Nouvelle stratégie de collecte de données pour les compteurs d'eau communicants

Julien Spiegel

Université de Haute Alsace

École Doctorale Mathématiques, Sciences de l'Information et de l'Ingénieur
(MSII, ED 269)
Institut de Recherche en Informatique, Mathématiques, Automatique et Signal
(IRIMAS, EA 7499)

Nouvelle stratégie de collecte de données pour les compteurs d'eau communicants

Thèse

préparée par

Julien SPIEGEL

présentée pour obtenir le grade de
Docteur de l'Université de Haute Alsace
Discipline : Électronique, Électrotechnique et Automatique

soutenue publiquement le 5 septembre 2019 devant le jury composé de :

Pr. Danielle Nuzillard, Université de Reims Champagne-Ardenne, rapporteure
Pr. Jean-Marie Dilhac , INSA de Toulouse, rapporteur
Pr. Vincent Hilaire, Université de Bourgogne Franche-Comté, examinateur
Dr. Guy Bach, Diehl Metering SAS, co-encadrant
Dr. Gilles Hermann, Université de Haute Alsace, co-encadrant
Pr. Patrice Wira, Université de Haute Alsace, directeur de thèse

Remerciements

Tout d'abord, je souhaite exprimer ma sincère gratitude à mon directeur de thèse Pr. Patrice Wira pour son soutien continu dans ses trois années de recherche. En plus de Pr. Patrice Wira, je suis également très reconnaissant aux Drs Gilles Hermann et Guy Bach de m'avoir guidé et aidé quotidiennement. Je tiens à les remercier pour leurs encouragements et leurs commentaires constructifs.

J'en profite pour remercier les membres de mon comité de thèse : Pr. Danielle Nuzillard, Pr. Jean-Marie Dilhac, Pr. Vincent Hilaire, Dr. Guy Bach, Dr. Gilles Hermann et Pr. Patrice Wira pour avoir généreusement offert leur temps et leurs conseils tout au long de la révision de ce manuscrit.

Je saisis cette occasion pour exprimer mes sincères remerciements à toute l'entreprise Diehl Metering SAS pour sa confiance et pour cette opportunité de pouvoir préparer une thèse de doctorat au sein de cette entreprise. Je remercie mes collègues de l'université et toute l'équipe R&D de Diehl Metering pour leur soutien continu et leur sympathie.

Enfin, je souhaite remercier toute ma famille, et spécialement mes parents, pour m'avoir soutenu tout au long de mes études. Des remerciements spéciaux et une immense gratitude vont à ma charmante épouse qui m'a soutenu dans ces trois années.

Julien Spiegel
Mulhouse, 2019

Abstract

Water meters are the key element for the automated consumption reading in a distribution network and for a better managing of the consumptions. The research works of this thesis is focused on the development of a new smart water meter for communicating the consumption data autonomously. The autonomy of the water meter consists in self-managing its behavior, communication and energy. Effectively, energy is essential for powering these meters which are embedded systems. Actually, smart meters integrate a battery for the metering, processing and data transmission. The data transmission represents the major energy costs. The contributions of this thesis are organized in two main parts and for optimizing the energy embedded in smart water meters to enhance the monitoring of water consumption.

Firstly, a water meter has been equipped by an energy harvester for recharging its battery in order to increase its lifetime. The energy harvester device is able to extract energy from the water flow and from pressure fluctuations within the pipe. Secondly, a new communication strategy has been proposed and developed to be integrated in the embedded microcontroller. This communication strategy has been investigated for reducing the energy while transmitting data and reducing the processing time. A lossless data compression algorithm and an efficient energy management strategy have been proposed for sending the maximum amount of useful data and for reducing the length of transmitted frames.

The contributions have been evaluated through an experimental prototype. An energy balance has been performed under real operating conditions. Such experimental autonomous smart water meters have been set up in existing water distribution networks for continuously collecting water consumptions in a database. The validity of the collected data has been verified.

Key words: Water metering; smart meters; energy harvesting; lossless data compression; energy management strategy; consumption profile; communicating meter; autonomous meter; embedded calculation; automatic meter reading

Résumé

Les compteurs d'eau intelligents sont des éléments clés pour automatiser les relevés des consommations dans un réseau de distribution et pour optimiser la gestion des consommations. Le travail de cette thèse se concentre sur le développement d'un compteur d'eau intelligent qui transmet les relevés de manière autonome. L'autonomie est définie comme la capacité à transmettre automatiquement des informations tout en maîtrisant l'aspect énergétique. En effet, le besoin d'énergie est crucial pour un compteur d'eau communiquant afin d'alimenter l'électronique embarquée. Actuellement, les compteurs communicants utilisent une pile pour assurer la mesure, la mise en forme et la transmission des données. La transmission des données représente le poste de dépense énergétique le plus important. Les contributions de cette thèse comportent deux volets pour tendre vers une optimisation des performances énergétiques des compteurs tout en assurant la transmission des consommations d'eau.

Dans un premier temps, le compteur d'eau a été équipé d'un récupérateur d'énergie pour recharger sa batterie afin d'améliorer sa durée de vie. Le dispositif de récupération d'énergie est capable d'extraire l'énergie électrique provenant du flux d'eau et l'énergie hydraulique provenant des variations de pression dans la canalisation. Puis, une nouvelle stratégie de collecte de données a été proposée pour être embarquée dans le compteur. Cette stratégie de communication cherche à économiser l'énergie en réduisant les temps de traitement et de transmission. Un algorithme de compression sans perte et une gestion efficace de l'énergie ont été mis en place pour émettre le maximum de données tout en réduisant la longueur des messages.

Les contributions proposées ont été validées par un prototype expérimental. Un bilan énergétique a été réalisé pour des compteurs en conditions réelles de fonctionnement. Ces compteurs ont été insérés dans un réseau de distribution d'eau existant pour transmettre en continu les données de consommations vers une base de données. La validité des données transmises a été vérifiée.

Mots clefs : Comptage de l'eau ; compteurs intelligents ; récolte d'énergie ; compression de données sans perte ; stratégie de gestion énergétique ; profil de consommation ; compteur communicant ; capteur autonome ; calculs embarqués ; télé-relève

Table des matières

Liste des figures

Liste des tableaux

Introduction

La mesure et le comptage de l'eau font partis des enjeux majeurs dans les sociétés actuelles qui cherchent à gérer efficacement leur consommation [1]. La lecture automatisée des compteurs est requise pour optimiser une collecte efficace des données à distance. Les compteurs d'eau doivent donc être capables de transmettre des données plus efficacement et de manière totalement autonome. C'est le contexte des travaux de recherche exposés dans cette thèse qui s'intitule "**Une nouvelle stratégie de collecte de données pour les compteurs d'eau communicants**".

Collecter les données de consommation d'eau à distance est devenu une priorité pour les distributeurs et les fournisseurs d'eau. Ceux-ci souhaitent détecter le plus rapidement possible les éventuelles anomalies dans le réseau de distribution afin de réduire les coûts liés aux pertes d'eau. Bien que l'utilisation des technologies de lecture automatique des compteurs, en anglais *Automatic Meter Reading* (AMR), tend à croître depuis plus de vingt ans, de nombreux compteurs sont encore lus manuellement [2]. Par conséquent, les informations de comptage sont limitées et la facturation est difficile à établir en temps réel. C'est pourquoi les compteurs d'eau sont de plus en plus équipés de modules radio permettant de communiquer et transmettre les informations à distance par radiofréquence (RF). Les modules radio sont composés d'un système électronique doté d'un microcontrôleur de faible puissance, d'une mémoire RAM et d'un récepteur RF. Ce système embarqué est principalement alimenté par pile en raison des localisations variées et hostiles des compteurs d'eau (e.g., citernes, sous-sols, regards extérieurs). L'environnement des compteurs est souvent contraint par des conditions difficiles avec des variations importantes de température et un fort taux d'humidité. Sauf exception, l'alimentation filaire n'est pas envisageable car elle engendrerait des coûts supplémentaires importants. Ce système à pile doit assurer des capacités de traitement et de communication pour une durée de vie allant de dix à vingt ans. La durée de vie est principalement définie par le nombre de message transmis, mais aussi par le nombre de traitements effectués dans le microcontrôleur [3].

Les données sont collectées, analysées, puis transmises à une borne réceptrice RF afin d'être centralisées dans une base de données. Toutes ces fonctionnalités sont directement embarquées dans le compteur, ce qui rend le module radio complexe et peu modifiable. La chaîne de transmission est illustrée sur la Fig. 1. Intégrer les fonctionnalités dans le compteur représente un véritable challenge pour les fabricants, d'autant plus que la pérennité des compteurs est fortement menacée par des spécifications qui évoluent rapidement au fil du temps [4]. Les

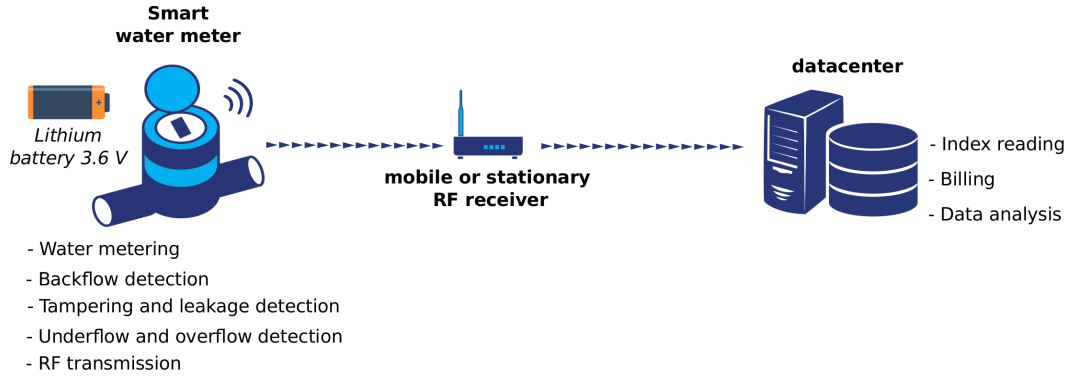


Figure 1 – Chaîne de communication générale entre un compteur d'eau intelligent et un serveur hébergeant une base de données

distributeurs d'eau requièrent de plus en plus d'information sur la consommation de l'eau, e.g., la durée d'une fuite, le volume d'eau perdu, le débit moyen entre deux émissions. Ils souhaitent avoir une mise à jour fréquente des données, disposer de davantage de services et avoir un système hautement sécurisé. C'est pourquoi, le nombre de transmissions augmente drastiquement et les messages deviennent de plus en plus longs. Ces contraintes affectent la durée de vie du compteur. Les fabricants doivent alors faire face à de véritables défis énergétiques. De manière générale, les distributeurs d'eau collectent les données une à cent fois par jour, ce qui limite la précision des données de consommation. Les enjeux actuels nécessitent une meilleure résolution de la consommation afin d'avoir une connaissance plus fine et plus juste des consommations dans le réseau de distribution. Il est donc essentiel d'optimiser la collecte et la transmission des données.

Dans le cadre de cette thèse, une nouvelle stratégie de collecte de données est proposée afin d'acquérir plus d'information sur les consommations et de les transmettre sans affecter la durée de vie du système embarqué qu'est le compteur. L'objectif de cette étude consiste à émettre plus de données pour proposer davantage de fonctionnalités aux distributeurs d'eau. Notre stratégie innovante a pour but de réduire la consommation du module radio et de déporter le traitement des données au niveau du serveur. Le fait de déporter le maximum de fonctionnalités permet également de réduire le nombre de tâches traité par le microcontrôleur embarqué. La volonté de transmettre plus de données à haute résolution et la nécessité de consommer moins d'énergie, nous pousse à nous intéresser aux algorithmes de compression. Ceux-ci ont pour objectif d'obtenir le meilleur compromis entre le taux de compression, la dégradation éventuelle des données et le temps de traitement.

La stratégie choisie consiste d'une part à réduire la consommation d'énergie du module radio et d'autre part transmettre l'information brute provenant du capteur pour pouvoir faire plus d'analyses. Nous proposons donc un nouveau compteur communicant, avec une collecte de données plus fine, et une transmission optimisée. La métrologie du compteur reste inchangée et ne fait pas l'objet de cette étude. Un système expérimental a été mis en place en conditions réelles dans plusieurs réseaux de distribution afin de valider la transmission des données.

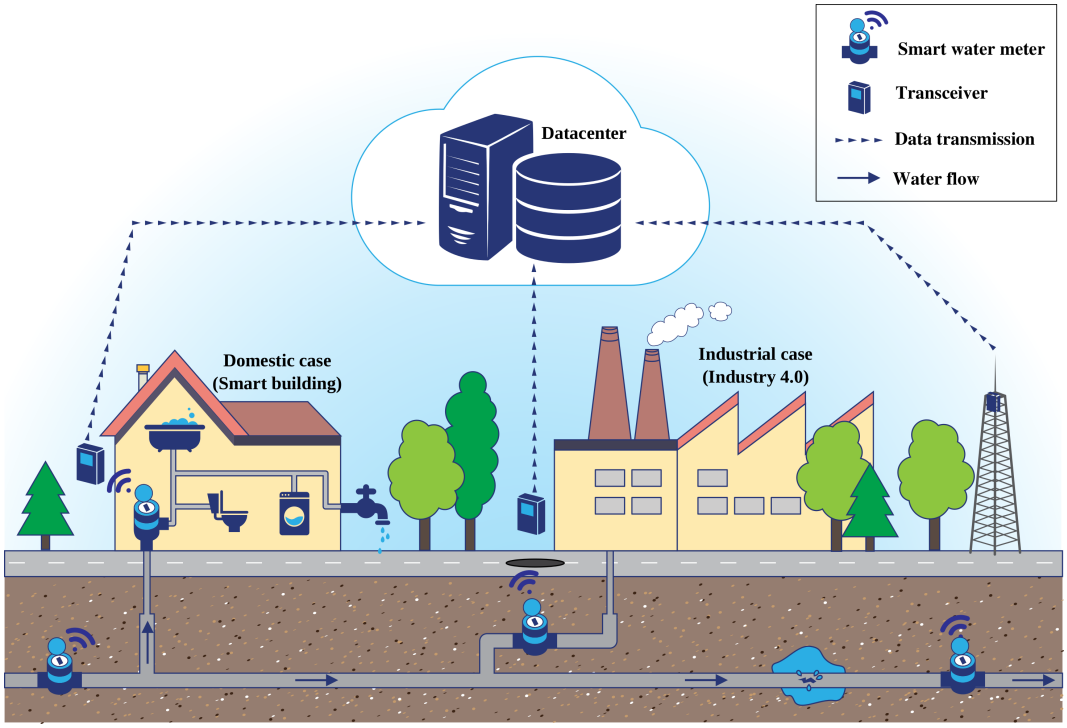


Figure 2 – Contexte général des compteurs d'eau communicants pour améliorer l'analyse de la consommation et la gestion du réseau de distribution

L'objectif consiste à transmettre les données mesurées par le capteur sans perte d'information. La Fig. 2 illustre le contexte de cette nouvelle stratégie de collecte de données. Celle-ci permet de simplifier le système embarqué et d'offrir de nouvelles fonctionnalités sur le serveur, telles que, la détection de fuites et de fraudes, la reconstruction de l'historique de consommation, la détection de particules dans la canalisation ou encore l'analyse de vieillissement des compteurs. Par exemple, les fuites vont pouvoir être plus facilement détectées et plus rapidement localisées.

Publications et contributions scientifiques

Cette thèse a été réalisée au sein de la société Diehl Metering SAS et en collaboration avec l'Université de Haute Alsace à Mulhouse. L'originalité de cette thèse industrielle (CIFRE) a été de regrouper des notions physiques et algorithmiques liées à la mesure et à la transmission des données de comptage. Les travaux de recherche ont permis de contribuer dans quatre domaines différents afin de pallier aux contraintes énergétiques et d'acquérir plus d'information sur les consommations d'eau. Les contributions de cette thèse peuvent être résumées comme suit :

Contribution 1 : réalisation de deux prototypes pour la récupération d'énergie dans les compteurs d'eau.
Ces prototypes ont été développés afin de récupérer les énergies mécaniques présentes dans l'environnement des compteurs d'eau. Les résultats obtenus sont présentés dans le **Chapitre 1** et ont fait l'objet de publications dans des articles publiés dans des conférences nationales et internationales [5], [6]. Un état de l'art sur la récupération d'énergie dans l'environnement des compteurs d'eau a également été soumis dans un journal scientifique [7].

Contribution 2 : proposition et développement d'une nouvelle stratégie de collecte de données. Cette contribution est complémentaire à la première et elle permet de répondre aux enjeux actuels. Cette contribution offre plus de fonctionnalités dans l'analyse de la consommation de l'eau car la stratégie proposée permet de déporter le maximum de données sur le serveur. Cette stratégie est présentée dans le **Chapitre 1** et sera développée tout au long du manuscrit. Cette contribution ouvre plus largement des perspectives pour la collecte et l'analyse d'autres grandeurs physiques (**Chapitre 4**).

Contribution 3 : étude et développement d'un nouvel algorithme de compression sans perte. La troisième contribution a permis de transmettre le maximum de données sur le serveur en compressant les données brutes provenant du capteur sans pertes d'information. Un nouvel algorithme de compression sans perte, appelé *Run-Length Binary Encoding* (RLBE), a été proposé, développé et validé expérimentalement, permettant alors de transmettre davantage de données tout en réduisant la consommation d'énergie de l'électronique embarqué. Les résultats sont présentés dans le **Chapitre 2** et ont fait l'objet de publications dans des articles publiés dans des conférences nationales et internationales [8], [9], [10].

Contribution 4 : mise en place d'un prototype complet de collecte de données pour le stockage et l'analyse de la consommation de l'eau (**Chapitre 3**).
Ce prototype permet d'exploiter des données issues de plusieurs compteurs communicants dans un contexte de télé monitoring pour bâtiments intelligents [11]. L'ensemble de ces données permettra de caractériser et de modéliser les usagers et les usages en terme de consommation d'eau.

Les travaux de recherche menés dans cette thèse ont été validés par les publications suivantes.

Poster présenté lors de congrès scientifiques

- Julien Spiegel, Gilles Hermann, and Patrice Wira. Towards autonomous smart water meters. Journée des Écoles Doctorales à l'Université de Haute Alsace, Mulhouse, 2018 [8]
- Julien Spiegel, Gilles Hermann, and Patrice Wira. Towards autonomous smart water meters. Upper Rhine Cluster for Sustainability Research - International Conference (URCforSR 2018), Strasbourg, 2018 [9]

Articles publiés lors de conférences nationales

- Julien Spiegel, Gilles Hermann, et Patrice Wira. La récolte d'énergie et son application dans le comptage de l'eau. Congrès National de la Recherche des IUT (CNRIUT 2017), Auxerre, 4-5 mai 2017 [5]
- Aida Boudhaouia, Julien Spiegel, et Patrice Wira. Recueil et exploitation de données issues de capteurs connectés dans un contexte de télémonitoring pour batiments intelligents. Congrès National de la Recherche des IUT (CNRIUT 2018), Aix-en-Provence, 7-8 juin 2018 [11]

Articles publiés lors de conférences internationales

- Julien Spiegel, Patrice Wira, and Gilles Hermann. Energy efficiency optimization in fluid flow metering. 19th International Conference of the IEEE Industrial Technology (ICIT 2018), Lyon, 2018 [6]
- Julien Spiegel, Gilles Hermann, and Patrice Wira. A comparative experimental study of compression algorithms for enhancing energy efficiency in smart meters. 16th International Conference of the IEEE Industrial Informatics (INDIN 2008), Porto, 2018 [10]

Article soumis dans des journaux scientifiques

- Julien Spiegel, Patrice Wira, and Gilles Hermann. Energy harvesting and its application in water metering: A state-of-the-art. Sensors, soumis, 2019 [7]

Plan général de la thèse

Ce rapport de thèse est composé de 4 chapitres qui sont structurés comme suit.

- Le **chapitre 1** présente un état de l'art sur les compteurs d'eau actuels et sur leurs capacités à transmettre les données. Les enjeux et la question énergétique sont rapidement abordés. Un nouveau type de compteur est alors proposé afin de répondre aux enjeux présentés. Il s'agit de pouvoir disposer sur le serveur de données de consommation aussi précises que celles mesurées par le compteur et avec un nombre d'échantillons aussi grand que possible. Les expérimentations ont validé la faisabilité de l'approche proposée. Ce nouveau compteur a pour objectif de déporter le traitement des consommations sur le serveur et de compresser les données pour économiser l'énergie.
- Le **chapitre 2** décrit les techniques de base les plus couramment utilisées dans la compression de données pour les systèmes embarqués. Dans un premier temps, nous avons considéré les algorithmes de compression sans perte tels que les codages Huffman, Even-Rodeh, Exponentiel-Golomb, Lempel-Ziv Welch (LZW), Fibonacci et l'algorithme hybride Bzip2. Une nouvelle approche de compression, appelée Run-Length Binary Encoding (RLBE), est ensuite proposée afin de combler les inconvénients de ces algorithmes. Une étude expérimentale est menée permettant de comparer leurs performances à partir de données réelles de consommation. Les résultats mettent l'accent sur le taux de compression obtenu et le temps de traitement nécessaire. Un bilan énergétique est effectué et les résultats expérimentaux démontrent une optimisation efficace de l'énergie consommée par le module radio.
- Le **chapitre 3** propose d'exploiter les données de consommation à l'aide d'une plate-forme expérimentale qui est composée de plusieurs compteurs, de récepteurs et d'une base de données. Cette plate-forme expérimentale permet d'extraire les données compressées à partir de plusieurs compteurs installés dans le réseau de distribution. Le décodage et le stockage des données ont été développés pour permettre une analyse plus approfondie des informations de comptage. Ce chapitre donne un aperçu du fort potentiel que fournit cette nouvelle stratégie de collecte de données.
- Le **chapitre 4** développe une généralisation de la collecte de données semblable à la mesure de l'eau. Cette généralisation permet d'étendre les contributions apportées au cours de cette thèse au comptage d'autres grandeurs physiques (pression, température, volume de gaz, etc.). Une structure universelle de données est proposée dans ce chapitre afin d'offrir plus de souplesse dans le choix des algorithmes, de la résolution, dans la quantité de données à transmettre, dans la technologie utilisée pour envoyer les données, etc. Basé sur les contributions du chapitre 2, notre objectif consiste à compresser dynamiquement les données de consommation. Ce chapitre fait l'objet d'une étude des compromis entre le temps de traitements nécessaire pour compresser les données, la taille des messages et l'affectation des données.

