

Au cours du développement de notre solution, nous avons détaillé en premier lieu les besoins de l'application grâce à des interviews avec les responsables du SGE. Puis, nous nous sommes tournés en second lieu vers la conception du modèle de données relatif à notre entrepôt de sorte que ce dernier réponde au mieux aux besoins des différents responsables. Nous avons alors proposé une solution qui comporte deux grandes étapes : l'une consiste à intégrer les données d'exploitation d'une durée de 6 mois dans une base de donnée SQL Server et l'autre consiste à concevoir une solution Big Data pour les données des historiques. La modélisation de la zone de stockage des données s'est faite grâce aux principes de la modélisation multidimensionnelle en adoptant un schéma en étoile. Nous avons par conséquent appliqué un processus d'extraction,de transformation et de stockage pour mettre en place cette solution. Cette partie d'ETL a été la partie du projet la plus fastidieuse et consommatrice en temps. Cette étape nous a permis de concevoir et de réaliser les routines d'extraction, transformation et chargement des données sous l'outil TOS (Talend Open Studio For Data Integration).

En dernier lieu, nous avons arrivé à consolider les données des deux systèmes via PcVue et nous avons mis en place l'outil Kibana qui permet d'analyser les données depuis la base de

données SQL Server en temps réel. Nous avons pu alors créer des courbes et des tableaux de bord qui facilitent l'analyse des données de capteurs/compteurs.

En guise de perspectives, notre solution a l'avantage d'être ouverte et extensible. Nous avons ainsi envisagé de compléter notre solution de système décisionnel le Big Data sur lequel nous pouvons appliquer l'analyse multidimensionnel à travers des outils Orienté Big Data et NoSQLà savoir Hadoop/HDFS, Hbase, Hive et Kylin pour l'interrogation.

CHAPITRE 7 : Références

[1] F. Ravat, O. Teste, R. Tournier, G. Zurfluh, **Algebraic and graphic languages for OLAP manipulations**. International Journal of Data Warehousing and Mining, IGI Publishing, D. Taniar, Vol. 4, N°1, p.17-46, 2008. doi: 10.4018/jdwm.2008010102

[2] F. Ravat, O. Teste, R. Tournier, G. Zurfluh, **Finding an Application-Appropriate Model for XML Data Warehouses**. Information Systems Journal, Elsevier Science Publisher, Vol. 36 N°6, p.662-687, 2010.doi:10.1016/j.is.2009.12.002

[3] F. Ravat, O. Teste, R. Tournier, G. Zurfluh, **Graphical Querying of Multidimensional Databases**. 11th East-European Conference on Advances in Databases and Information Systems (ADBIS'07), Springer-Verlag, LNCS 4690, Y.E. Ioannidis, B. Novikov, B. Rachev, p.298-313, Varna (Bulgaria), September 2007.

[4] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, R. Tournier, **How Can We Implement a Multidimensional Data Warehouse Using NoSQL?**, Lecture Notes in Business Information Processing (LNBIP), Vol. 241, Springer, S. Hammoudi, L. Maciaszek, E. Teniente, O. Camp, J. Cordeiro, ISBN 978-3-319-29132-1, p.108-130, 17th International Conference on Enterprise Information Systems (ICEIS'15) , Revised Selected Papers, Barcelona, Spain, April 27-30, 2015. doi: 10.1007/978-3-319-29133-8_6

[5] E. Annoni, F. Ravat, O. Teste, G. Zurfluh, **Towards Multidimensional Requirement Design**. 8th International Conference on Data Warehousing and Knowledge Discovery (DAWAK'06), Springer-Verlag, LNCS 4081, A. Min Tjoa, J. Trujillo, p.75-84, Krakow (Poland), September 2006.

[6] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, R. Tournier, **Benchmark for OLAP on NoSQL Technologies**, 9th International Conference on Research Challenges in Information Science (RCIS'15), p. 480-485, IEEE, Athens, Greece, April 2015.

[7] F. Ravat, O. Teste, **A Temporal Object-Oriented Data Warehouse Model**. 11th International Conference on Database and Expert Systems (DEXA'00), Springer-Verlag, LNCS 1873, M. Ibrahim, J. Jüng, N. Revell, p.583-592, London (UK), September 2000.

[8] O. Teste, **Towards Conceptual Multidimensional Design in Decision Support Systems**. 5th East-European Conference on Advances in Databases and Information Systems (ADBIS'01), Research Communications Vol.1, A. Caplinskas, J. Eder, p.77-88, Vilnius (Lithuania), September 2001.

[9] F. Ravat, O. Teste, G. Zurfluh, **Langages pour Bases Multidimensionnelles : OLAP-SQL**. Revue des Sciences et Technologies de l'Information, ISI (Ingénierie des Systèmes d'Information), Hermès, Vol.7, N°3, p.11-38, novembre 2002.

[10] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, R. Tournier, **Implementation of multidimensional databases in column-oriented NoSQL systems**, 19th East-European Conference on Advances in Databases and Information Systems (ADBIS'15), p.79-91, Poitiers, France, September 2015.

[11] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, R. Tournier, **Implementation of Multidimensional databases with document-oriented NoSQL**, 17th International Conference on Big Data Analytics and Knowledge Discovery (DAWAK'15), p.379-390, Valencia, Spain, September 2015.

[12] G. Pujolle, F. Ravat, O. Teste, R. Tournier, G. Zurfluh, **Multidimensional Database Design from Document-Centric XML Documents**. 13th International Conference on Data Warehousing and Knowledge Discovery (DAWAK'11), p.51-65, Toulouse (France), September 2011.

[13] Berro, I. Megdiche, O. Teste, **A Content-Driven ETL Processes for Open Data**, 18th East-European Conference on Advances in Databases and Information Systems (ADBIS'14), Proceedings II, New Trends in Database and Information Systems II, Vol. 312, p.19-40, Ohrid, republic of Macedonia, September 2014.

[14] Berro, I. Megdiche, O. Teste, **Graph-Based ETL Processes For Warehousing Statistical Open Data**, 17th International Conference on Enterprise Information Systems (ICEIS'15), p.271-278, Barcelona, Spain, April 2015.

[15] MF Canut, S On-At, A Péninou, F Sèdes, **Time-aware egocentric network-based user profiling**, Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 569-572, 2015.

[16] H Ezzedine, A Peninou, H Maoudji, **Towards Agents Oriented Specification of Human-Machine Interface: Application to Transport Systems**, IFAC Proceedings Volumes 34 (16), pp.363-368, 2001

[17] D Tchuente, MF Canut, NB Jessel, A Péninou, F Sedes, **Visualizing the relevance of social ties in user profile modeling**, Web Intelligence and Agent Systems: An International Journal Volume 10 (2), pp. 261-274, 2012.

[18] CA Zayani, A Péninou, MF Canut, F Sèdes **Towards an adaptation of semi-structured document querying**, Held in conjunction with the 6th International and Interdisciplinary Conference on Modeling and Using Context, 2007.