1 Comptage et relève appliqués à la consommation d'eau

Un état de l'art sur le comptage actuel de l'eau est présenté dans ce chapitre. Les compteurs et les systèmes de relevé automatique sont présentés afin de définir les contraintes rencontrées par les services des eaux. Une nouvelle génération de compteur est proposée dans le but de répondre aux enjeux actuels. Les notions de *courbe de charge* et de *profil de consommation* sont également introduites car elles sont importantes à l'analyse des consommations d'eau.

1.1 Les compteurs d'eau actuels : état de l'art

1.1.1 Architecture générale des compteurs

Un compteur d'eau est un appareil qui permet de mesurer le volume d'eau qui transite dans une canalisation. Cet appareil est un instrument de mesure qui est utilisé pour facturer la consommation de l'eau et pour maîtriser la distribution dans un réseau. Afin d'automatiser la collecte des données, économiser les coûts de relève, et améliorer la gestion du réseau de distribution, les compteurs d'eau sont équipés d'un circuit électronique embarqué. Ce dernier comprend entre autre un module de communication radio et permet de traiter et de transmettre les données.

La Fig. 1.1 montre les trois parties principales d'un compteur d'eau communicant :

- le **mesureur** qui est composé d'un capteur immergé dans l'eau pour mesurer le volume d'eau consommé par unité de temps ;
- le **totalisateur**, mécanique ou électronique, qui permet de totaliser le volume détecté et de l'afficher ;
- l'électronique embarquée qui est composée d'un **module de communication radio**, pour traiter et transmettre les informations mesurées.

Un compteur d'eau seul n'est composé que des deux premières parties, le mesureur et le totalisateur. Pour transmettre et communiquer les informations de comptage, un module

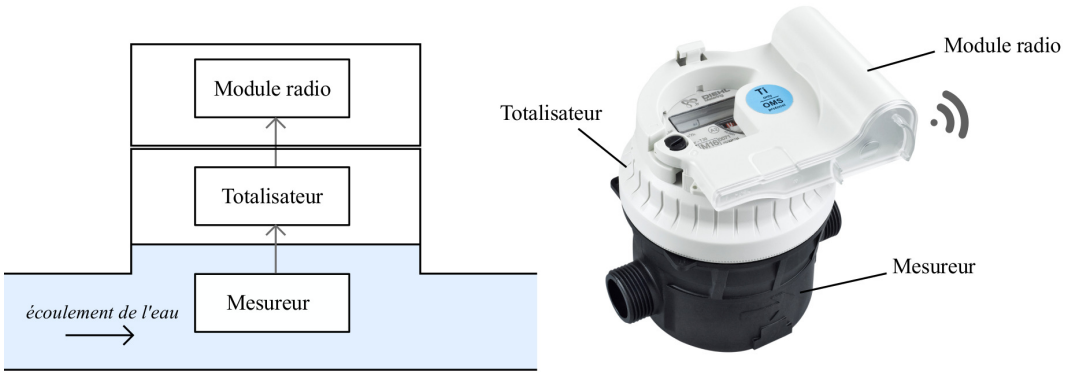


Figure 1.1 – Architecture générale d'un compteur d'eau, a) principe et b) illustration d'un compteur d'eau mécanique communicant

radio s'interface sur le totalisateur. Ce module radio contient toute l'électronique du système embarqué, soit le microcontrôleur, la RAM, l'émetteur radio, etc. La Fig. 1.1 montre un compteur modulaire où le module radio est clipsé sur le compteur d'eau.

Les compteurs d'eau sont classés en deux familles différentes : les compteurs mécaniques et les compteurs statiques.

Les compteurs mécaniques sont équipés d'une partie mobile et d'un transducteur mécanique. Parmi ces compteurs, on distingue deux types différents : les compteurs de vitesse et les compteurs volumétriques. La Fig. 1.2 illustre le principe de fonctionnement d'un compteur de vitesse à jet unique. La vitesse d'écoulement est mesurée à l'aide d'une turbine afin de déterminer le volume d'eau qui passe. La conception de la turbine joue un rôle important car elle différencie les compteurs à jet unique des compteurs à jet multiple. Les compteurs volumétriques, également appelés compteurs de déplacement, utilisent le déplacement d'un piston dans une chambre comme illustré sur la Fig. 1.3. Le volume d'eau consommé est déterminé par le nombre de cycles effectué par le piston dans la chambre. Les conditions d'utilisation de ces compteurs sont variées mais ils sont particulièrement appréciés pour leur capacité de mesure à faibles débits. La partie mobile du mesureur est reliée à un axe rotatif sur lequel est placé un aimant permanent. Le champ magnétique de ce dernier entraîne la rotation d'un autre aimant raccordé au totalisateur. Au-dessus du totalisateur, un demi-disque métallisé est utilisé comme témoin de la rotation de la partie mobile. L'électronique embarquée est équipée d'un capteur inductif permettant de détecter la position du demi-disque [12].

Les compteurs statiques ne sont pas équipés de partie mobile et sont totalement électroniques [13]. Le principe de mesure utilisé (électromagnétique ou ultrasonique) est directement intégré dans l'électronique embarquée. Les compteurs statiques utilisent des technologies qui leurs permettent d'être insensibles à la présence de sable et de particules dans la canalisation [14], [15].

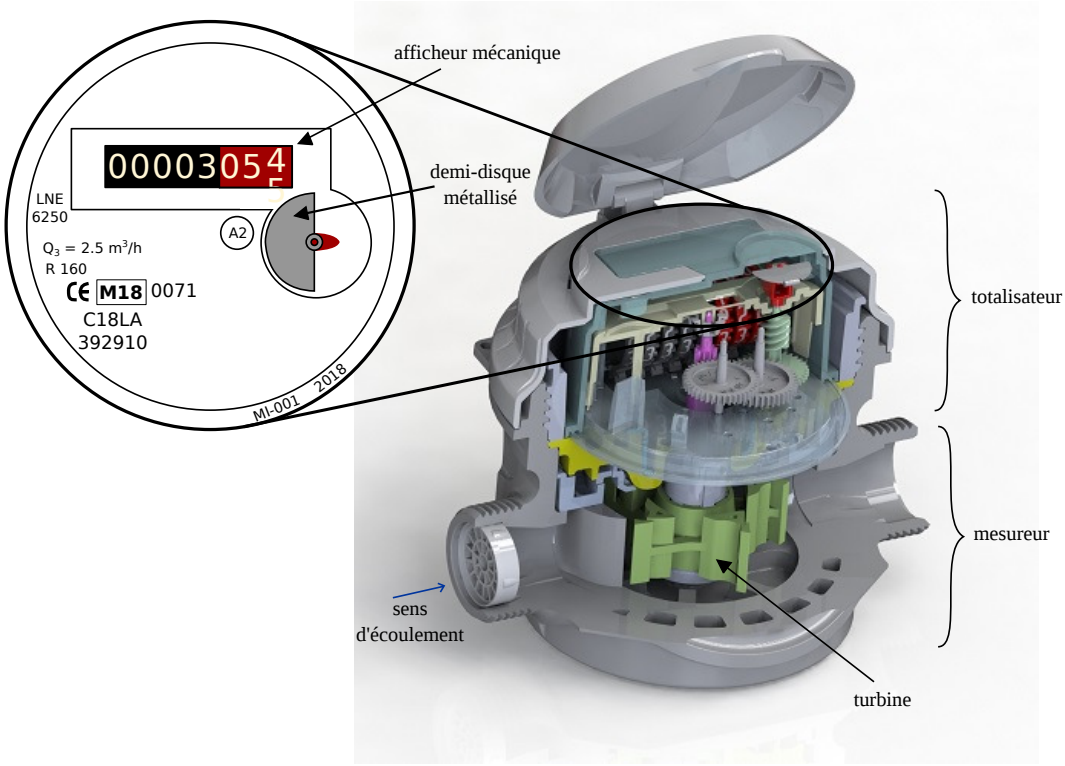


Figure 1.2 – Vue en coupe détaillée d'un compteur de vitesse à jet unique sans module radio

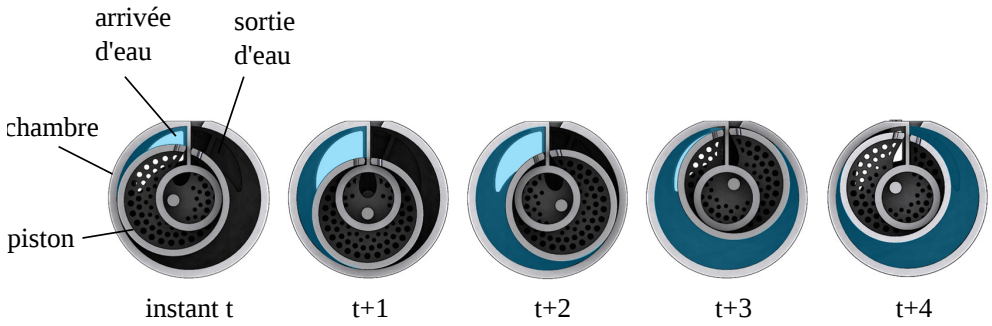


Figure 1.3 – Principe de fonctionnement d'un compteur d'eau volumétrique

1.1.2 Caractéristiques métrologiques

Les compteurs d'eau sont classifiés en fonction de leur métrologie et de leur précision de mesure. Les caractéristiques métrologiques peuvent varier selon les zones géographiques et les pays. Par exemple aux États-Unis, le développement des compteurs d'eau est décrit par des exigences recommandées par l'association *American Water Works Association* (AWWA) [16]. En Europe, la métrologie légale est strictement réglementée par la directive *Measuring Instruments Directive* (MID) pour les instruments de mesure [17]. Cette directive est considérée comme une référence dans de nombreux pays dans le monde.

La MID concerne essentiellement les compteurs d'eau utilisés pour la facturation dans les secteurs résidentiels, commerciaux et industriels. Les installations et les conditions d'utilisations sont définies dans la norme EN 14154-1. Cette dernière permet de spécifier la métrologie légale en Europe pour la production des compteurs d'eau. Bien que la métrologie d'un compteur soit indirectement liée à sa technologie, le choix d'un compteur d'eau est principalement défini en fonction de sa plage de mesure. Comme illustré sur la Fig. 1.4, cette plage est déterminée par quatre points de mesures relatifs aux débits réglementaires, appelés Q_1, Q_2, Q_3 et Q_4 :

- Q_1 est le débit de démarrage correspondant au plus petit débit détecté par le compteur d'eau ;
- Q_2 est le débit de transition entre Q_1 et Q_3. Il divise la plage de mesure en deux zones différentes qui sont utilisées pour définir le canal de tolérance ;
- Q_3 est considéré comme le débit le plus élevé attribué dans les conditions de fonctionnement permanentes ;
- Q_4 est considéré comme la limite réglementaire pour la mesure de l'eau. Les débits mesurés au-delà de Q_4 sont des débits de surcharge. Le compteur d'eau n'est pas conçu pour fonctionner en permanence dans ces conditions.

Les compteurs sont classés en fonction de leur section interne S, appelée Diamètre Nominal (DN). La section S (m^2) et la vitesse d'écoulement de l'eau v (m/s) permettent de calculer le débit instantané $Q = v \cdot S$ (m^3/s).

La métrologie d'un compteur d'eau est déterminée par une courbe métrologique qui doit être incluse dans un canal de tolérance. Le canal de tolérance contient les quatre débits précédents et est limité par l'erreur maximale tolérée (en anglais, *Maximum Permissible Error (MPE)*). La Fig. 1.4 montre qu'une erreur de mesure maximale de ±5% est tolérée entre Q_1 et Q_2 et ±2% entre Q_2 et Q_4. L'erreur de mesure ξ, exprimée en pourcentage, est calculée avec le volume mesuré d'eau V_m et le volume réel d'eau consommée V_r :

$$\xi = \frac{V_m - V_r}{V_r} \cdot 100. \tag{1.1}$$

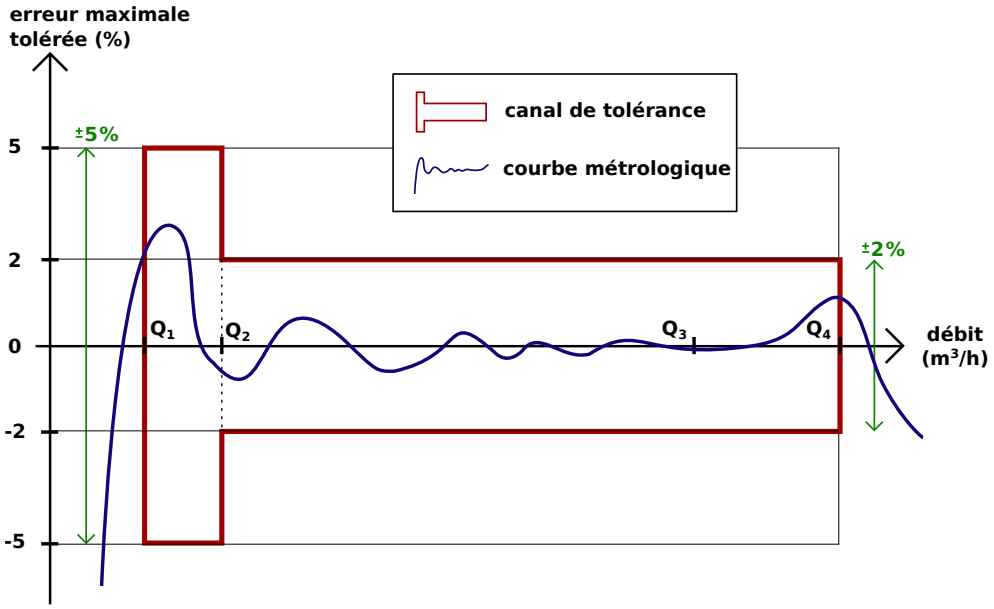


Figure 1.4 – Erreur maximale tolérée en fonction du débit dans le comptage de l'eau selon la norme EN 14151-1

Un compteur d'eau est principalement caractérisé par sa plage de mesure $R = \frac{Q_3}{Q_1}$. La plage de mesure est un critère important dans le choix du compteur. Par ailleurs, un compteur d'eau ne doit pas engendrer de pertes de charge importante. En effet, la perte de charge, correspondant à la différence de pression entre l'entrée et la sortie du compteur, est normalisée et limitée conformément à la MID. Typiquement, un compteur d'eau ne doit pas générer une perte de pression supérieure à 0,63 bar au débit nominal Q_3 et 1 bar à Q_4. Si ces caractéristiques métrologiques sont respectées, le compteur d'eau est conforme à la MID et donc commercialisable en Europe et dans de nombreux autres pays du monde.

1.1.3 Électronique embarquée

L'environnement des compteurs d'eau ne favorise pas toujours un accès facile pour le relevé manuel. C'est pourquoi depuis une vingtaine d'années, les compteurs d'eau sont majoritairement équipés de modules radio. L'électronique embarquée dans le module radio comprend un microcontrôleur de faible puissance, un émetteur radio et un circuit électronique nécessaire pour communiquer les informations de comptage à distance et de manière autonome. Un schéma bloc montre les différentes parties d'un module radio utilisé dans cette étude sur la Fig. 1.5. Un firmware est intégré dans le microcontrôleur afin de réaliser tous les traitements nécessaires sur les données mesurées.

Les compteurs communicants utilisent des techniques de relevés automatique AMR (Automatic Meter Reading) et AMI (Advanced Metering Infrastructure) [18], [19]. Ces technologies sont utilisées dans des applications de comptage telles que les compteurs électriques, de gaz et

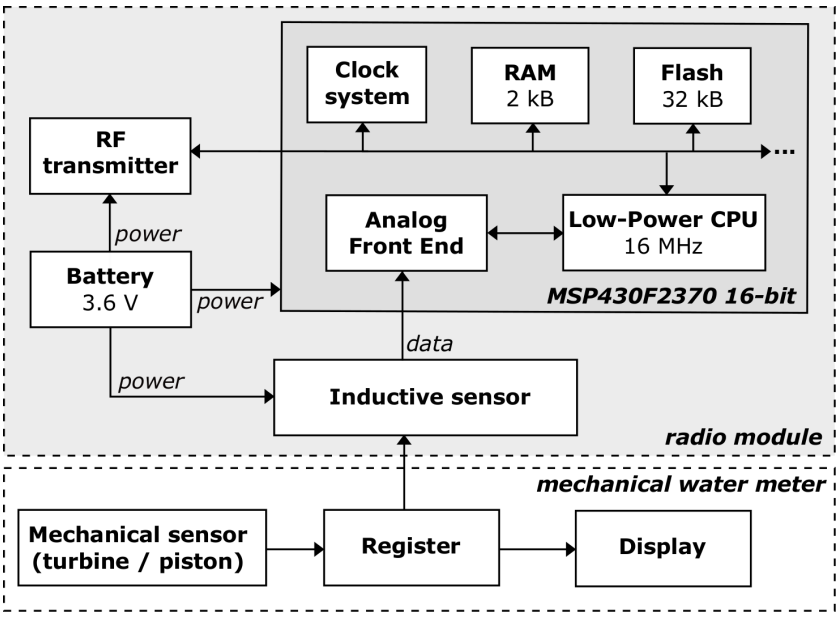


Figure 1.5 – Schéma bloc d'un module radio IZAR RCi de Diehl Metering et utilisé pour les tests expérimentaux

d'énergie thermique [20]. La facturation est effectuée sur une consommation réelle plutôt que sur des estimations basées sur les consommations passées. En raison de sa rentabilité et de sa fiabilité, la communication par radiofréquence (RF) est le moyen de communication privilégié des compteurs. Les solutions de relève automatique peuvent être effectuées par réseau mobile (Walk-By/Drive-By) mais également par réseau fixe (plus connu sous l'appellation de télé-relève). Ces architectures réseaux sont illustrées sur la Fig. 1.6. Bien qu'il existe une multitude de protocoles radios (Zigbee, Sigfox, LoRa, etc.), le standard Wireless M-Bus est l'un des plus utilisé dans le domaine du comptage de l'eau [21], [22].

Standard Wireless M-Bus

Le standard Wireless M-Bus, également appelé Wireless Meter-Bus (w-MBus), spécifie la communication RF entre les compteurs communicants, les concentrateurs et les enregistreurs de données en Europe [23]. Cette norme est utilisée pour la connectivité sans fil dans les systèmes AMR/AMI. Les fréquences les plus couramment utilisées sont 868 MHz, 433 MHz et 169 MHz en Europe. Ces différentes fréquences sont spécifiées selon plusieurs modes avec des communications unidirectionnelles et bidirectionnelles. Les systèmes AMR sont conçus pour collecter les données de consommation de manière unidirectionnelle, tandis que les systèmes AMI utilisent des communications bidirectionnelles entre les compteurs et le centre de données. La communication bidirectionnelle permet par exemple d'obtenir des informations complémentaires et/ou de mettre à jour le firmware du module radio.

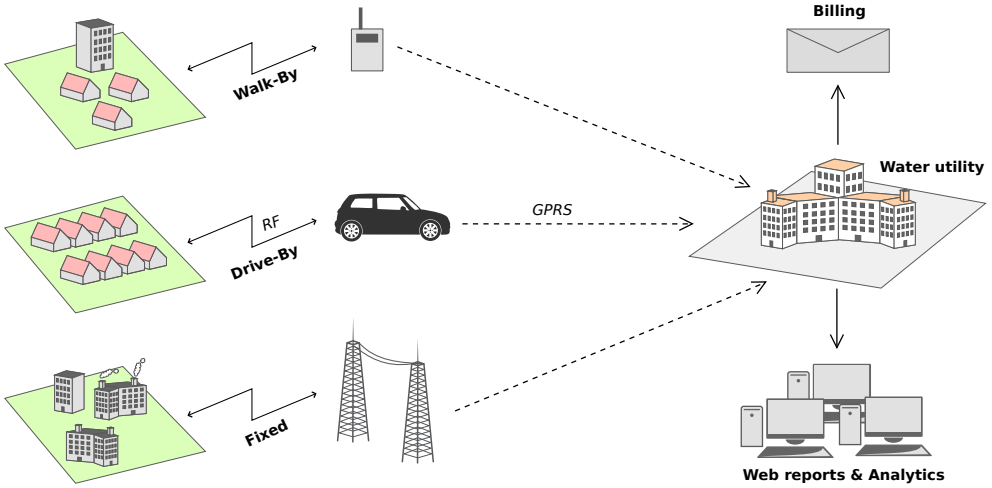


Figure 1.6 – Types de communication des technologies AMR/AMI utilisées dans le comptage de l'eau

Module de communication RF

Le module de communication RF est composé d'un microcontrôleur de faible puissance, d'une mémoire RAM et d'un récepteur RF. L'objectif de ce système embarqué consiste à mesurer et analyser les données de consommation, détecter les anomalies, puis de les transmettre.

Le système embarqué est principalement alimenté par pile en raison de la localisation variée des compteurs d'eau (e.g., citernes, sous-sols, regards extérieurs). L'environnement des compteurs est souvent contraint par des conditions difficiles avec des variations importantes de température et un fort taux d'humidité. Sauf exception, l'alimentation filaire n'est pas envisageable car elle engendre des coûts supplémentaires importants. La pile définie la durée de vie de l'électronique embarquée (une quinzaine d'années en moyenne). La consommation du module radio est principalement déterminée par la transmission des données. Comme illustré sur la Fig. 1.7, le module radio opère sous deux modes de fonctionnement : le mode veille et le mode communication. La communication est effectuée dans une durée t_E de seulement quelques millisecondes avec une puissance P_{Max} de plusieurs milliwatts. Par ailleurs, quelques microwatts sont encore nécessaires entre deux communications pour maintenir le compteur en mode veille. Ce mode est dédié uniquement au comptage et au traitement des données. La consommation moyenne P_{Avg} est déterminé par l'intervalle de transmission T et par le nombre de message émis.

L'électronique est conçue de manière à limiter la consommation d'énergie afin de respecter la durée de vie initialement définie. Au cours de cette thèse, les tests expérimentaux seront effectués sur un module de communication RF commercialisé par la société Diehl Metering. Ses caractéristiques sont présentées dans le Tableau 1.1.

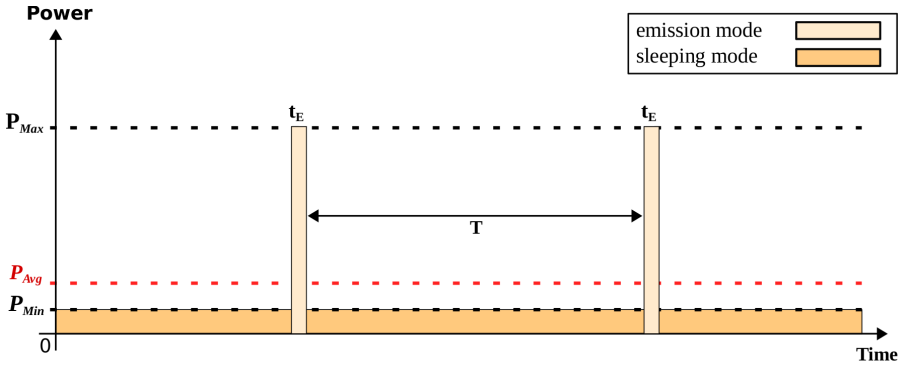


Figure 1.7 – Illustration de la consommation d'énergie lors du mode veille et du mode de communication pour les compteurs d'eau intelligents

Tableau 1.1 – Caractéristiques techniques du module radio utilisé pour les tests expérimentaux

Microcontrôleur	Texas Instruments MSP430F2370 16-bit
Mémoire flash	32 kB
Mémoire RAM	2 kB
Fréquence RF	868 MHz
Portée RF en champs libre	≈ 1 km
Consommation moyenne P_{Avg}	58 μW
Type de capteur	capteur inductif

1.1.4 Performances et limites

Le module radio permet de transmettre les données de consommation à distance et de manière totalement autonome. Les données sont encapsulées dans une trame avant d'être transmises à un récepteur RF. Comme le montre la Fig. 1.8, le récepteur est considéré comme une passerelle qui permet de centraliser les données collectées vers un serveur. Toutes les fonctionnalités (analyses, détection d'anomalies, etc.) sont directement embarquées dans le module radio, ce qui le rend complexe et peu modifiable. De plus, les distributeurs d'eau souhaitent disposer de plus en plus d'information sur la consommation de l'eau, e.g., la durée d'une fuite, le volume d'eau perdu, le débit moyen entre deux émissions. Par conséquent, le nombre de calculs augmente drastiquement et les messages deviennent de plus en plus longs. Ces contraintes affectent la durée de vie du système et les fabricants doivent faire face à de véritables défis énergétiques.

La structure générale des trames envoyées est illustrée sur la Fig. 1.9. Cette structure est basée sur le standard EN 13757-4. Celle-ci est composée d'un en-tête *HEADER*, d'un champ *PAYLOAD* contenant les données utiles, et d'un champ *CRC* réservé aux contrôles d'erreur. Le contenu du champ *PAYLOAD* est généralement composé de :

- l'index correspondant au volume total d'eau consommée ;
- les éventuelles alarmes traitées par le microcontrôleur ;
- les informations complémentaires liées aux besoins spécifiques.

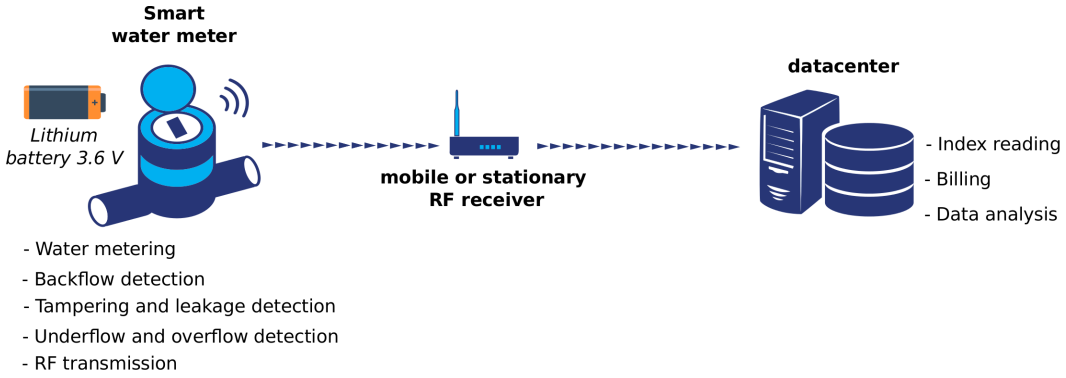


Figure 1.8 – Collecte de données actuelle entre les compteurs d'eau et le centre de données

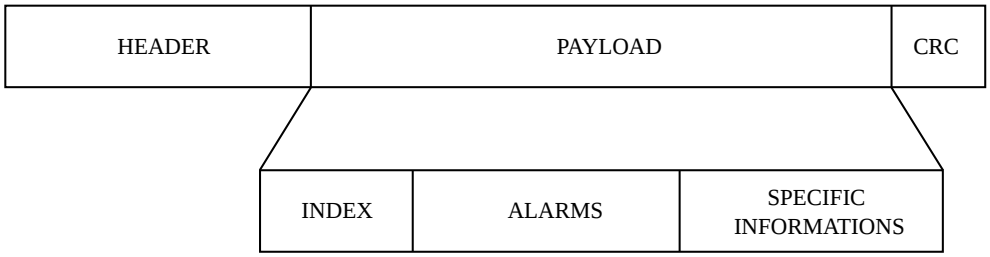


Figure 1.9 – Structure de trame générale respectant le standard européen EN-13757-4

Bien que plusieurs types de trames différentes peuvent être envoyés, l'index doit obligatoirement figurer dans la trame utilisée pour la facturation de la consommation. La consommation de l'eau est un processus continu qui est régulièrement mesuré au cours du temps. Défini par $I(t_i)$, l'index représente la consommation totale d'eau en litres depuis l'installation du compteur. A l'instant t_i, $I(t_i)$ est exprimé comme suit, avec $V(t_i)$ le volume d'eau consommé entre t_i et t_{i-1} :

$$I(t_i) = \sum_{i=1}^{n}(V(t_i)). \tag{1.2}$$

La courbe de charge $C(t_i)$ est représentée par la série des valeurs de l'index sur une période de temps [24] :

$$C(t_i) = \{I(t_i)\}_{i=1...n} \tag{1.3}$$

$C(t_i)$ est reconstituée au niveau du serveur pour analyser la consommation d'eau. La Fig. 1.10 montre la reconstitution de la courbe de charge, avec l'index $I(t_i)$ correspondant à la mesure effectuée à l'instant t_i. L'intervalle de transmission entre l'index $I(t_i)$ et $I(t_{i+1})$ est défini de manière à ne pas engendrer des coûts énergétiques importants. Généralement, les distributeurs

d'eau collectent les données une à cent fois par jour, ce qui limite la précision de la courbe de charge. Avec cette résolution des données il n'est pas possible de détecter toutes les anomalies dans le réseau de distribution. Les enjeux actuels nécessitent une meilleure résolution afin d'avoir une connaissance plus fine et plus juste des consommations.

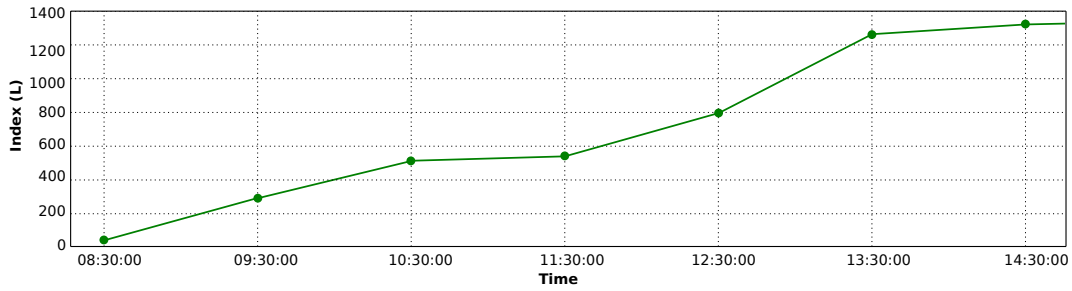


Figure 1.10 – Exemple d'une courbe de charge établie à partir des index collectés toutes les heures

1.2 Enjeux actuels

1.2.1 Motivations et objectifs

La finalité principale des compteurs intelligents est d'offrir une meilleure gestion de l'eau et une facturation plus juste. La collecte automatique des données concerne quatre entités différentes : les fournisseurs, les distributeurs, les sous-distributeurs et les consommateurs finaux. Ceci est illustré sur la Fig. 1.11. Les objectifs et les exigences peuvent différer pour chacun de ses acteurs. Les principaux objectifs de ces quatre entités sont décrits dans le Tableau 1.2.

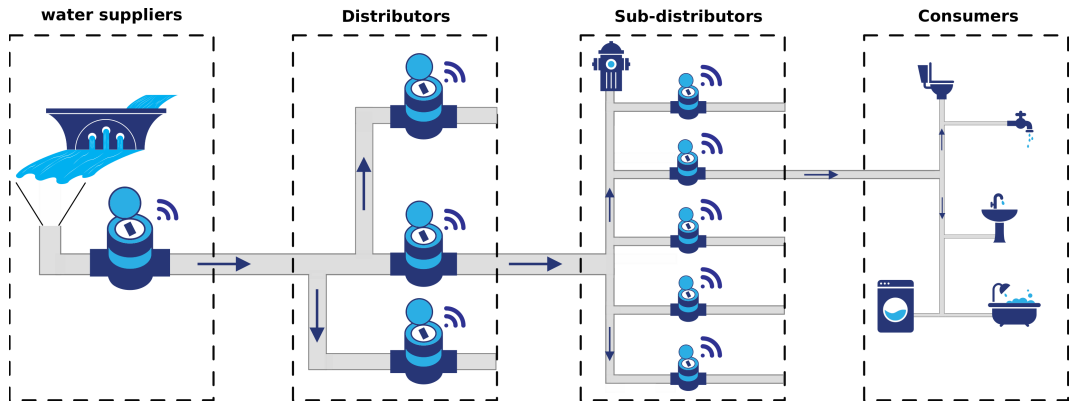


Figure 1.11 – Chaîne de distribution d'eau, du fournisseur au consommateur final

Fournisseurs
• Vendre une eau de bonne qualité • Répondre à la demande croissante en eau • Veiller et préserver la qualité de l'eau potable • Prévenir des éventuelles pollutions
Distributeurs
• Assurer la distribution complète de l'eau avec le minimum de pertes • Réduire et limiter les fuites dans le réseau de distribution • Détecter la détérioration de la canalisation et des systèmes vieillissants • Gérer et maintenir le réseau de distribution • Réduire les coûts associés à l'utilisation de systèmes complémentaires • Réduire les pertes d'eau non facturées (NRW)
Sous-distributeurs
• Gérer l'installation des compteurs d'eau et des abonnements • Protéger les données et la vie privée des consommateurs • Détecter et réduire les anomalies dues aux erreurs de comptage • Réduire les pertes d'eau non facturées (NRW)
Consommateurs
• Réduire les factures et la consommation de l'eau • Surveiller et comprendre l'utilisation quotidienne des ménages • Détecter les principales parts de consommation • Adapter ou modifier sa manière de consommer en fonction des ménages

Tableau 1.2 – Objectifs et spécifications des fournisseurs, distributeurs, sous-distributeurs et consommateurs d'eau

L'objectif des fournisseurs d'eau consiste à vendre de l'eau potable tout en garantissant une certaine qualité. Le rôle des distributeurs et des sous-distributeurs est d'assurer la distribution de l'eau entre les fournisseurs et les consommateurs tout en réduisant les pertes d'eau non facturées, en anglais *Non Revenue Water* (NRW). Enfin, certains consommateurs (les propriétaires, les industries, les services publics, etc.) souhaitent comprendre leurs consommations afin de mieux gérer les ressources.

Les compteurs d'eau sont classés en plusieurs catégories :

- les compteurs gros calibres, supérieurs au DN50, destinés aux distributeurs ;
- les compteurs divisionnaires et les compteurs de première prise, entre le DN20 et le DN50, destinés aux sous-distributeurs ;
- les compteurs divisionnaires, inférieurs au DN20, installés chez l'habitat.

L'ensemble de ces compteurs créé un réseau de capteurs qui permet de connaître précisément la consommation de l'eau et les anomalies liées au réseau de distribution. Les informations collectées ne suffisent pas à répondre aux spécifications définies dans le Tableau 1.2, c'est pourquoi il est nécessaire de définir une nouvelle stratégie de collecte de données afin de répondre aux objectifs de chaque partie prenante.

Une collecte de données avec une résolution plus élevée permet de surveiller plus efficacement le comportement de l'eau dans le réseau de distribution. En revanche, cela nécessite de transmettre davantage de données, ce qui augmente le temps de transmission et la consommation d'énergie du module radio. Nous allons montrer dans cette thèse comment augmenter la résolution des données mesurées et réduire la consommation d'énergie.

1.2.2 Question énergétique

La question énergétique est une question majeure pour pallier la nécessité de transmettre plus de données. C'est pourquoi, nous nous sommes intéressé, dans un premier temps, à la récupération d'énergie au sein des compteurs d'eau.

État de l'art : la récolte d'énergie

La récolte d'énergie est critique pour les systèmes embarqués de faible puissance car elle permet de les rendre plus autonome [25]. L'énergie solaire est une des sources d'énergie les plus connue, avec une densité de puissance pouvant aller jusqu'à 80mW/cm^2 lors d'une journée ensoleillée [26]. En règle générale, les compteurs d'eau ne sont pas exposés aux énergies lumineuses. C'est pourquoi, cette source d'énergie n'est pas considérée dans cette étude. Les principales technologies de récolte d'énergie envisageables sont les énergies thermique, RF et mécanique [27], [28]. La Fig. 1.12 illustre les moyens de récolte d'énergie au sein d'un compteur d'eau communicant.

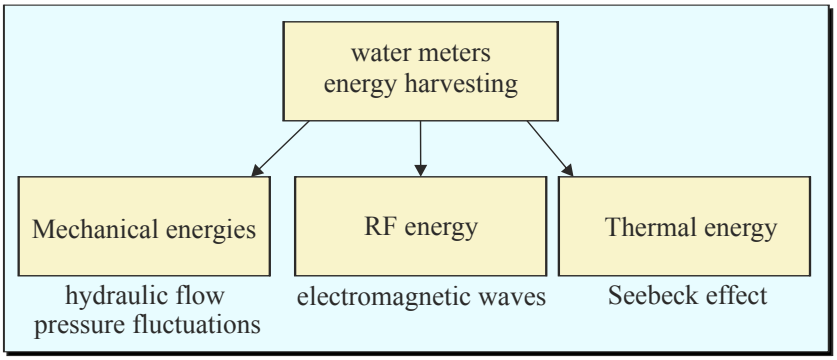


Figure 1.12 – Principes physiques de récolte d'énergie au sein des compteurs d'eau

L'énergie thermique

L'énergie thermique peut être convertie en utilisant l'effet thermoélectrique découvert par Thomas Johann Seebeck en 1821. La thermoélectricité est un phénomène physique qui fait apparaître une tension électrique à la jonction de deux métaux différents [29]. En effet, un générateur thermoélectrique récupère le flux de chaleur via une thermopile afin de le convertir en énergie électrique. Il a été montré qu'un courant électrique peut être généré à partir d'une différence de température δT. Plus la différence de température est grande, plus les performances du générateur sont élevées.

Dans le cas d'un compteur d'eau, le générateur peut être installé sur la canalisation de manière à obtenir une différence de température δT entre l'air ambiant et l'eau qui circule dans le compteur. La Fig. 1.13 illustre un compteur d'eau équipé d'un générateur thermoélectrique. La différence de température n'est pas toujours significative et son amplitude dépend de plusieurs facteurs (e.g., environnement, saison). La température de l'air peut fortement différer d'une saison à une autre avec parfois une différence de plusieurs degrés. La saison peut également avoir un impact sur la température de l'eau à l'intérieur de la canalisation. Comme décrit dans [30], une différence de température moyenne air-eau d'environ 3°C a été mesurée pour un an d'enregistrement. Cette différence de température a permis d'obtenir une puissance de sortie d'environ 2,8 mW. Cette valeur a été obtenue avec une canalisation déportée afin d'utiliser l'air extérieur au lieu de l'air ambiant du compteur. Cette configuration n'est pas souhaitable car elle nécessite l'utilisation de câbles entre le générateur déporté et l'électronique embarquée sur le compteur. Bien que cette configuration puisse réduire l'amplitude de δT, le générateur doit être inclus dans le compteur comme illustré sur la Fig. 1.13.

La récupération de l'énergie thermique dépend des fluctuations de température, mais également des caractéristiques du générateur thermoélectrique. Un générateur thermoélectrique reste encore trop onéreux pour être intégré dans un compteur commercialisé. Par conséquent, l'énergie thermique n'est pas une solution envisageable à ce jour. Cette forme d'énergie n'est néanmoins pas négligeable et pourra toujours encore être considérée comme appoint dans les années à venir.

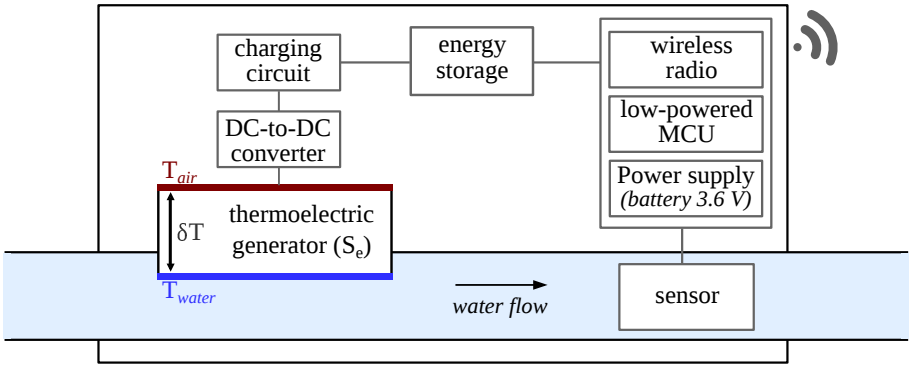


Figure 1.13 – Principe d'un compteur d'eau communicant qui intègre un générateur thermoélectrique pour récolter l'énergie thermique

L'énergie RF

L'énergie RF est présente dans les ondes électromagnétiques générées par les émetteurs radio ambiants. Ces ondes électromagnétiques peuvent être captées et converties en une tension continue afin d'alimenter des systèmes de très faibles puissances composés de capteurs, de microcontrôleurs et de composants électroniques. Le transducteur est composé d'une antenne, d'un convertisseur RF-DC et d'un système de stockage [31]. La Fig. 1.14 illustre l'intégration d'un transducteur RF dans un compteur d'eau communicant. Ce transducteur est appelé "rectenna" et est utilisé pour convertir l'énergie RF en énergie électrique [32].

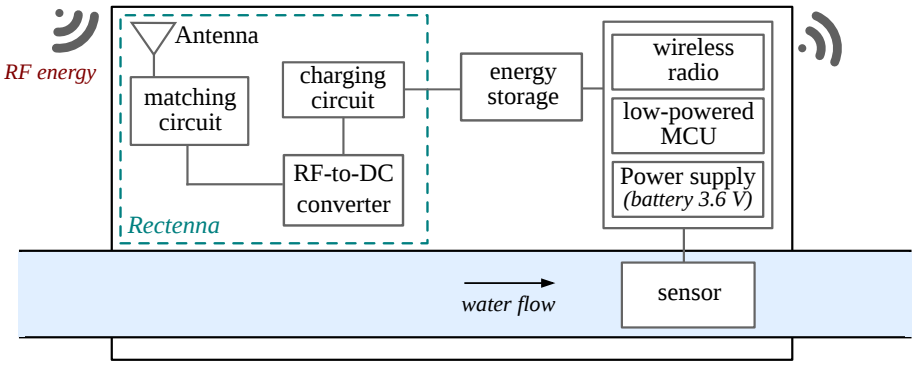


Figure 1.14 – Principe d'un compteur d'eau communicant qui intègre une *rectenna* pour récupérer l'énergie RF ambiant

La performance de cette solution de récupération d'énergie dépend de plusieurs critères :

- la distance entre la source RF et la rectenna ;
- le gain de l'antenne ;
- l'efficacité de conversion ;
- la puissance rayonnée.

Le gain de l'antenne dépend de sa taille et de ses caractéristiques [33]. En prenant l'exemple d'une antenne dipôle demi-onde, sa longueur L peut être calculée par (1.4) avec $\lambda = \frac{c}{f}$. La longueur de l'antenne est exprimée en mètre avec c la vitesse de la lumière ($c = 3 \cdot 10^8$ m/s), f la fréquence du signal (Hz) et λ la longueur d'onde (m) :

$$L = \frac{\lambda}{2} = \frac{c}{2 \cdot f}. \tag{1.4}$$

Dans ce cas, λ est limité au diamètre du compteur. La taille de l'antenne ne doit donc pas excéder 10 cm. La fréquence f doit donc être supérieure ou égale à 1.5 GHz. Par conséquent, la récupération d'énergie concerne la bande de fréquences supérieure aux Ultra Haute Fréquences (UHF) et concerne en particulier les réseaux Wifi ou cellulaires avec des fréquences aux alentours de 2,4 GHz. Plusieurs études sur la récupération d'énergie RF ont été réalisées dans des zones couvertes par un certain nombre d'ondes électromagnétiques [34], [35], [36], [37], [38]. L'un des prototypes réalisé est capable de récolter quelques microwatts pour une tension de sortie de 0,45 V. Ce prototype permet de charger un super-condensateur en 7 heures.

On peut constater que la puissance récoltée est insuffisante pour auto-alimenter l'électronique embarquée du compteur. De plus, les compteurs d'eau peuvent être à proximité de différents obstacles, ce qui augmente considérablement les pertes de puissance des ondes électromagnétiques.

Les énergies mécaniques

Dans le comptage de l'eau, le flux d'eau peut être considéré comme la principale source d'énergie. Les énergies mécaniques sont généralement présentes sous la forme de déplacements et de vibrations. Ces sources d'énergie peuvent être converties en énergie électrique à l'aide d'un transducteur électromagnétique ou piézoélectrique [39].

La solution la plus directe pour récupérer l'énergie cinétique est d'utiliser un transducteur électromagnétique en combinaison avec la turbine. Ce principe est uniquement valable pour les compteurs mécaniques. L'énergie électrique récoltée est proportionnelle à la vitesse de rotation de la turbine et délivre une tension alternative. Le transducteur électromagnétique est illustré sur la Fig. 1.15. Il permet de convertir l'énergie cinétique en une tension électrique alternative. Cette tension nécessite d'être redressée en une tension continue avant d'être stockée. La puissance de sortie du générateur électromagnétique dépend donc du débit et plus particulièrement de la consommation de l'eau [40], [41]. Plus le débit est élevé, plus l'énergie cinétique est importante. Par ailleurs, les caractéristiques du générateur sont très importantes et ne doivent pas modifier la métrologie du compteur définie dans la section 1.1.2. En faisant référence au prototype développé dans [41], la perte de charge maximale réglementaire (0.63 bars à Q_3) est déjà atteinte

pour un débit de 600 l/h. Ce débit est nettement inférieur au débit nominale d'un compteur d'eau DN15. Le générateur électromagnétique commence à récolter de l'énergie à partir d'un débit minimal, ce qui limite la plage de production d'énergie. L'énergie cinétique est nulle lorsqu'il n'y a pas de consommation d'eau et elle est difficilement exploitable pour les compteurs sans parties mobiles (compteurs statiques). Une solution de récupération supplémentaire est donc requise pour imaginer un compteur totalement autonome et auto-alimenté.

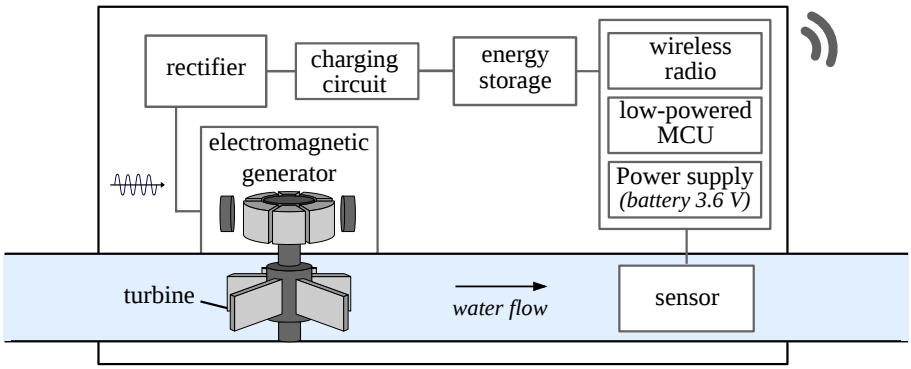


Figure 1.15 – Principe d'un compteur d'eau communicant qui intègre un générateur électromagnétique pour récupérer l'énergie cinétique provenant du flux d'eau

L'énergie hydraulique est également présente dans le compteur d'eau sous forme de variations aléatoires de pression. Les variations de pression peuvent être converties de plusieurs manières en énergie électrique [30]. Les transducteurs électromagnétiques, connus pour leur densité de puissance élevée, sont généralement utilisés pour convertir des vibrations à grandes échelles [42] [43]. Les transducteurs piézoélectriques permettent d'extraire l'énergie pour des mouvements mécaniques plus petits, en contrepartie d'une densité de puissance plus faible. De nombreuses études ont montré l'efficacité de ces transducteurs [44], [45]. Un prototype basé sur un transducteur piézoélectrique a même été proposé afin de récolter les variations de pression causées par une pompe [46]. La Fig. 1.16 illustre le principe de fonctionnement d'un élément piézoélectrique au sein d'un compteur d'eau.

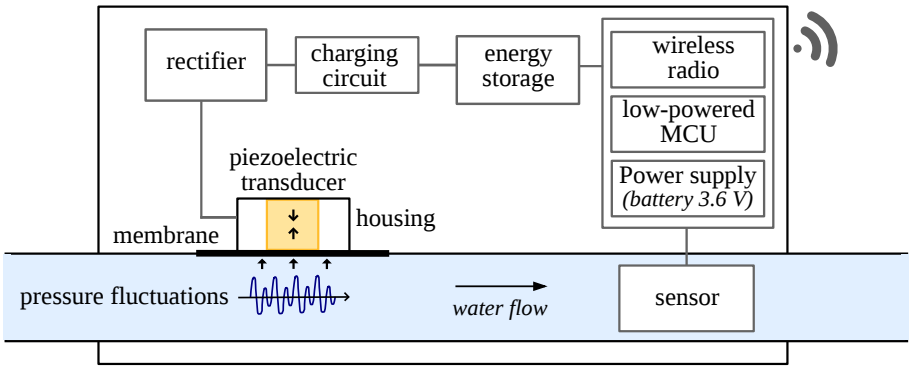


Figure 1.16 – Principe d'un compteur d'eau communicant qui intègre un transducteur piézoélectrique pour récupérer l'énergie hydraulique provenant des variations de pression

L'énergie mécanique est considérée comme la source d'énergie principale au sein des compteurs d'eau. C'est cette forme d'énergie qui est retenue dans l'étude expérimentale qui suit.

Étude expérimentale

Une étude expérimentale a été effectuée pour évaluer la quantité d'énergie électrique qu'il est possible de récupérer à partir des deux énergies mécaniques. En effet, l'énergie cinétique provenant du flux d'eau et l'énergie hydraulique qui découle des variations de pression dans la canalisation sont indépendantes mais peuvent être complémentaires, notamment pour les compteurs mécaniques.

Un premier prototype est composé d'un générateur électromagnétique qui concerne uniquement les compteurs d'eau mécaniques. A titre d'exemple, ce prototype est couplé à un compteur de vitesse à jet unique DN15 utilisé dans des applications domestiques. Il est illustré sur la Fig. 1.17. Afin de faciliter l'expérimentation, nous avons remplacé le transducteur mécanique d'origine par le générateur électromagnétique. Un aimant permanent immergé dans l'eau et relié à l'axe rotatif de la turbine entraîne un autre aimant qui permet de déporter le mouvement en dehors du compteur. Celui-ci a pour but d'assurer la rotation d'un second axe sur lequel est fixé un troisième aimant. Ce dernier émet un flux de champ magnétique en direction de la bobine. Le champ magnétique est constant mais lorsque l'aimant se déplace par rapport à la bobine, le flux de champ magnétique qu'elle subit varie au cours du temps. La bobine devient donc un générateur de tension alternative.

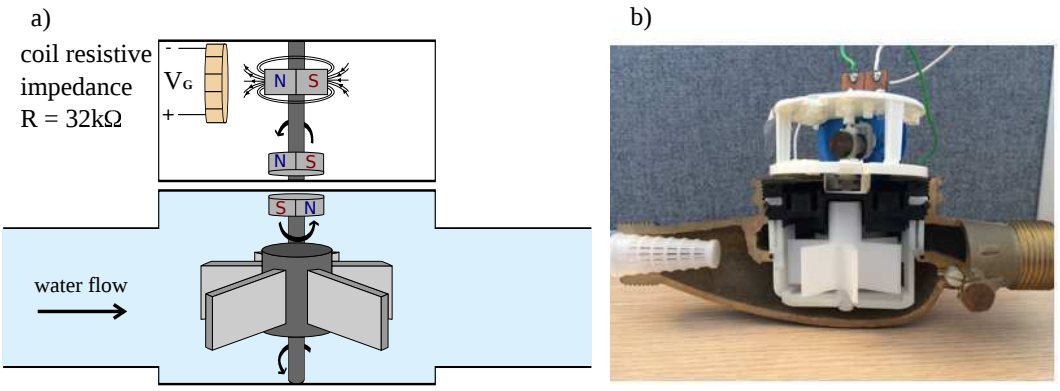


Figure 1.17 – Prototype expérimental qui combine un compteur de vitesse DN15 avec un générateur électromagnétique, a) principe de fonctionnement, b) vue du prototype élaboré

La tension alternative produite est redressée en une tension continue destinée à alimenter un circuit électrique. Le redresseur mis en place est réalisé à partir d'un multiplicateur de tension basé sur le *montage de Greinacher* comme illustré sur la Fig. 1.18. Ce redresseur a été choisi pour obtenir une tension continue égale au double de la tension alternative d'entrée. Les tests expérimentaux décrits dans le Tableau 1.3 montrent que l'énergie récoltée est directement proportionnelle au débit de l'eau. Le générateur fournit une puissance de sortie suffisante pour

que l'électronique embarquée fonctionne en mode veille à partir de 40 l/h. Au-delà de 300 l/h, la puissance de sortie excède la puissance moyenne requise pour assurer la transmission des données toutes les 8 secondes (Walk-By/Drive-By). Par ailleurs, le Tableau 1.4 montre que les caractéristiques du générateur n'affectent pas la métrologie du compteur et que les pertes de charge sont négligeables. Les tests ont été effectués sur un banc d'essai et la puissance de sortie générée en fonction du débit de l'eau est illustré sur la Fig. 1.19. A titre d'exemple, cette figure montre également l'énergie qu'il est possible de récupérer lors de l'utilisation d'un WC, d'une douche ou d'un robinet. En dessous de 40 l/h, l'énergie récoltée n'est pas suffisante pour alimenter le compteur communicant.

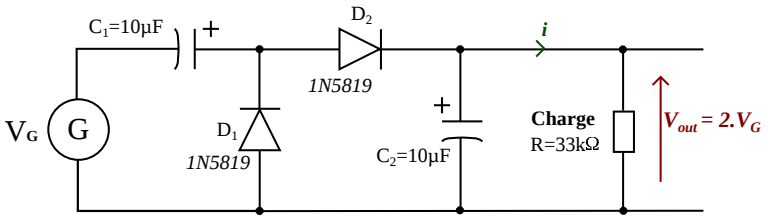


Figure 1.18 – Schéma électrique pour redresser et filtrer la tension alternative récoltée par le générateur électromagnétique proposé

Tableau 1.3 – Énergie récoltée à partir du générateur électromagnétique illustré sur la Fig. 1.17

Débit (l/h)	Tension efficace (V_{out})	Courant (μA)	Puissance (μW)
40	0.192	5.82	1.12
80	0.359	10.88	3.91
150	0.712	21.58	15.36
300	1.43	43.33	61.97
360	1.65	50.00	82.50
700	2.96	89.70	265.50
1000	4.21	127.58	537.09
1350	5.6	169.70	950.30
2400	10.15	307.58	3121.89
3125	12.65	383.33	4849.17

Tableau 1.4 – Comparaison des pertes de charge générées sans et avec le générateur électromagnétique

Débit (l/h)	Pertes de charge générées (bar)	
	Sans générateur	Avec générateur
Q_3 = 2500	0.610	0.625
Q_4 = 3125	0.972	0.991

Un second prototype a été mis au point pour récolter l'énergie provenant des variations de pression dans la canalisation. Il est composé d'un transducteur piézoélectrique situé dans un corps usiné. En annexe A.2, la Fig. A.1 montre le transducteur placé dans une chambre étanchéifiée par une membrane Ethylène-propylène-Diène Monomère (EPDM). Les variations

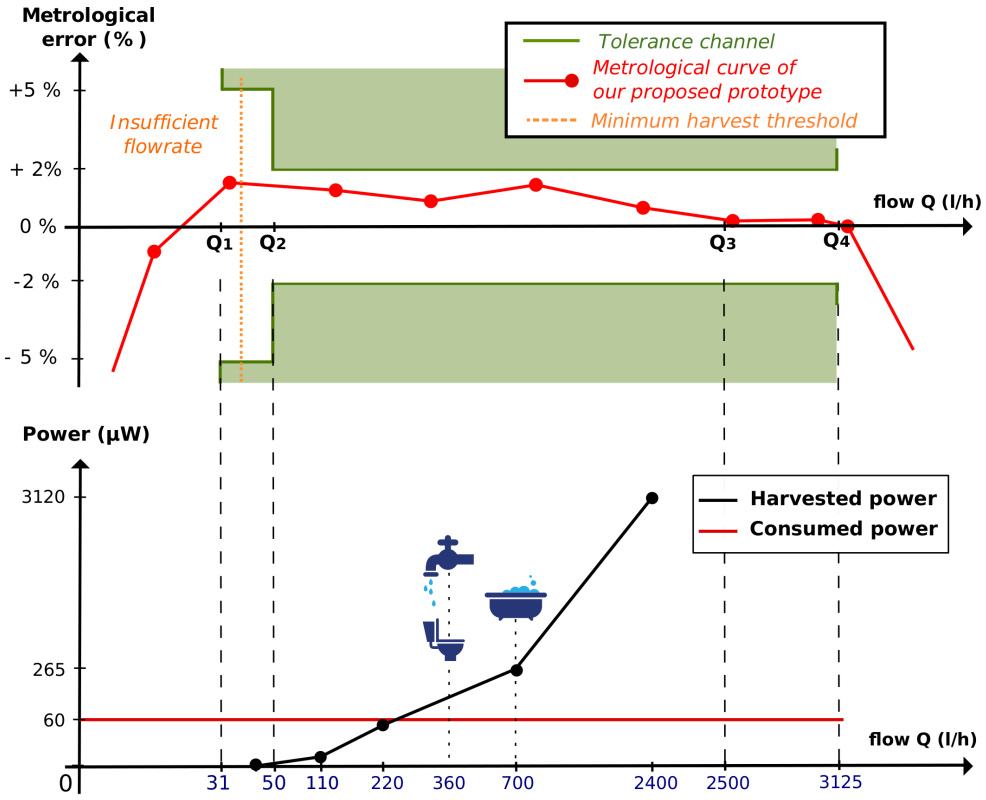


Figure 1.19 – Performances de récupération d'énergie et caractéristiques métrologiques obtenues avec le prototype développé

de pression présentent dans la canalisation sont directement exercées sur cette membrane et sont translatées sur la surface du transducteur piézoélectrique. Ces variations de pression ont été mesurées dans trois réseaux de distribution différents : en entreprise, en ville et en campagne. Les oscillations sont composées de variations aléatoires qui fluctuent en fonction de l'utilisation de l'eau dans le réseau de distribution. Les résultats sont présentés sur la Fig. 1.20. Bien que de fortes variations peuvent parfois excéder 1 bar, les variations de pression les plus fréquentes sont généralement comprises entre 5 et 100 mbar. Nous avons utilisés deux transducteurs piézoélectriques différents, développés par les fabricants *Multicomp* et *Argillon*. Dans notre étude, nous les avons utilisés dans le but de convertir ces variations de pression en énergie électrique. Leurs caractéristiques sont illustrées sur le Tableau 1.5.

Le transducteur d'Argillon fournit les meilleures performances mais ne peut générer que quelques nanowatts de puissance moyenne. L'énergie récoltée ne permet pas de maintenir le système embarqué en mode veille. Un empilement d'élément piézoélectrique pourrait amplifier la puissance de sortie, mais cela augmenterait considérablement le prix du générateur. Le transducteur piézoélectrique pourrait être couplé au générateur électromagnétique dans les compteurs mécaniques pour assurer le mode veille pour un débit inférieur à 40 l/h.

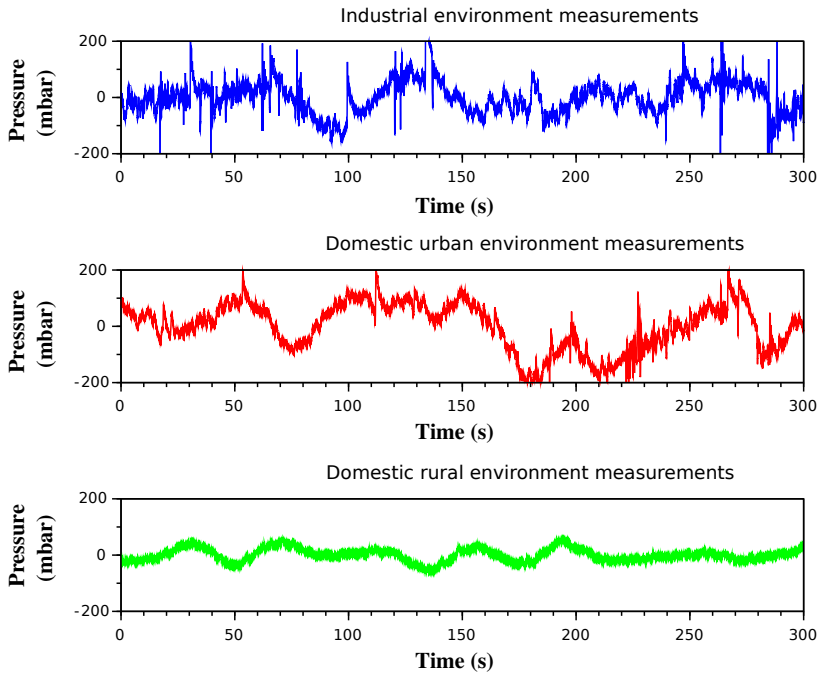


Figure 1.20 – Mesures des variations de pression dans un réseau de distribution industriel, urbain et rural

Tableau 1.5 – Caractéristiques techniques des deux transducteurs piézoélectriques testés au sein d'un prototype expérimental de récupération d'énergie

Transducteur	PZT Multicomp	PZT Argillon
Type	Capteur diaphragme céramique	Actionneur de flexion trimorphe
Longueur (mm)	35	50
Hauteur (mm)	0.5	7.2
Épaisseur (mm)	0.25	0.78
Capacité	37 pF	43 nF

1.2.3 Discussion

Un état de l'art sur la récupération de l'énergie dans l'environnement du compteur d'eau a été effectué. L'énergie RF et l'énergie thermique n'ont pas été considérées davantage dans cette étude. Il a été montré que l'énergie provenant des variations de pression ne suffit pas à alimenter l'électronique embarquée du compteur d'eau dans son fonctionnement de veille. La plus grande part de l'énergie récoltée est donc obtenue à travers l'énergie cinétique. Cependant, le débit n'est pas toujours suffisant en valeur et en durée et l'énergie récoltée dépend principalement de la consommation de l'eau. Les valeurs numériques présentés dans le Tableau 1.3 sont présentées en annexe A.3 pour trois consommations différentes. On peut constater que la puissance de sortie moyenne générée varie de 3 à 61 μW. Ces chiffres sont à relativiser avec la consommation moyenne d'un module radio actuel Walk-By Drive-By de 58 μW (Tableau 1.1).

Actuellement les compteurs d'eau communicants et commercialisés sont alimentés par une pile lithium qui permet d'assurer le traitement et la transmission des données de manière autonome. Ce dispositif de stockage n'est pas adapté pour être rechargé ou remplacé. La pile est solidaire du circuit électrique; le tout étant noyé dans de la résine pour les protéger vis à vis de l'humidité. Les piles rechargeables ont la capacité d'augmenter la longévité du système et peuvent assurer de nombreux cycles de charges et de décharges. Cependant, la recharge de la pile est un réel défi lorsque la température est négative. C'est pourquoi, nous proposons de stocker l'énergie récoltée dans un super-condensateur. Ce dernier est capable d'améliorer le temps de charge et offre une capacité 10 000 fois supérieure à celle d'un condensateur électrolytique standard [47]. La pile serait alors la principale source d'énergie et le générateur permettrait de rallonger sa durée de vie.

L'optimisation énergétique est uniquement valable lorsque les énergies mécaniques sont efficacement converties. Dans le cas contraire, la récolte d'énergie ne répond pas à la question énergétique. Nous avons donc choisi de ne pas poursuivre dans cette voie. Nos travaux de recherche ont été redirigés vers une autre stratégie, permettant d'économiser l'énergie lors de la transmission des données.

1.3 Proposition d'une nouvelle stratégie de collecte de données

Nous proposons une nouvelle stratégie de collecte de données qui consiste à transmettre toutes les données brutes mesurées par le compteur tout en réduisant les efforts de calculs et de transmission qui sont très énergivores. Par conséquent, cette stratégie permettra alors d'offrir davantage de fonctionnalités aux distributeurs d'eau. Le mesureur et les caractéristiques métrologiques ne changent pas et ne font pas l'objet de cette étude.

1.3.1 Définition des données brutes

La stratégie proposée de collecte de données a pour but de réduire la consommation du module radio et de déporter le traitement des données au niveau du serveur. Le fait de déporter le maximum de fonctionnalités embarquées permet également de réduire le nombre de tâches traitées par le microcontrôleur. L'idée principale de cette nouvelle stratégie est de transmettre toutes les données brutes mesurées. Les données brutes sont définies comme étant les horodatages de chaque événement $\{t_1, t_2, ..., t_i\}$. Ces événements sont générés dès qu'une variation Δ dans la grandeur mesurée est détectée. Dans le cadre d'un compteur d'eau, la variation Δ est fixe et correspond à l'écoulement d'un litre pour un compteur d'eau DN15 (10 litres pour un compteur d'eau DN100). Sa valeur dépend de la taille du compteur et des caractéristiques du capteur. Le signe de Δ fournit l'information sur le sens de l'écoulement. De fait, plus le débit est élevé, plus les événements sont rapidement générés. La succession des événements est illustrée sur la Fig. 1.21 avec les horodatages $\{t_1, t_2, t_3, t_4, t_5, t_6, t_7$ et $t_8\}$.

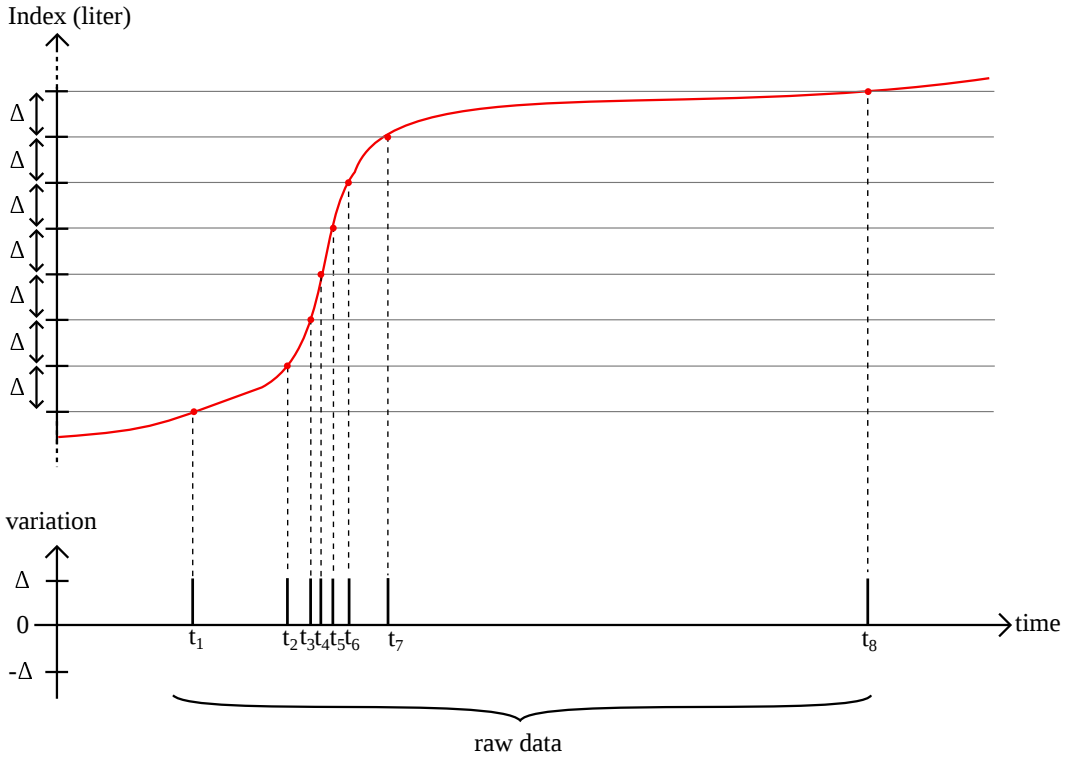


Figure 1.21 – Illustration des données brutes générées dès qu'une variation est détectée par le capteur

Les instants horodatés de chaque événement reflètent la consommation d'eau et définissent la courbe de charge. A ce stade, il s'agit de la résolution maximale qu'il est possible d'avoir dans la mesure de la consommation d'eau. La Fig. 1.22 illustre les données brutes mesurées et montre la courbe de charge déduite et celle issue à partir seulement des index relevés toutes les heures.

Dans le cadre des compteurs mécaniques, une démultiplication est effectuée dans le totalisateur entre la rotation de la partie mobile (la turbine par exemple) et la rotation du demi-disque métallisé. Enfin, un événement est généré par l'électronique embarquée dès qu'un tour a été effectué par le demi-disque présent sur le totalisateur. Le demi-disque métallisé est illustré sur la Fig. 1.2. La rotation est détectée par un capteur inductif. La sortie du capteur est échantillonnée avec une fréquence d'échantillonnage de 64 Hz qui permet de garantir une acquisition des données avec une résolution de 15.625 ms. Pour des raisons de simplifications, nous n'allons pas manipuler des chiffres à virgules. Nous avons défini une résolution à la milliseconde.

1.3.2 Traitements déportés des consommations

Les données brutes permettent d'effectuer des analyses descriptives précises, telles que des consommations anormales, des fraudes ou encore des fuites. Ces analyses peuvent être effectuées par la courbes de charge, mais également avec le profil de consommation. La courbe de charge a

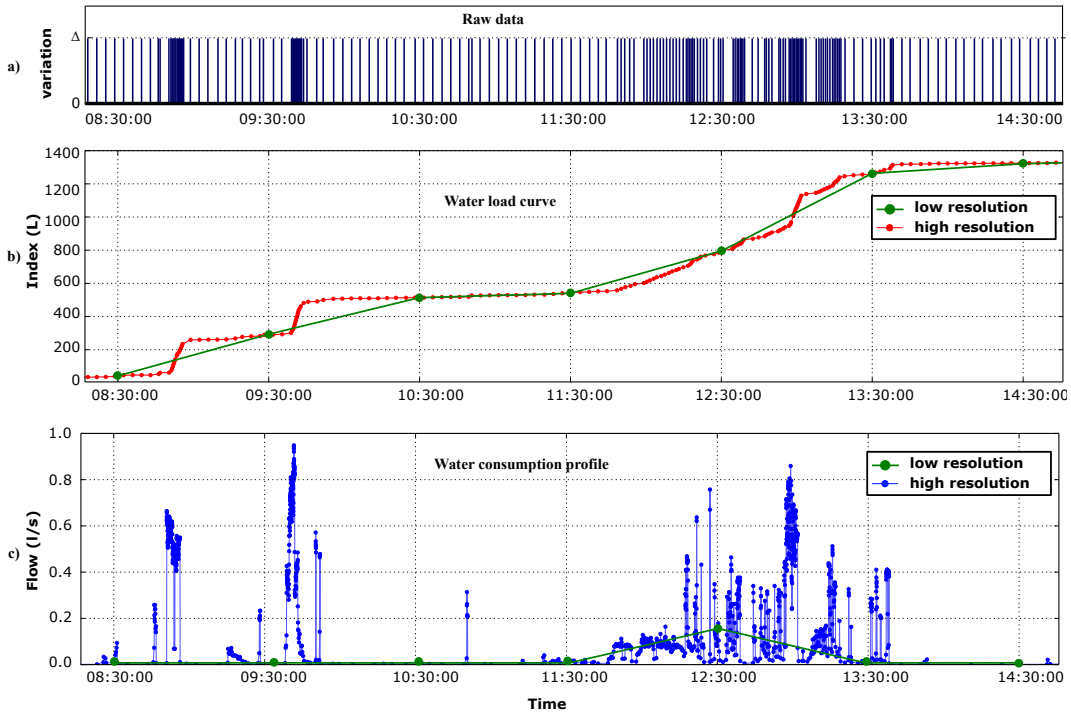


Figure 1.22 – Représentation des données brutes, des courbes de charge et des profils de consommation associés, a) données brutes mesurées, b) comparaison entre la courbe de charge à partir des index relevés toutes les heures et celle à partir des données brutes (événements horodatés), c) profils de consommation calculés à partir des données brutes

été définie par (1.3). Le profil de consommation peut être assimilé comme la dérivé de la courbe de charge et est considéré comme le débit d'eau moyen entre deux instants consécutifs :

$$P(t_i) = \frac{I(t_{i+1}) - I(t_{i-1})}{t_{i+1} - t_{i-1}} \tag{1.5}$$

Le profil de consommation $P(t_i)$ à l'instant t_i est calculé par (1.5) avec $I(t_{i-1})$ et $I(t_{i+1})$ les index aux instants t_{i-1} et t_{i+1}. Notons que $P(t_0)$ est toujours initialisé à 0 l/s.

La Fig. 1.23 montre que toutes les fonctionnalités sont déportées au niveau du serveur. Cette stratégie nécessite un réseau fixe car les données brutes doivent être collectées en permanence. Cela offre beaucoup plus de possibilités et davantage de flexibilité dans l'évolution des exigences. Cette stratégie offre une réelle pérennité pour cette nouvelle génération de compteur. L'analyse des données sur le serveur pourra être mise en place en fonction des besoins, sans pour autant modifier la transmission des données. La précision du profil de consommation issue des données brutes est comparée à celle où les données ont été mesurées toutes les heures sur la Fig. 1.22.

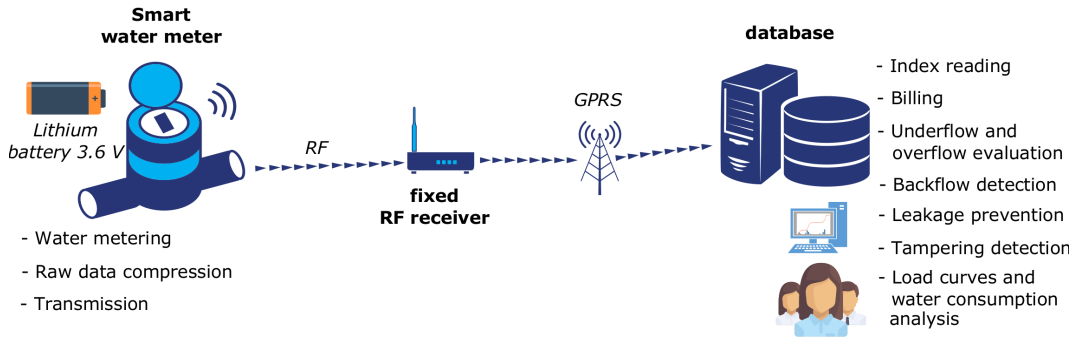


Figure 1.23 – Nouvelle stratégie de collecte de données afin de déporter les traitements et les analyses au niveau du serveur

1.3.3 Compression de données

Bien que la taille des messages RF soit limitée par le protocole radio et par les caractéristiques du matériel employé, le volume de données joue un rôle essentiel dans la taille et dans le nombre des messages à transmettre. Typiquement, la norme w-MBus présentée dans la section 1.1.3 impose une longueur de données maximale de 255 octets [23]. Par ailleurs, la taille du message définit également la durée de transmission. Un long message conduit à un temps de transmission plus élevé, donc à une consommation d'énergie plus importante.

Due aux contraintes énergétiques, il n'est pas concevable de transmettre directement l'horodatage de chaque événement. Cela engendrerait des coûts de transmission trop importants. Il est donc nécessaire de compresser les données pour réduire la taille des messages et de les stocker en mémoire RAM pendant une période de temps T pour limiter le nombre de transmissions. Cette période est variable et correspond à la différence de temps entre deux émissions. La stratégie de transmission et la compression des données seront détaillées dans le prochain chapitre. Afin d'optimiser les performances de compression, nous proposons de transformer les données en utilisant les différences de temps entre chaque événement horodaté. En utilisant des différences de temps, un instant de référence est nécessaire pour pouvoir par la suite reconstruire fidèlement les événements au cours du temps dans le serveur.

Dans un premier temps, nous proposons d'insérer les données brutes dans la trame. Nous avons décidé de modifier le champ *PAYLOAD* de la Fig. 1.9 de la manière suivante :

- l'index qui est obligatoire pour la facturation;
- l'instant de référence utile pour la reconstruction des données brutes;
- les données brutes encodées relatives à l'horodatage des variations détectées.

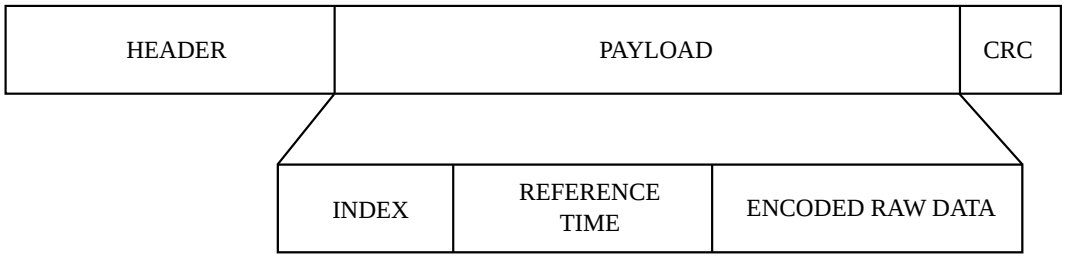


Figure 1.24 – Structure de trame respectant le standard européen EN 13757-4 utilisé dans les tests expérimentaux

Cette nouvelle structure de trame est illustrée sur la Fig. 1.24. Sa longueur est directement liée à la quantité de données brutes compressées. La compression et l'encodage des données sont les principales contributions de cette thèse. Basé sur le module de communication RF présenté dans la section 1.1.3, un prototype sera développé dans le chapitre suivant pour collecter, compresser et transmettre les données de comptage. L'architecture matérielle du module radio ne sera pas modifiée.

1.4 Conclusion

Un état de l'art sur le comptage de l'eau a été présenté dans ce chapitre. Le fonctionnement général des compteurs d'eau actuels a été décrit, allant du mesureur jusqu'à la transmission des données. Leurs performances et leurs limites ont également été présentées. Ces compteurs communicants sont des systèmes embarqués de faible puissance où l'électronique embarquée est conçue pour assurer la transmission des données à distance et de manière autonome. Parfois soumis à des variations importantes de températures et à de forts taux d'humidité, cette électronique est alimentée par une pile. La durée de vie du système est de l'ordre d'une quinzaine d'années. Cette durée est définie en fonction du nombre de traitements effectués dans le microcontrôleur et du nombre de messages transmis.

Les besoins du marché poussent à transmettre plus d'information issue de l'analyse de la consommation et du compteur lui-même. Dès lors, les messages sont plus longs et le nombre de transmissions augmente tout comme la consommation énergétique. Les fonctionnalités dans le compteur sont multipliées, augmentant de fait le nombre de traitements dans le module radio. La question énergétique et les exigences évolutives des distributeurs d'eau sont les enjeux majeurs de ces systèmes. La récupération d'énergie est perçue comme une des solutions permettant de résoudre la question énergétique. Un état de l'art et une étude expérimentale sur la récolte d'énergie au sein des compteurs d'eau ont été présentés. Deux prototypes de récupération d'énergie ont été réalisés et testés. Les résultats montrent que la source d'énergie la mieux adaptée est l'énergie cinétique provenant du flux d'eau consommé. L'énergie récupérée n'est pas toujours suffisante et ne permet donc pas d'assurer une autonomie énergétique totale. Nous avons donc complété cette approche en proposant une nouvelle stratégie de collecte de

données afin d'économiser l'énergie dans la transmission des consommations. L'idée consiste à transmettre toutes les données brutes détectées par le capteur. Ces données fournissent une information plus précise sur la consommation. Cette nouvelle stratégie permet d'assurer une certaine pérennité des compteurs car le traitement des données et les fonctionnalités sont totalement déportés sur le serveur.

Transmettre les données brutes avec une meilleure résolution des données augmente le temps de transmission car les messages deviennent plus longs. Dans le chapitre 2, nous nous intéresserons aux algorithmes de compression de données afin de réduire la taille des messages et d'optimiser la consommation d'énergie du module radio.

2 Compression de données sans perte pour la transmission

La compression de données consiste à réduire la taille des données pour la transmission et le stockage. Ce deuxième chapitre présente les concepts de base et les techniques associés à la compression des données sans perte en général. L'objectif est de compresser efficacement les données liées à la consommation d'eau sans perte d'information et avec une résolution à la milliseconde pour répondre aux objectifs définis dans le chapitre précédent. Autrement dit, les données présentes dans le champs *PAYLOAD* de la Fig. 1.24 seront efficacement compressées pour réduire la longueur du message à transmettre.

La section 2.1 introduit la compression de données et présente les concepts de base. Les principales techniques de compression sans perte sont présentées dans la section 2.2. Nous détaillerons les avantages et les inconvénients de chacune d'entre elles. Les limites de chacune des techniques seront évaluées. Une nouvelle méthode de compression sans perte est proposée dans la section 2.3 afin d'obtenir des performances adaptées à nos besoins. Ces performances seront comparées aux autres algorithmes dans la section 2.4. Les contributions de cette étude ne se limitent pas uniquement à réduire la taille des messages à transmettre. Une stratégie est également proposée pour assurer la bonne réception des informations compressées. Enfin, un bilan énergétique est effectué dans la section 2.5, pour montrer la réduction de la consommation d'énergie liée à cette nouvelle collecte de données.

2.1 Compression de données : les définitions

La compression de données est un processus qui permet de réduire le nombre de bits nécessaire pour représenter des données. La compression est utile pour réduire la longueur des messages à transmettre mais également pour limiter la quantité d'information à stocker dans le serveur [48]. Avant d'introduire les contraintes de stockage de données au chapitre 3, nous nous intéressons ici à optimiser la transmission des informations dans le contexte du comptage de l'eau.

2.1.1 Évaluation des performances

La méthode de compression dépend intrinsèquement de la nature des données brutes à compresser. Un fichier texte n'est pas compressé de la même manière qu'une image ou qu'un fichier audio. Plusieurs critères doivent être pris en considération pour évaluer les performances de compression, notamment le taux de compression, le temps de traitement et la dégradation éventuelle des données causée par le processus de compression [49], [50]. Le critère principal d'évaluation des performances d'un algorithme de compression est le taux de compression τ comme exprimé dans (2.1). Exprimé en pourcentage, ce taux de compression est le rapport entre la taille des données brutes compressées C_d et la taille des données brutes originales (non compressées) O_d :

$$\tau = \left(1 - \frac{C_d}{O_d}\right) \times 100. \tag{2.1}$$

2.1.2 Types de compression

Les données à compresser peuvent être dégradées dans le but d'optimiser le taux de compression. C'est le cas des algorithmes de compression avec perte. Un algorithme de compression avec perte est considéré comme une opération irréversible et non conservatrice. Ce type de compression utilise des approximations pour représenter les données d'origine. Certaines informations peuvent être répétitives et donc amenées à être supprimées ou transformées. Généralement, cette étape de transformation implique une petite distorsion des données d'origine, ce qui favorise la répétition des données. Cette répétition, appelée *redondance d'information*, est très utilisée dans la compression de données pour optimiser le taux de compression. La compression de données avec perte a été initialement établie par Shannon avec la théorie du taux de distorsion [51], [52]. Les algorithmes de compression avec perte sont utiles pour les images, le son et la vidéo, à l'instar des formats JPEG et MP3 [53], [54].

La compression de données sans perte, par opposition à la compression de données avec perte, ne tolère aucune distorsion entre les données d'origine et les données reconstruites [55]. Ce type de compression est considéré comme non destructif. Les fichiers binaires, les codes sources ou encore les textes sont de parfais exemples dans lesquels la suppression de lettres, de chiffres ou de mots est dommageable. Comme défini dans la section 1.3, les données à compresser, dans le cas du comptage, sont des séries temporelles composées de plusieurs nombres entiers. Ils correspondent à des instants où l'index enregistré par le compteur évolue au cours du temps. La perte sur le nombre d'événement détectés ne peut pas être tolérée. Ce chapitre se concentre donc sur les algorithmes de compression sans perte.

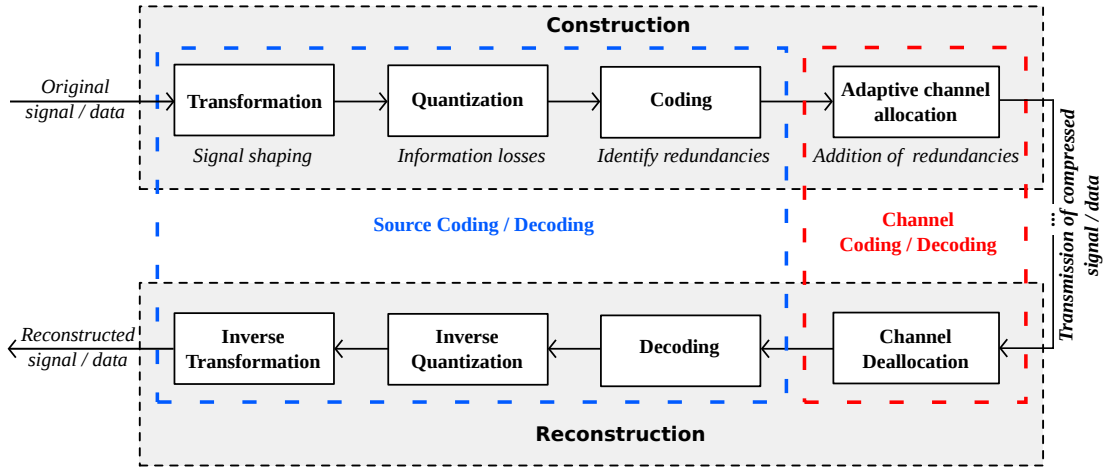


Figure 2.1 – Architecture générale pour la compression et la décompression de données

2.1.3 Codage de source et codage de canal

La compression de données s'effectue selon une architecture générale qui regroupe plusieurs étapes fondamentales successives. Ces étapes sont illustrées sur la Fig. 2.1. La première étape de compression consiste à transformer les données d'origine pour augmenter le nombre de redondances. Cette étape regroupe les symboles identiques pour créer des séquences redondantes. La deuxième étape est la quantification qui induit de la perte d'information pour favoriser davantage les redondances. C'est ici que le type de compression (avec ou sans perte) est déterminé. Enfin, la compression des données est principalement effectuée lors de l'étape de codage où des codeurs entropiques, par dictionnaire ou encore hybrides sont utilisés pour compresser les données redondantes. Ces algorithmes seront davantage détaillés dans la section 2.2. L'étape de codage clôt le processus de compression, appelé également *codage de source*. Le *codage de canal* est conçu pour assurer la transmission des informations. Cette étape rajoute des redondances dans les messages en cas de perte de trame, d'encombrement du réseau ou d'erreur dans la transmission des données. Le processus de décodage et de décompression est réalisé par le destinataire au niveau du serveur. Cette étape sera développée dans le chapitre 3.

2.1.4 Théorie de l'information

La théorie de l'information provient de Shannon et est basée sur la distribution de probabilité pour mesurer l'efficacité des codeurs entropiques [56]. La quantité d'information, plus connue sous la dénomination *entropie à la source*, peut être calculée pour représenter l'incertitude dans l'information. L'entropie est une fonction mathématique qui permet de quantifier l'information présente dans un ensemble de données [51], [57].

Soit $S = \{s_1, s_2, s_3, ..., s_k\}$ l'ensemble des symboles possibles survenus avec les probabilités

d'apparition respectives p_1, p_2, p_3, ..., p_k. La quantité d'information $I(s_i)$ est définie en bits :

$$I(s_i) = \log_2\left(\frac{1}{p_i}\right). \tag{2.2}$$

L'entropie H est donc calculée pour représenter le nombre moyen de bits par symbole :

$$H = \sum_{i=1}^{k} p_i I(s_i) = \sum_{i=1}^{k} p_i \log_2\left(\frac{1}{p_i}\right). \tag{2.3}$$

Nous considérons que les données brutes s'expriment en symboles. Tous les symboles (lettre, chiffre, etc.) sont regroupés et encodés dans des mots de code exprimés en binaire. Les mots de code sont donc des regroupements de symboles encodés. Un mot de code est représenté par une longueur moyenne L. Cette longueur est liée à la probabilité d'occurrence p_i du symbole i et permet de définir l'efficacité du code, où l_i exprime la longueur du mot de code. L'efficacité du code est définie par le rapport R, exprimé en pourcentage, entre l'entropie de la source H et la longueur moyenne du code L :

$$L = \sum_{i=1}^{k} p_i l_i, \tag{2.4}$$

$$R = \frac{H}{L} \cdot 100. \tag{2.5}$$

Le rapport R permet de déterminer si un code est optimal ou non. Un code optimal est un code ayant la plus petite longueur moyenne possible. La compression est d'autant plus forte que la longueur moyenne des mots de code est faible. Trouver un code optimal revient donc à choisir les longueurs des mots de code en fonction de la distribution de probabilité de chaque symbole.

2.1.5 Description des codes

Code à longueur variable et à longueur fixe

Conformément au standard ASCII (American Standard Code for Information Interchange), chaque symbole est représenté par une longueur fixe de 8 bits. Les codes ASCII sont des codes à longueur fixe (FLC) et sont très répandus dans les systèmes informatiques. Les FLC sont souvent

Tableau 2.1 – Comparaison des performances de compression entre les codes à longueurs variables (VLC) et les codes à longueurs fixes (FLC)

Symboles	Probabilités	VLC	FLC	$I(s_i)$
A	1/4	10	00	2
B	1/8	110	01	3
C	1/8	111	10	3
D	1/2	0	11	1

comparés aux codes à longueur variable (VLC). La longueur des VLC est assignée en fonction de la distribution de probabilité des symboles apparents. En effet, les VLC sont capables de coder un mot de code en un nombre variable de bits, contrairement aux FLC qui utilisent une longueur fixe. Certains symboles peuvent apparaître plus fréquemment que d'autres et peuvent donc être codés avec des codes plus petits. Les VLC comportent des longueurs de code implicite, c'est ce qui les caractérisent car aucun délimiteur n'est requis entre chaque mot de code [57].

Considérons une séquence S = "AABDCDDD" composée de huit caractères et qui utilise quatre symboles différents. Leurs probabilités d'apparition sont présentées dans le Tableau 2.1. Dans cet exemple, il faut au minimum deux bits pour coder chaque symbole avec une longueur fixe. En revanche, les VLC peuvent être codés de différentes manières. Les deux exemples suivants montrent que les VLC doivent être choisis soigneusement afin de pouvoir être correctement interprétés dans le décodage.

Exemple 1 :
Soient les mots de code à longueur variable {A:10, B:110, C:111, D:0} que nous avons défini pour représenter les symboles de la séquence S. Leurs longueurs sont liées à leurs probabilités d'apparition. Dans la plupart des cas, un VLC utilise moins de bits qu'un FLC. Le code FLC utilise toujours la même longueur de code, il est donc évident que le nombre moyen de bits reste identique. La longueur moyenne L des VLC permet de comparer ces deux codes et elle est calculée à partir de l'équation (2.4) : $L = (2)(1/4) + (3)(1/8) + (3)(1/8) + (1)(1/2) = 1.75$ bits/symbole.

Le nombre de bits requis pour représenter un symbole compressé est plus petit avec un VLC qu'avec un FLC. Cependant, la longueur moyenne du code n'est pas le seul critère à prendre en compte. Le décodage est tout aussi important que le codage. Le choix des VLC est essentiel pour assurer un décodage correct des informations. En effet, la décodabilité unique des mots de code est le principal problème des VLC. Prenons un autre exemple où le choix des VLC pose un réel problème dans le décodage des mots de code.

Exemple 2 :
Soient les mots de code à longueur variable {A:10, B:100, C:101, D:0} que nous avons défini pour représenter les symboles de la séquence S. La séquence S est alors codée comme "10101000110000" et les premiers bits "1010100" peuvent être interprétés de différentes manières, tels que "AAB", "CDB" ou encore "ACDD". Cet ensemble de bits n'est pas uniquement décodable car elle ne permet pas d'obtenir une séquence unique de symboles. On voit que le choix des VLC est essentiel

pour reconstruire les données codées, d'autant plus que cela n'engendre pas de traitements supplémentaires [58].

Code à préfixe

Un code à préfixe, également appelé *code instantané*, peut être décodé sans information complémentaire [59], [60]. Sa particularité principale est de ne posséder aucun mot ayant pour préfixe un autre mot. En reprenant les deux exemples précédents :
{A:10, B:110, C:111, D:0} est un code à préfixe,
{A:10, B:100, C:101, D:0} n'est pas un code à préfixe.

Le principal avantage des codes à préfixe est l'unicité du décodage. Le message est codé comme une séquence de mots de code sans utiliser de délimiteur. La séquence est donc décodée distinctement en trouvant chaque mot de code.

Code universel

Un code universel est un code à préfixe qui ne suppose rien sur la source et ne se concentre pas sur la distribution de probabilité des données d'origine. Ce type de code concerne essentiellement le codage de nombres entiers. Les entiers positifs sont représentés sur des mots de code binaires où la plupart des codes à préfixes ont tendance à attribuer des mots de code plus longs pour des entiers plus grands. Les codages de Levenstein, Exp-Golomb, Elias Omega, Elias Delta, Elias Gamma et Fibonacci sont de parfait exemples de codes universels et sont spécialement utilisés pour les codes entiers [61], [62], [63].

2.2 Les algorithmes de compression sans perte

La séquence de données à compresser n'est pas toujours connue à l'avance et l'efficacité d'un algorithme de compression repose sur la réduction des redondances présente dans la séquence. Les redondances peuvent être compressées de différentes manières et c'est ce qui différencie les algorithmes actuels. Les techniques récentes de compression reposent sur des modèles probabilistes permettant d'obtenir les meilleures performances [64], [65], [66]. Ces techniques sont connues pour engendrer d'importants traitements additionnels avec des temps de calculs conséquents, ce qui n'est pas souhaitable dans un système embarqué de basse consommation.

L'algorithme de compression doit fournir le meilleur compromis entre le taux de compression et le temps de traitement nécessaire pour compresser les données. Les données compressées peuvent être codées de manière adaptative, semi-adaptative ou non adaptative. Dans le cas d'un compteur d'eau, la manière d'encoder va dépendre des contraintes mémoires et des limitations de calculs liées à l'électronique embarquée. À ce jour, trois familles de codage sans perte se distinguent : le codage entropique, le codage par dictionnaire et le codage par plage.

2.2.1 Codage entropique

Le codage entropique, issu de la théorie de l'information, est une méthode de codage statistique à longueur variable [67]. Cette méthode regroupe plusieurs algorithmes qui permettent d'attribuer les mots de codes les plus courts aux symboles les plus fréquents. Ce type de codage affecte un code à préfixe pour chaque symbole (ou nombre entier) afin de rendre la longueur moyenne minimale et de se rapprocher de l'entropie de la source. Le codage de Huffman et le codage arithmétique sont les deux techniques les plus couramment utilisées pour compresser une séquence de symboles dans un texte, un alphabet ou un dictionnaire. C'est d'ailleurs pour cette raison que le codage entropique est très souvent la dernière étape de compression du codage de source. Une séquence de données peut aussi être composée d'un ensemble de nombre entiers, pouvant être codé avec des codes universels. Le codage des entiers est donc également considéré comme un codage entropique.

Codage de Huffman

Le codage de Huffman est un des codages entropiques les plus connus dans la compression de données. Introduit par Huffman en 1952, ce codage est basé sur un algorithme qui permet de compresser essentiellement les symboles [68]. Cet algorithme construit un arbre, appelé *arbre de Huffman*, à partir de la distribution de probabilité des symboles de la séquence. L'arbre de Huffman produit des codes à préfixes optimaux qui permettent de réduire la longueur moyenne de code. C'est ce qui le distingue du codage de Shannon-Fano [69] qui ne permet pas toujours d'obtenir une longueur de code optimale. Le codage de Huffman est donc, dans la plupart des cas, préféré au codage de Shannon-Fano.

Considérons une nouvelle fois la séquence d'entrée S = "AABDCDDD" à compresser. L'arbre de Huffman est construit à partir des symboles présents dans la séquence. Sa représentation est illustrée sur la Fig. 2.2. On peut constater que les symboles les plus fréquents sont codés dans les mots de codes les plus courts. Cette distribution de probabilité est obtenue avec une méthode de modélisation simple et la longueur de la chaîne de bits compressée est plus petite qu'avec un codage binaire avec des codes de longueur fixe. C'est encore plus vrai pour de longues séquences de données avec beaucoup de redondances.

L'arbre de Huffman présenté sur la Fig. 2.2 n'est pas la seule représentation possible car les symboles B et C apparaissent avec la même probabilité. Les mots de code "110" et "111" peuvent donc être inversés, sans pour autant affecter la longueur de la séquence compressée. Le codage de Huffman est très répandu dans la compression de données car son implémentation est simple. Le temps de traitement nécessaire pour compresser les données est très acceptable, bien qu'il dépend fortement du type de codage utilisé. En effet, il existe deux types de codage de Huffman : le codage de Huffman statique et adaptatif [70], [71]. Le premier compresse les données de la même manière qu'un codage non-adaptatif, tandis que le deuxième met à jour l'arbre de Huffman après chaque nouveau symbole compressé. Ce processus de mise à jour augmente

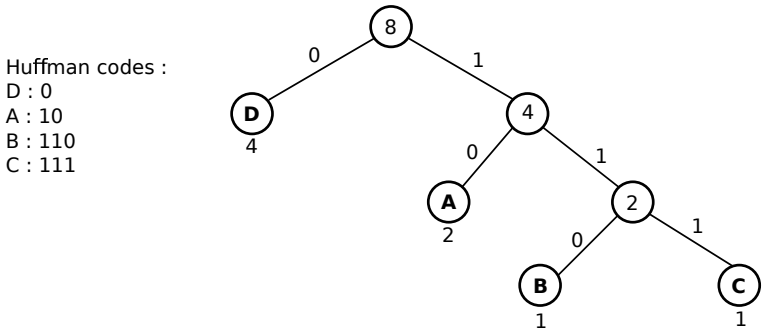


Figure 2.2 – Exemple de construction d'un arbre de Huffman à partir de la séquence S = "AABDCDDD"

le nombre de calculs et peut impacter le temps de traitement. Bien qu'optimal pour coder les symboles séparément, le codage de Huffman n'obtient pas toujours les meilleures performances.

Codage arithmétique

Le codage arithmétique est un autre type de codage entropique. Il est réputé pour offrir un taux de compression meilleur que celui obtenu par le codage de Huffman. Basé sur son précurseur Shannon-Fano-Elias, le codage arithmétique utilise un nombre flottant pour représenter un morceau de symbole [72]. Ce nombre est inclus dans un intervalle proportionnel à la probabilité d'apparition du symbole. L'ensemble de ces nombres est ensuite regroupé dans un tableau statistique fixe qui permet de coder le dernier nombre flottant [73], [74]. L'avantage de ce codage est que la taille du code optimal peut être inférieur à 1 bit, ce qui n'est pas le cas du codage de Huffman [75]. En revanche, le fait d'utiliser un tableau statistique fixe limite considérablement l'espace de stockage des intervalles [76]. La solution la plus efficace est donc un codage adaptatif où les intervalles sont mis à jour après chaque symbole rencontré [77]. L'utilisation de nombres à virgule est plus lourd à manipuler et cela complexifie davantage le traitement et l'encodage des données. Le codage arithmétique n'est donc pas considéré dans cette étude.

Codage des entiers

Le codage des entiers est un codage entropique très utilisé pour encoder les nombres entiers avec des codes à préfixe. Les codes à préfixe sont assignés en fonction de la taille du nombre entier à compresser. Autrement dit, les algorithmes utilisés assignent des mots de code plus longs à des entiers plus grands. Ce type de codage est très répandu et souvent considéré comme une composante essentielle dans le fonctionnement d'autres algorithmes de compression [78] [79].

Codage unaire

Le codage unaire est l'un des codages des entiers les plus simples pour représenter des nombres positifs n. Construit sur la base 1, ce codage est optimal pour coder une séquence dont la

Tableau 2.2 – Comparaison des longueurs de code d'Elias

Entiers n		1	2	3	4	5	10	50	64	100	10^3	10^4
Longueurs de code (bits)	Code γ	1	3	3	5	5	7	11	13	13	19	27
	Code δ	1	4	4	5	5	8	10	11	11	16	20
	Code ω	2	3	4	4	5	7	10	10	11	16	20

distribution est non uniforme. Chaque nombre entier positif n est représenté par n occurrences de 0. La fin du mot de code est marquée par un 1 faisant office de délimiteur. Par exemple, les nombres {1, 2, 3, 4} sont codés comme suit {01, 001, 0001, 00001}. La longueur L du mot de code unaire associé au nombre entier n s'écrit donc : $L = n + 1$.

Notons que le codage unaire est rarement utilisé seul car ces performances de compression sont limitées et inférieures aux autres codages entropiques. En revanche, un code unaire peut servir dans d'autres algorithmes (e.g., Elias, Golomb et Rice) du fait de sa simplicité et sa rapidité. En effet, le principal avantage du codage unaire est son temps de traitement et sa faible complexité.

Codages Elias

Très populaire pour sa contribution à la théorie de l'information, Peter Elias a proposé le célèbre codage de Shannon-Fano-Elias qui est connu pour être le précurseur du codage arithmétique. Il est considéré également pour être à l'origine de trois codes universels utilisés pour encoder des nombres entiers strictement positifs [80].

Le premier est caractérisé par le code Elias gamma (code γ). Le code γ encode un nombre entier n avec un codage binaire, puis utilise le codage unaire pour représenter le nombre de bits nécessaires pour coder n. Par conséquent, le décodeur sait exactement combien de bits doivent être lus car le nombre unaire est décodé en premier. C'est ce qui le distingue d'un code à longueur fixe. Cette technique encode chaque nombre entier avec une longueur L utilisant $2(\log_2(n)) + 1$ bits. Le deuxième code universel est le code omega (code ω) qui est basé sur le même principe. Le code ω encode le nombre entier n en binaire et fait précéder sa représentation avec un code à préfixe. Contrairement au code γ, le code ω n'utilise pas un codage unaire mais un code omega. Ce codage est donc récursif. Enfin, le troisième code d'Elias est le codage delta (code δ). Ce dernier utilise le code-γ pour obtenir un code sans préfixe. Sa longueur peut se calculer ainsi : $L = \log_2(n) + 2(\log_2(\log_2(n) + 1)) + 1$ bits. Le Tableau 2.2 compare la longueur des codes Elias γ, δ et ω. On peut constater que code δ permet d'obtenir des longueurs de code asymptotiquement meilleures. Les codes γ et ω sont généralement utilisés lorsque les petits entiers (inférieurs à 8) apparaissent plus fréquemment que les plus grands [80]. Souvent comparé au codage d'Even-Rodeh [81], le code ω est construit à partir de cinq étapes de codage, le rendant donc plus lourd en terme de calculs que les deux autres codes d'Elias.

Tableau 2.3 – Exemple des codes Exp-Golomb attribués à partir des entiers n pour différents paramètres de k

n (non signés)	n (signés)	Codes exp-Golomb			
		k=0	k=1	k=2	k=3
0	0	1	10	100	1000
1	1	010	11	101	1001
2	-1	011	0100	110	1010
3	2	00100	0101	111	1011
4	-2	00101	0110	01000	1100
5	3	00110	0111	01001	1101
...					
10	-5	0001011	001100	01110	010010

Codage exponentiel-Golomb

Le codage exponentiel-Golomb (ou exp-Golomb) est la généralisation du code γ d'Elias avec la capacité de coder tout entier non négatif, y compris le zéro. Rapidement étendu aux nombres négatifs, le codage exp-Golomb est très populaire dans les normes de compression vidéo telles que H.264/MPEG-4 AVC [82]. Le codage exp-Golomb utilise un paramètre k comme ordre pour encoder un nombre entier n. L'ordre k est un nombre entier non nul qui permet d'adapter l'efficacité du code.

Le principe est le suivant. Soit n = 10 le nombre entier non signé à coder et k=2 le paramètre choisi. Deux étapes se suivent :
1. La première étape consiste à coder u=[$\log_2(n + 2^k)$]-k avec un codage unaire. Le résultat unaire de u est 01.
2. Ensuite, il suffit d'écrire le code binaire de $n + 2^k$ (1110) en enlevant le bit de poids fort. Son résultat est alors b=110. La représentation du code exp-Golomb est la concaténation de u et de b, ce qui donne le code final 01110.

La longueur du code exp-Golomb peut donc se calculer de la manière suivante : $L = \log_2\left(n + 2^k\right) + 1$ bits. Le Tableau 2.3 montre les codes exp-Golomb respectifs aux nombres entiers n, avec plusieurs paramètres k. Notons que lorsque k est égal à 0, le code exp-Golomb est similaire au code γ. L'implémentation du codage exp-Golomb est assez simple, notamment pour une faible valeur de k. C'est ce paramètre qui définit la complexité de l'algorithme [83].

Codage de Fibonacci

Le codage de Fibonacci est basé sur le théorème de Zeckendorf [84]. Ce codage utilise la suite de Fibonacci pour encoder les nombres entiers strictement positifs avec chaque nombre de la suite qui est la somme des deux nombres consécutifs précédents. Un nombre de Fibonacci est défini par $F_i = F_{i-1} + F_{i-2}$, pour $i \geq 1$ et avec $F_0 = F_1 = 1$. Le codage de Fibonacci est un codage

Tableau 2.4 – Exemples de codes de Fibonacci respectifs à des nombres strictement positives

n	$F(n)$									Codes de Fibonacci
1	1									11
2	0	1								011
3	0	0	1							0011
4	1	0	1							1011
5	0	0	0	1						00011
10	0	1	0	0	1					010011
50	0	0	1	0	0	1	0	1		001001011
F_i	1	2	3	5	8	13	21	34	...	

d'entier où chaque bit désigne un nombre de Fibonacci F_i avec la propriété principale de ne pas contenir deux 1 consécutifs, ce dernier fait office de délimiteur [85]. La représentation binaire du code de Fibonacci $a_0 a_1 a_2 a_3 ... a_k$ est $F(n)$ qui représente un nombre positif n dans une suite de bits avec $u_i \in \{0, 1\}$, $0 \leq i \leq k$ est :

$$F(n) = \sum_{i=0}^{k} u_i F_i. \tag{2.6}$$

Le Tableau 2.4 illustre les codes de Fibonacci respectifs aux nombres entiers n. Le codage de Fibonacci est essentiellement adapté au codage de petites valeurs de n car le codage d'entiers plus grands peut affecter l'efficacité du code. Les codes de Fibonacci conviennent donc au codage de séquences courtes; ils n'utilisent pas de délimiteurs supplémentaires.

Autres types de codage

D'autres codages existent pour compresser des nombres entiers. Certains d'entre eux sont même des variantes des algorithmes déjà présentés. Un cas intéressant est le codage de Levenstein [86] qui code de manière récursive tous les entiers non négatifs de la même manière que le code ω d'Elias. Cependant, ce code est toujours un bit plus long, mais avec une implémentation plus simple [87]. Les codes d'Elias sont également utilisés dans le codage Even-Rodeh [81] afin d'optimiser la longueur des codes pour les plus petits entiers (inférieur à 8). Ce codage est souvent comparé au code ω car il peut conduire à des coûts de calculs supplémentaires. Dans certains cas, le temps de traitement peut être un critère plus important que le taux de compression. C'est le cas du codage de Golomb-Rice par exemple, qui est utilisé pour son traitement très rapide [88]. Optimal pour toute distribution géométrique, ce code à préfixe est souvent employé par d'autres algorithmes tels que le codage par plage [89]. Ce dernier sera détaillé dans la section 2.2.3.

2.2.2 Codage par dictionnaire

Le codage par dictionnaire appartient à une classe d'algorithmes de compression connus pour réduire la taille des données à l'aide d'un dictionnaire. Ces algorithmes cherchent les correspondances entre les données non compressées et le dictionnaire. Ce dernier est construit en fonction des caractères déjà rencontrées dans la séquence de données. Lorsqu'une correspondance est détectée, celle-ci est remplacée par la position du caractère dans le dictionnaire. Le dictionnaire peut être géré de différentes manières et c'est ce qui différencie les différents algorithmes actuels.

LZ77 et LZ78

Le codage par dictionnaire a commencé avec les algorithmes LZ77 et LZ78 proposés par Abraham Lempel et Jacob Ziv en 1977 et 1978 [90], [91]. L'algorithme LZ77 utilise une fenêtre glissante pour construire le dictionnaire. Le caractère déjà rencontré est référencé par sa position p dans le dictionnaire et par sa longueur l. Lorsqu'il y a de fortes redondances de symboles dans la séquence à compresser, un ensemble de caractères est alors référencé dans le dictionnaire. C'est tout l'avantage de cette technique de compression. Une paire $P(p, l)$ remplace donc le caractère déjà rencontré. Si aucun caractère n'est trouvé dans le dictionnaire, le caractère est codé par la paire P suivi du nouveau caractère. C'est l'inconvénient principal de cette technique car, dans ce cas-là, la longueur du code compressé est plus grande que celle du code original.

Basé sur LZ77, l'algorithme LZ78 remplace la fenêtre glissante par un dictionnaire global fixe permettant d'obtenir un nombre de correspondance plus important. Il résout les principaux inconvénients de LZ77 mais peut parfois être contraint par la taille du dictionnaire notamment dans les systèmes où les ressources mémoires sont limitées.

Ces deux algorithmes ont inspiré d'autres variantes de codage par dictionnaire avec par exemple l'algorithme LZSS. Ce dernier utilise un bit supplémentaire pour indiquer si le caractère codé provient d'une paire P ou d'un caractère seul. Autrement dit, l'inconvénient du LZ77 est totalement supprimé dans LZSS, c'est d'ailleurs pour cette raison que LZSS est utilisé dans les logiciels d'archivage RAR et ARJ [92]. LZ77 est également à l'origine d'autres algorithmes, souvent couplés à des codages entropiques pour améliorer les performances de compression [93]. Très utilisés dans la compression de format de fichier ZIP, les algorithmes LZMA et DEFLATE en sont de parfaits exemples [94].

LZW

Un des codages par dictionnaire les plus connus est l'algorithme Lempel-Ziv Welch (LZW) qui est basé sur LZ78 [95]. Développé par Terry Welch en 1984, le codage LZW est un algorithme très répandu, mais également reconnu pour sa capacité à être facilement adapté aux systèmes embarqués à faibles ressources [96]. LZW construit dynamiquement un dictionnaire, permettant de remplacer la séquence de caractères en une séquence de pointeurs. Les pointeurs dans

Tableau 2.5 – Principe de fonctionnement de l'algorithme LZW pour le codage de la séquence S="ABRACADABRACADABRA"

Mot m	Prochain caractère c	Position	Code binaire	Dictionnaire ϕ $(C : m)$
A	B	1	10100	27 : AB
B	R	2	01111	28 : BR
R	A	18	00010	29 : RA
A	C	1	00101	30 : AC
C	A	3	01111	31 : CA
A	D	1	10010	32 : AD
D	A	4	01110	33 : DA
AB	R	27	01111	34 : ABR
RA	C	29	010100	35 : RAC
CA	D	31	011011	36 : CAD
DA	B	33	011101	37 : DAB
BR	A	28	011111	38 : BRA
A	#	1	000001	
#		0	000000	

le dictionnaire sont des entiers positifs indiquant la position de la séquence de caractères rencontrée dans le dictionnaire. Son principe de fonctionnement est le suivant. L'algorithme LZW maintient un dictionnaire ϕ et un compteur C relatif au nombre de caractères dans le dictionnaire. Initialement, ϕ contient le même nombre de bits pour chaque symbole de l'alphabet Γ, avec C égal à la longueur de Γ. Chaque fois qu'un mot m est ajouté à ϕ, le compteur C est incrémenté et le nouveau mot est stocké à la position K-1. Pour tout mot donné m, $\phi[m]$ renvoie la position de m dans ϕ. Tant que le mot m est contenu dans le dictionnaire ϕ, le prochain caractère c est lu puis ajouté à m. Cette condition est maintenue jusqu'à ce que le nouveau mot formé (c+m) ne soit pas contenue dans ϕ. Auquel cas, c+m est stocké dans ϕ à la position C, et C est incrémenté de 1. L'algorithme encode donc la valeur du mot et poursuit le codage de la séquence. Enfin, un symbole spécial (#) est utilisé pour indiquer la fin de la séquence.

Soit S=*ABRACADABRACADABRA*, la séquence à encoder. Nous avons défini un dictionnaire ϕ composé des symboles de l'alphabet Γ. C est donc initialisé à 26, où chaque symbole est initialement codé sur 5 bits. Le codage de S est illustré dans le Tableau 2.5. En utilisant un codage binaire, LZW permet de coder S avec une longueur de 53 bits, au lieu de 90 bits pour la séquence originale. Ce résultat peut être optimisé avec l'utilisation de codes à longueur variable pour encoder les pointeurs. Dans la littérature, l'algorithme LZW est souvent couplé à un codage d'entier.

2.2.3 Codage par plage

Le codage par plage, en anglais *Run-Length Encoding* (RLE), est le plus simple des algorithmes de compression de données sans perte. Conçu et adapté pour des séquences successives de symboles, RLE détecte toutes les séquences répétitives et les remplace par une paire (S_i,n_i), avec S_i le symbole détecté et n_i le nombre d'apparition.

Soit *S=AAAAAABAAAABCDEF*, la séquence à compresser. L'encodage des premiers caractères est très efficace et l'algorithme RLE compresse les 11 premiers caractères successifs comme suit *6A1B4A*. En revanche, cet algorithme n'est pas performant pour la suite de la séquence *BCDEF* car elle ne contient pas de symboles répétitifs. La séquence compressée correspondante *1B1C1D1E1F* est plus longue que la séquence originale. Afin d'éviter ce phénomène, l'algorithme RLE est souvent précédé d'algorithmes de transformation ou de prétraitement. Ces algorithmes permettent de réorganiser les caractères dans la séquence et donc d'augmenter les redondances successives [97]. Les données réorganisées favorisent les plages plus longues de données et c'est dans ces conditions que l'algorithme RLE est le plus performant.

2.2.4 Transformations

Certains algorithmes de compression nécessitent un prétraitement pour réorganiser les données avant l'étape de codage. Au lieu de compresser directement les données, la séquence est transformée de manière à favoriser les redondances dans les données d'origine. Cette étape de prétraitement n'est pas conçue pour compresser les données mais elle est généralement utilisée pour améliorer par la suite la performance du codage par plage et des codages entropiques.

Move-To-Front

L'algorithme Move-To-front (MTF) est une technique utilisée dans la compression de données pour transformer une séquence de symboles en une séquence de nombres entiers [98]. Cette technique initialise un alphabet de longueur fixe relatif au nombre de symbole potentiellement codable. Chaque symbole est lié à un nombre entier indiquant sa position dans l'alphabet. Dès lors qu'un symbole est détecté, l'algorithme remplace le symbole par sa position. Le dernier symbole détecté devient donc le premier symbole du nouvel alphabet, ce qui décale d'une position l'ensemble des symboles précédent. Par exemple, la séquence *S=ABRACADABRA* se transforme donc en T={0,1,17,2,3,1,4,1,4,4,2}. La séquence transformée contient plus de redondances, ce qui est favorable à un codage entropique ou un codage d'entiers.

Transformée de Burrows-Wheeler

La transformée de Burrows-Wheeler (BWT) est une autre méthode permettant de réorganiser les données dans un bloc d'occurrences [99]. Tout d'abord, la séquence à compresser S est

Tableau 2.6 – Principe de fonctionnement de l'algorithme BWT, illustrant la transformation de la séquence *S=ABRACADABRA* en une séquence *T=3RDARCAAAABB*

Listes construites à partir de S	Listes triées par ordre lexicographique	Indice
ABRACADABRA	AABRACADAB**R**	1
AABRACADABR	ABRAABRACA**D**	2
RAABRACADAB	ABRACADABR**A**	**3**
BRAABRACADA	ACADABRAAB**R**	4
ABRAABRACAD	ADABRAABRA**C**	5
DABRAABRACA	BRAABRACAD**A**	6
ADABRAABRAC	BRACADABRA**A**	7
CADABRAABRA	CADABRAABR**A**	8
ACADABRAABR	DABRAABRAC**A**	9
RACADABRAAB	RAABRACADA**B**	10
BRACADABRAA	RACADABRAA**B**	11

copiée dans un tableau. Ce dernier est utilisé pour transformer la séquence S d'origine dans une nouvelle séquence. A chaque nouvelle ligne du tableau, l'ensemble des symboles est décalé vers la droite afin d'obtenir plusieurs autres séquences, et ce jusqu'à retrouver la première séquence S. Ces lignes sont ensuite triées par ordre lexicographique. La séquence transformée T correspond à la concaténation des derniers symboles provenant des listes triées. L'indice du tableau où se trouve la séquence S est ajouté au début de T. Cet indice est utile dans le processus de décodage pour retrouver S dans le tableau reconstruit.

Soit *S=ABRACADABRA*, la séquence à transformer. Les étapes de transformation sont illustrées sur le Tableau 2.6. La transformation de S a permis d'obtenir une séquence *T=3RDARCAAAABB*. Par conséquent, T contient une plage *AAAABB* pouvant être compressée plus efficacement avec un codage RLE ou un codage entropique.

En pratique, la transformée de Burrows-Wheeler est utilisée en conjonction de l'algorithme MTF. Ce dernier est à la base de l'algorithme Bzip2, où les redondances sont amplifiées avant d'être encodées avec un codage de Huffman [100].

Codage différentiel

Le codage différentiel est un prétraitement utilisé pour transformer une séquence de données $S=s_1, s_2, ..., s_i$ par la différence entre les données successives s_i et s_{i+1}. Bien que ce prétraitement est avant tout considéré comme une transformation, le codage différentiel peut également se comporter comme un algorithme de compression. En effet, cet algorithme peut réduire la taille des données originales, notamment pour la compression des versions logiciels et de séries temporelles.

2.2.5 Discussion

Plusieurs algorithmes de compression sans perte ont été présentés dans cette section. Le codage entropique se distingue par sa capacité à encoder des symboles et des nombres entiers, en utilisant des codes statistiques à longueurs variables (VLC). La manière d'utiliser les VLC diffère en fonction du choix de l'algorithme de compression (codage des entiers ou codage de Huffman). Typiquement, le codage de Huffman est adapté pour encoder une séquence de caractères dans un alphabet connu (e.g., ASCII), mais il peut très bien être modifié et adapté à l'usage d'un alphabet personnalisé. Il en est de même pour le codage par dictionnaire, où le choix de l'algorithme va dépendre notamment des ressources mémoires et de la nature des données à compresser.

Comme définit dans la section 1.3, la stratégie de collecte de données proposée est basée sur l'encodage d'un profil de consommation $P(t_i)$. Ce profil est caractérisé par une série temporelle composé d'un certain nombre d'entiers relatifs à la consommation de l'eau. Ce type de données est très favorable à l'utilisation de codes entiers permettant d'encoder la série temporelle avec des VLC. Ces codes ont l'avantage d'encoder les nombres entiers en fonction de leurs fréquences d'apparition et de leurs tailles. Lorsque la taille des nombres entiers varie fortement, le codage des entiers devient plus avantageux qu'un codage binaire. Mais c'est d'autant plus vrai quand les nombres entiers sont petits.

Prenons l'exemple d'une série temporelle $S = \{4320, 124, 42, 1035, 1162, 1750, 2003, 456218, 6313, 85733, 385733\}$ dans laquelle la représentation binaire des nombres fluctuent fortement. Un codage binaire permettrait d'encoder chaque nombre avec la même longueur de code. Celle-ci correspond à la plus grande valeur de S (19 bits). L'efficacité de code n'est donc pas toujours optimale et peut être améliorée en fonction de la fréquence d'apparition de chaque digit, ou encore par la taille de sa représentation binaire. La Fig. 2.3 illustre la comparaison entre un codage binaire (code à longueur fixe) et les codages de Fibonacci, exp-Golomb, Even-Rodeh et Omega d'Elias étant des VLCs. L'avantage de ces derniers est qu'aucun délimiteur ne soit requis entre chaque nombre entier codé. Ce n'est pas le cas des codes binaires. Les codes entiers sont encore plus efficaces lorsque les nombres entiers à coder sont petits. Le Tableau 2.7 montre que le codage de Fibonacci permet d'obtenir des longueurs de code très proche des codes binaires, et asymptotiquement meilleures que les autres codes entiers. Une alternative serait de créer une nouvelle méthode de compression hybride afin de profiter de l'avantage de plusieurs algorithmes existants. Cette nouvelle méthode permettrait d'améliorer le taux de compression, sans pour autant augmenter le temps de traitement.

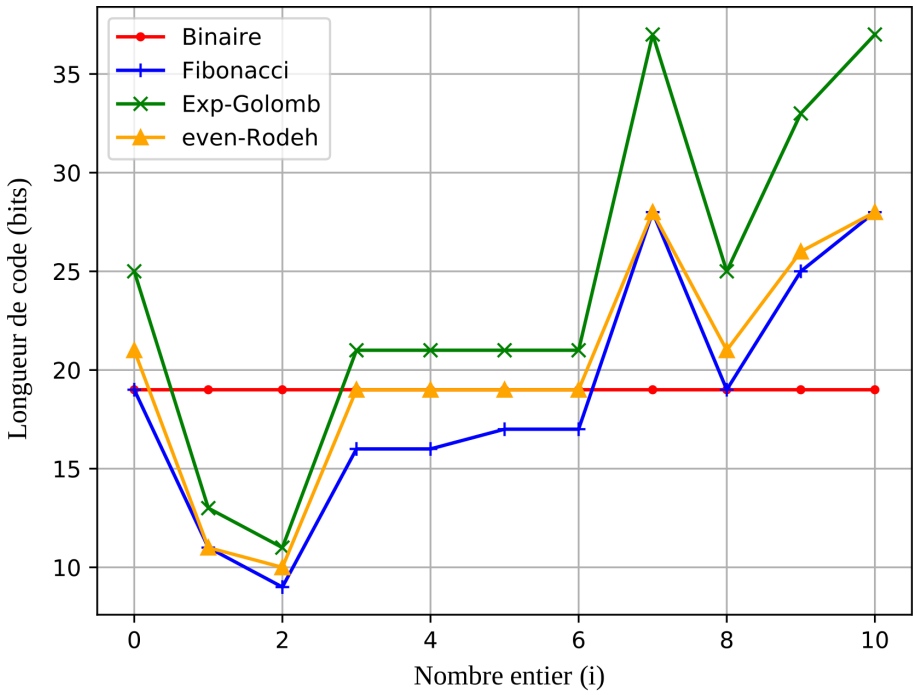


Figure 2.3 – Comparaison du nombre de bits nécessaire pour coder les nombres entiers de la séquence S

Tableau 2.7 – Comparaison entre les codes universels et non universels pour l'encodage des nombres entiers positifs

Entiers	**0**	**1**	**2**	**3**	**4**	**5**	**10**	**50**
Longueur des mots de code (bits)								
Codes binaire	1	1	2	2	3	3	4	6
Codes exp-Golomb (k=0)	3	3	3	5	5	5	7	11
Codes Even-Rodeh	2	3	3	3	4	4	8	10
Codes Rice (k=2)	3	3	3	3	4	4	5	15
Codes Levenstein	1	2	4	4	7	7	8	13
Codes unaire	1	2	3	4	5	6	11	51
Codes Fibonacci	-	2	3	4	4	4	6	9
Codes Gamma	-	1	3	5	5	6	8	13
Codes Zeta (k=1)	-	1	3	5	5	6	8	13
Codes Omega	-	1	3	3	6	6	7	12

2.3 Nouvelle méthode de compression sans perte

2.3.1 Architecture générale

Une nouvelle méthode de compression sans perte est proposée afin de compresser efficacement les données mesurées par le capteur. Elle est destinée à être implémentée dans un système embarqué pour encoder des séries de nombre entiers. Cette méthode de compression est basée sur le processus général introduit dans la section 2.1.3. La Fig. 2.4 illustre le processus de compression qui est composé d'un codage de source et d'un codage de canal. Le codage de source repose sur un nouvel algorithme de compression qui permet de réduire la longueur des messages à transmettre. Le codage de canal est assuré par une stratégie qui garantit qu'aucune donnée utile ne soit perdue entre le système de comptage et le récepteur RF. Cette étape est cruciale pour ne pas perdre de message dans la transmission des données.

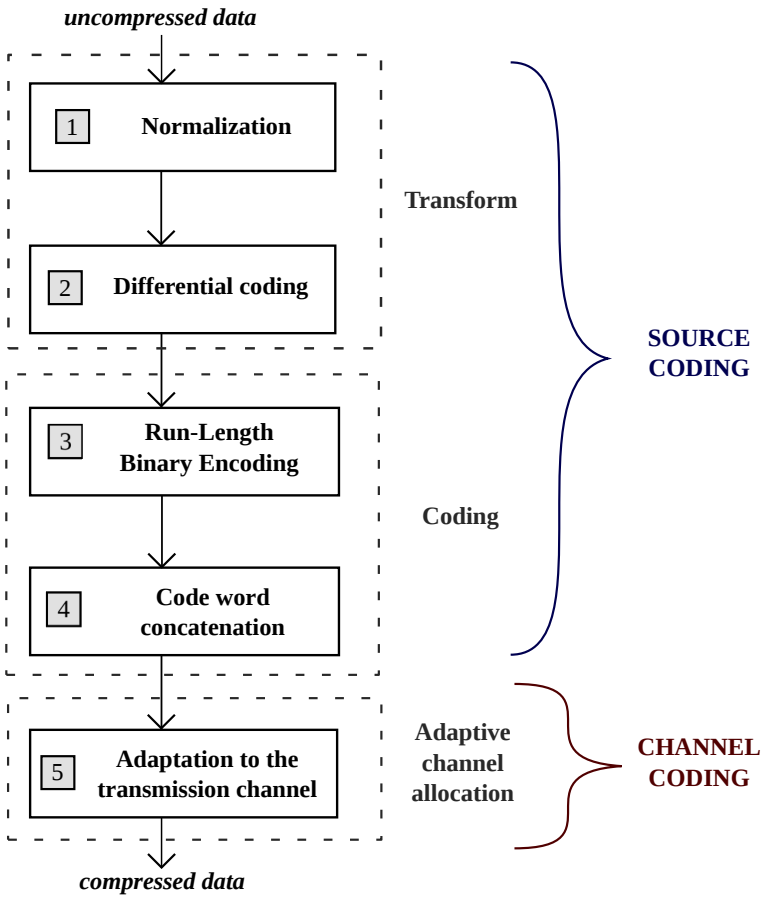


Figure 2.4 – Nouvelle méthode de compression sans perte basée sur cinq étapes de codage différentes

2.3.2 Prétraitements : normalisation et codage différentiel

Les données brutes sont des instants à la milliseconde. Ces instants correspondent à la mesure de l'écoulement d'un volume d'eau connu. Avant la compression des données, un prétraitement composé de deux étapes a été mis en place, il s'agit d'une normalisation et d'une transformation des données afin d'optimiser les performances de compression.

La normalisation consiste à distinguer le sens d'écoulement de l'eau à travers le compteur. L'écoulement de l'eau est déterminé par le sens de la variation Δ. Lorsque la variation Δ est positive, l'index s'incrémente et les événements générés correspondent à un flux positif (flow). En revanche, une variation négative correspond à un écoulement inversé (backflow). La normalisation de ces événements est illustrée sur la Fig. 2.5.

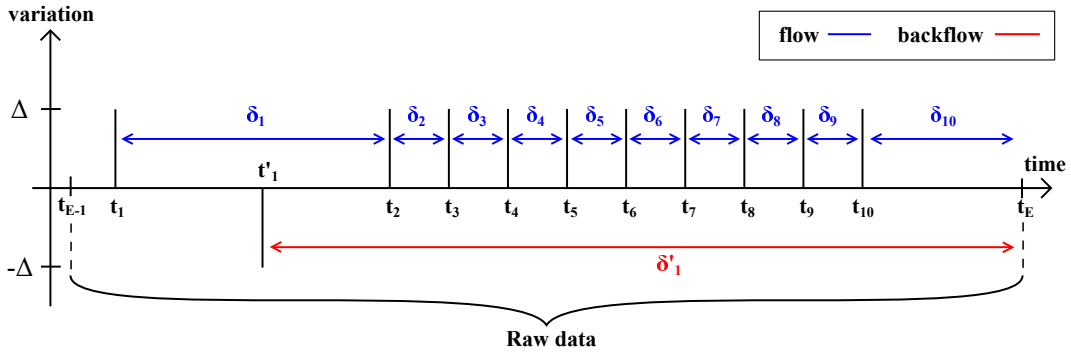


Figure 2.5 – Normalisation et transformation des données brutes entre les instants t_{E-1} et t_E

La transformation qui est proposée utilise le codage différentiel introduit dans la section 2.2.4. Le codage différentiel permet d'obtenir la différence de temps δ_i entre deux événements horodatés consécutifs t_i et t_{i+1}. Chaque valeur δ_i est calculée comme suit à partir de l'instant de référence t_E où le message est transmis :

$$\delta_i = t_{i+1} - t_i. \tag{2.7}$$

La transformation réduit la taille des données à coder et favorise les redondances. Le Tableau 2.8 illustre un exemple de codage différentiel effectué sur des données normalisées. Les instants $t_1, t_2, ..., t_{10}$ correspondent aux variations positives et t'_1 correspond à l'instant d'un événement généré par un retour d'eau (en anglais "backflow"). Ce phénomène est nettement moins fréquent et est principalement présent en cas de fuites, tentatives de fraude ou de chutes de pression dans le réseau de distribution.

L'instant de référence t_E correspond à l'instant où les données sont encapsulées juste avant la transmission. Les données normalisées t_i sont donc transformées en différences de temps δ_i. L'annexe A.5 donne un exemple de données réelles (la grandeur "flow" contient la série de δ_i).

Tableau 2.8 – Exemple d'application du codage différentiel sur des données normalisées t_i, avec t_E = 1491279519051

	Données normalisées		Différences de temps	
flow	t_1	1491277821609	$\delta_1 = t_2 - t_1$	1456391
	t_2	1491279278000	$\delta_2 = t_3 - t_2$	7062
	t_3	1491279285062	$\delta_3 = t_4 - t_3$	7126
	t_4	1491279292188	$\delta_4 = t_5 - t_4$	7062
	t_5	1491279299250	$\delta_5 = t_6 - t_5$	7125
	t_6	1491279306375	$\delta_6 = t_7 - t_6$	6938
	t_7	1491279313313	$\delta_7 = t_8 - t_7$	6438
	t_8	1491279319751	$\delta_8 = t_9 - t_8$	7062
	t_9	1491279326813	$\delta_9 = t_{10} - t_9$	7188
	t_{10}	1491279334001	$\delta_{10} = t_E - t_{10}$	185050
backflow	t'_1	1491277925173	$\delta'_1 = t_E - t'_1$	1593878

Les différences de temps δ_i sont utilisées pour être codées avec un algorithme de compression adéquat (codage entropique, par dictionnaire, etc.).

Le fait d'échantillonner à la milliseconde ne permet pas d'obtenir des plages de données favorables au codage par plage. La fréquence d'apparition de chaque symbole ne dépend pas toujours directement de la consommation de l'eau. En effet, la probabilité d'apparition d'un symbole est uniforme et influe aussi sur la redondance d'information dans les données. Ce qui ne permet de pas de gérer efficacement les redondances à compresser. L'efficacité des codages RLE, Huffman ou encore LZW n'est donc pas toujours garantie car les redondances sont relatives aux symboles présents dans les différences de temps. Un nouvel algorithme de compression sans perte est alors proposé afin de mettre en évidence des blocs de données liés à l'utilisation de l'eau.

2.3.3 Proposition d'un nouvel algorithme de compression : Le codage RLBE

Le codage *Run-Length Binary Encoding* (RLBE) est un nouvel algorithme de compression que nous avons réalisé pour encoder efficacement les différences de temps avec la résolution initialement définie. Cet algorithme a la particularité d'être insensible aux faibles écarts dans les différences de temps. Cela revient à encoder efficacement les données lorsque le débit est constant mais également lorsqu'il fluctue légèrement. Basé sur trois étapes de compression, le codage RLBE est considéré comme un algorithme de compression hybride car il est composé d'un code à longueur fixe (code binaire), d'un code à longueur variable (code de Fibonacci) et d'un codage par plage (RLE). Les étapes de compression sont illustrées sur la Fig. 2.6.

La première étape du codage RLBE consiste à encoder les δ_i et δ'_i avec un code binaire. La longueur binaire de chaque δ_i permet de créer des plages successives composées de plusieurs redondances. Ces plages de codes sont illustrés sur le Tableau 2.9 et peuvent être mises en évidence avec un codage RLE.

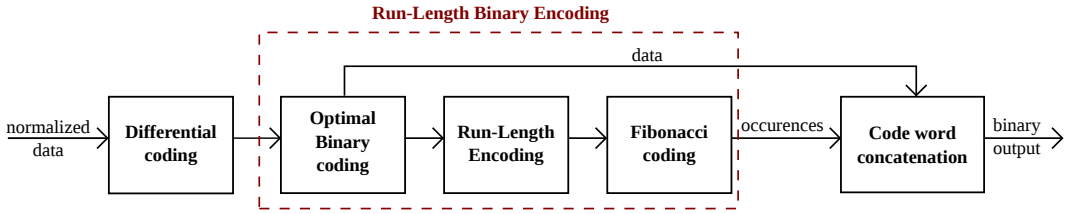


Figure 2.6 – Étapes de codage de l'algorithme de compression RLBE

L'algorithme RLE utilise les redondances dans la longueur des mots de code binaire pour créer de nouvelles plages (D_n, R_n). Les données primaires D_n sont codées avec le codage binaire et les redondances R_n avec un code à longueur variable. Comme introduit dans la section 2.2.5, un code à longueur variable comporte une longueur implicite où aucun séparateur n'est requis entre D_n et R_n. Pour le codage de petits nombres entiers strictement positifs, il a été montré que le codage de Fibonacci est généralement plus efficace que les autres codes entiers. D'autant plus qu'il permet également d'encoder des nombres plus grands. C'est pourquoi nous avons choisi ce codage pour compresser les redondances R_n. La longueur des plages est définie par la représentation binaire des données D_n, où la longueur des mots de code doit être renseignée. Un en-tête est donc requis pour délimiter chaque plage de données. La longueur maximale des codes binaires est proportionnelle à la plus grande différence de temps δ_i. Cela dépend de l'intervalle de transmission maximum T_{max} entre deux émissions t_{E-1} et t_E. Dans un premier temps, nous avons défini T_{max} à 5 minutes afin d'obtenir régulièrement une nouvelle information. Le choix de T_{max} sera d'avantage justifié dans la section 2.4.1 car ce paramètre influe sur le nombre d'émission et donc sur la consommation d'énergie. Rapporté en millisecondes, T_{max} se code en 19 bits, ce qui correspond au nombre de bits maximum L_{max} des codes binaires dans chaque plage. La représentation binaire de L_{max} est "10011" et correspond à une longueur d'en-tête H_n de 5 bits. Comme montré sur le Tableau 2.9, l'en-tête H_n est toujours codé sur 5 bits, et sa représentation dépend de la longueur α_n des codes binaires de chaque plage n.

Le codage RLBE est basé sur le codage RLE. Mais contrairement à ce dernier, les plages peuvent contenir des données qui diffèrent légèrement. Ces plages sont composées de trois champs respectifs (H_n, R_n, D_n) comme illustré sur la Fig. 2.7. Chaque plage est encodée avec un en-tête de 5 bits H_n, un champ respectif aux redondances R_n et la représentation binaire des différences de temps D_n. Lorsque des retours d'eau sont détectés, les horodatages des variations "négatives" sont encodées de la même manière que les horodatages des variations "positives". Un délimiteur codé sur 5 bits est rajouté afin de distinguer les données relatives au sens d'écoulement. Afin de ne pas être confondu avec un en-tête H_n, le délimiteur est initialisé avec cinq bits à 0. Une valeur nulle de H_n n'est pas possible car c'est ce qui indique la longueur des codes binaires de la plage. Chaque plage est ensuite concaténée en suivant la structure illustrée sur la Fig. 2.7.

La concaténation des mots de codes est la dernière étape avant la transmission des données. Les données compressées présentes dans le Tableau 2.9 sont regroupées et concaténées dans une suite de bits comme illustré sur la Fig. 2.8. La chaîne de bits concaténés est conçue pour

regrouper l'ensemble des données dans une trame commune. La trame concaténée est donc prête à être transmise.

Tableau 2.9 – Description du codage RLBE pour les différences de temps initialisées dans le Tableau 2.8

n	D_n		H_n	R_n	Code length (bits)
	δ_i, δ'_i	Binary code (α_n bits)	Decimal/Binary (5 bits)	Decimal/Fibonacci (k_n bits)	
1	1456391	101100011100100000111	21 / 10101	1 / 11	21x1
2	7062	1101110010110	13 / 01101	8 / 000011	13x8
	7126	1101111010110			
	7062	1101110010110			
	7125	1101111010101			
	6938	1101100011010			
	6438	1100100100110			
	7062	1101110010110			
	7188	1110000010100			
3	185050	101101001011011010	18 / 10010	1 / 11	18x1
4	1593878	110000101001000010110	21 / 10101	1 / 11	21x1

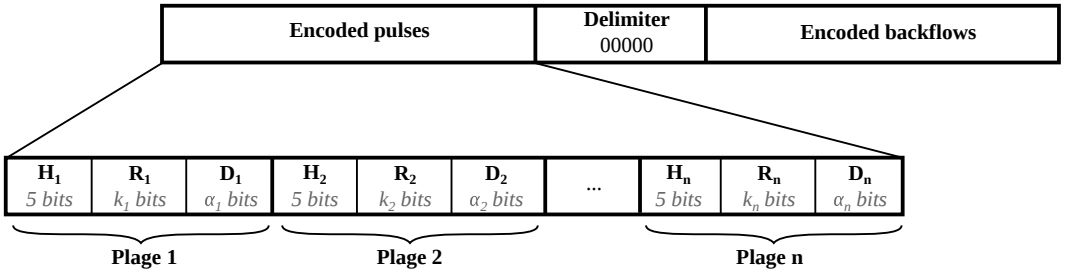


Figure 2.7 – Structure correspondante à l'algorithme RLBE pour l'encapsulation des données compressées

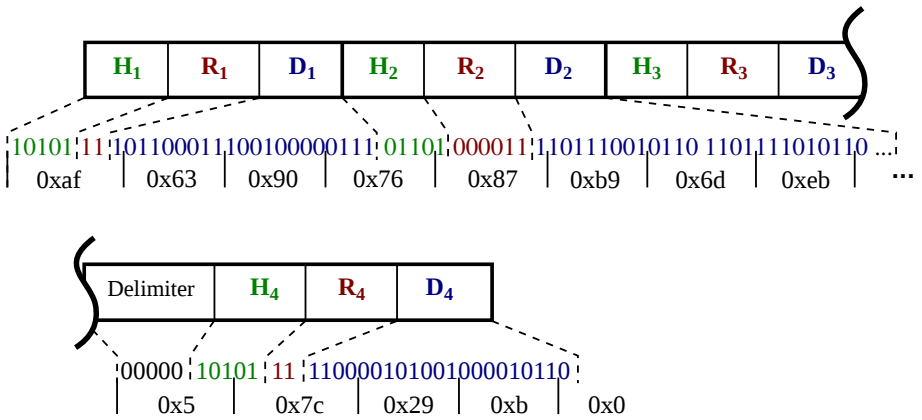


Figure 2.8 – Construction d'un bloc de données selon la structure RLBE définie sur la Fig. 2.7

2.3.4 Adaptation au canal de transmission

Les performances de transmission RF dépendent de l'environnement du compteur et du bilan de liaison. La réception des messages n'est pas toujours garantie. Les raisons peuvent être multiples, e.g., encombrement du réseau, erreurs de transmission de données, collisions de trames. La perte de trames est un phénomène qui est difficilement prédictible et cela altère la fiabilité de la transmission des données. Les trames transmises doivent être les plus courtes possibles, ce qui réduit le temps de transmission et le nombre de collision. La longueur maximale des trames est limitée par le protocole radio utilisé. A titre d'exemple, le protocole w-MBus ne permet pas d'émettre des trames plus longues que 255 octets.

Pour répondre à ces contraintes, nous avons mis en place une stratégie qui adapte la longueur des trames au canal de transmission. Cette stratégie repose sur une fenêtre glissante et est basée sur trois paramètres variables :

- la longueur maximale de la trame L_f pour limiter la taille des trames et pour minimiser le nombre de collision;
- l'intervalle de transmission maximal T_{max} pour assurer une transmission fréquente des informations;
- le nombre de redondances R_E pour répéter les données plusieurs fois et assurer la réception de l'information.

Garantir la fiabilité de la transmission des données est une des problématiques majeures dans les réseaux de capteurs sans fils, en anglais *Wireless Sensor Networks* (WSN). Typiquement, la retransmission, ou l'ajout de redondances, est la méthode la plus courante pour assurer la fiabilité des informations [101].

Au sein de la société Diehl Metering SAS, un protocole propriétaire impose d'émettre six trames successives pour répéter l'information. Cette stratégie permet d'assurer la réception de plus de 99,9% des trames. Cette variable peut être modifiée à tout moment en fonction des performances RF. Par exemple, un compteur qui est situé à proximité d'un récepteur, n'est pas amené à répéter six fois la même information. Dans ce cas, R_E peut être réduit.

La Fig. 2.9 illustre le principe de fonctionnement de la fenêtre glissante proposée. Les paramètres T_{max}, R_E et L_f sont variables et peuvent être modifiés à tout moment pour les évolutions possibles des technologies RF et des dispositifs matériels. Sur cette figure, la trame est décomposée en six paquets de l_p octets maximum car la longueur maximale de la trame est divisée par le nombre de redondance. Par conséquent, le nombre de redondance R_E définit le nombre de paquet dans la trame. Dans cet exemple, R_E = 6, les messages sont répétés six fois. C'est d'ailleurs cette valeur qui sera retenue pour les tests expérimentaux afin de respecter le protocole propriétaire de Diehl Metering. Cette technique permet d'obtenir des paquets de données homogènes, définis par une longueur maximale l_p et par un intervalle de transmission T_{max} entre deux émissions.

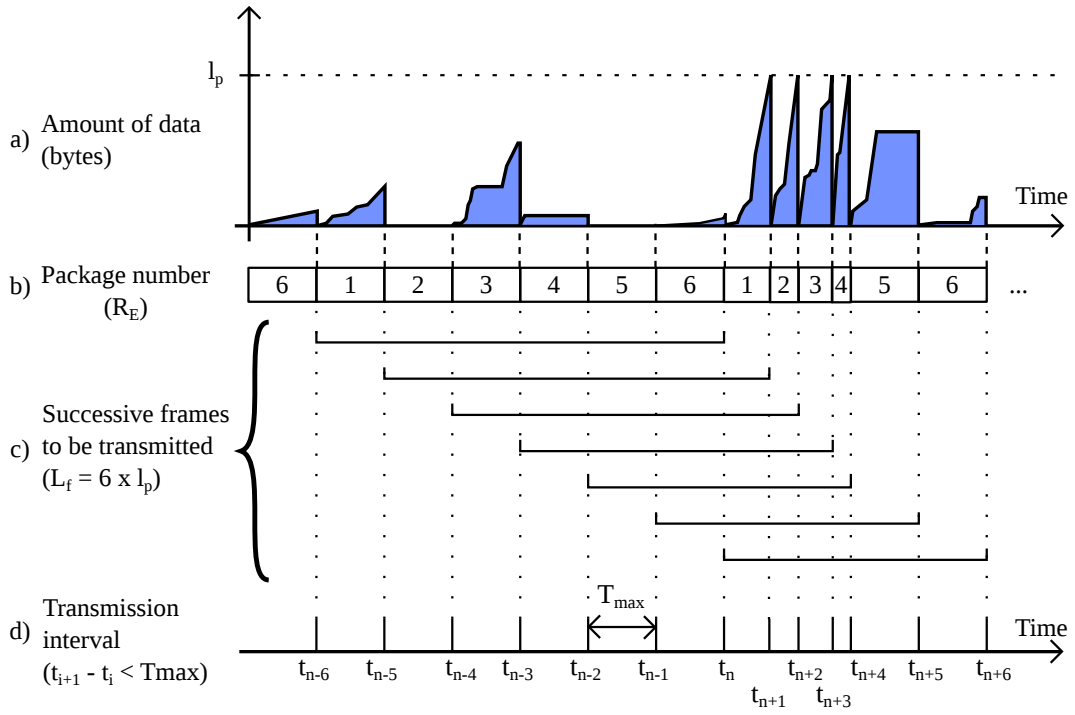


Figure 2.9 – Fiabilisation de la transmission des données à l'aide d'une fenêtre glissante basée sur les paramètres T_{max}, R_E=6 et L_f

La fenêtre glissante permet d'adapter la longueur des données et la période d'émission en fonction de la consommation de l'eau. Plus le débit est grand, plus la quantité de données à transmettre est importante. L'intervalle de transmission n'est pas fixe et dépend de la longueur l_p et du nombre de redondances R_E. Si la longueur maximale de la trame est atteinte avant la durée T_{max}, la trame est transmise pour ne pas excéder l_p. Les données qui suivent sont encodées dans le prochain paquet. Ce dernier sera envoyé dès que la première condition est atteinte entre T_{max} et l_p. Le choix des paramètres est d'avantage argumenté dans la section 2.4.1.

2.4 Résultats expérimentaux

Dans cette section, une étude expérimentale est réalisée sur la base d'un module radio utilisé pour les compteurs d'eau actuels. Les caractéristiques matérielles du module radio sont décrites dans le Tableau 1.1 du chapitre 1. Le firmware du module radio a été modifié pour intégrer les algorithmes de compression. Les données réelles sont générées et transmises à l'aide de compteurs d'eau dotés de ces modules radio intégrant les algorithmes de compression. L'objectif de cette étude est de trouver le meilleur compromis entre le taux de compression, le temps de traitement et les coûts énergétiques liés à la transmission des données.

2.4.1 Ajustement des paramètres pour la transmission

La compression des données peut réduire significativement la taille des messages, et par conséquent limiter le nombre de transmission. Avant d'évaluer les performances du codage RLBE, nous avons utilisé la fenêtre glissante proposée précédemment dans le but de fiabiliser la transmission des données. L'étude menée a pour objectif de déterminer le meilleur compromis entre les paramètres T_{max}, R_E et L_f de la fenêtre glissante. L'impact des trois paramètres a été évalué sur l'énergie consommée en transmission. Les résultats sont regroupés dans les Tableaux 2.10 et 2.11. Un codage RLBE a été utilisé. L'étude expérimentale a été menée sur des données réelles de consommation dans deux environnements différents : une habitation domestique hébergeant une personne et un bâtiment tertiaire où travaillent environ 80 personnes. Nous avons choisi ces deux environnements car le volume de consommation diffère fortement. Le volume d'eau joue un rôle important dans la quantité de données à transmettre. Afin de réduire les coûts énergétiques de la transmission, il est nécessaire de trouver un compromis entre la longueur des trames, la répétition des informations et la fréquence d'émission.

Tableau 2.10 – Étude expérimentale des paramètres de la fenêtre glissante T_{max}, R_E et L_f pour un cas domestique composé d'une personne avec une consommation d'eau de 145 litres en 24h

Information freshness T_{max}	Maximum frame length L_f (bytes)	Number of redundancies R_E	Number of transmitted frames	Total amount of encoded raw data for 24h (bytes)	Average power for data transmission (μW)
1 minute	120	1	1441	360	2.90
	120	3	1441	1021	2.95
	120	6	1441	2037	3.02
5 minutes	120	1	289	687	0.62
	120	6	291	2037	0.65
	500	1	289	687	0.62
1 hour	120	1	26	681	0.10
	120	6	34	1897	0.19
	500	1	25	686	0.09
6 hours	120	1	10	654	0.06
	120	6	22	1626	0.15
	500	1	5	685	0.05

Dans un premier temps, nous avons défini une longueur de trames L_f de 120 octets pour minimiser les collisions. Dans la pratique, au-delà de cette limite, le nombre de collision augmente drastiquement. Une longueur plus petite conduit à des émissions plus fréquentes. Ces émissions fréquentes de données sont souhaitées car elles permettent d'interagir rapidement sur le réseau de distribution. Par conséquent, l'intervalle de transmission joue un rôle important sur le nombre de message à transmettre. Plus l'intervalle est court, plus le nombre de transmission augmente. Nous avons choisi un intervalle de transmission T_{max} de 5 minutes. Cet intervalle

Tableau 2.11 – Étude expérimentale des paramètres de la fenêtre glissante T_{max}, R_E et L_f pour un bâtiment tertiaire de 80 personnes avec une consommation d'eau de 1410 litres en 24h

Information freshness T_{max}	Maximum frame length L_f (bytes)	Number of redundancies R_E	Number of transmitted frames	Total amount of encoded raw data for 24h (bytes)	Average power for data transmission (μW)
1 minute	120	1	1441	3170	3.09
	120	3	1441	9130	3.49
	120	6	1441	18 117	4.09
5 minutes	120	1	291	6067	0.99
	120	6	374	17 906	1.94
	500	1	289	6071	0.98
1 hour	120	1	85	5960	0.57
	120	6	221	17 854	1.63
	500	1	31	5992	0.46
6 hours	120	1	75	5956	0.55
	120	6	219	17 460	1.60
	500	1	19	5990	0.44

conduit à un bon compromis entre la consommation d'énergie et la fréquence de transmission. Même sans consommation d'eau et donc sans informations supplémentaires, le compteur émet toutes les 5 minutes. Ce dispositif permet au serveur d'être informé que le compteur fonctionne correctement.

Pour la suite de l'étude expérimentale, nous avons donc choisi une fenêtre glissante avec les paramètres suivants :

- intervalle de transmission maximale : T_{max} = 5 minutes;
- longueur maximale des trames : L_f = 120 octets;
- nombre de redondances : R_E = 6.

Comme le montre les Tableaux 2.10 et 2.11, les valeurs de ces paramètres constituent le meilleur compromis pour réduire la consommation d'énergie dans la transmission, garantir la réception complète de toute l'information et limiter la probabilité d'avoir une collision dans la transmission des messages. En effet, une longueur maximale de trame L_f, plus petite que 120 octets, diminuerait la probabilité de collision mais augmenterait le nombre de transmission. La longueur maximale des trames serait plus souvent atteinte, ce qui augmenterait également la consommation d'énergie. Ces paramètres sont ajustables en fonction des besoins, du protocole radio et des caractéristiques matériels de l'électronique embarquée.

2.4.2 RLBE : performances de compression

De manière générale, les performances liées aux algorithmes de compression sans perte dépendent fortement du volume et de la nature des données à compresser. L'efficacité d'un algorithme est définie par sa capacité à limiter les temps de calcul tout en réduisant au maximum la taille des données d'origine. Nous proposons d'évaluer les performances de l'algorithme RLBE que nous avons développé. Ce dernier est comparé avec d'autres algorithmes, tels que le codage de Huffman, LZW, Even-Rodeh, Exponential-Golomb, Fibonacci et l'algorithme hybride Bzip2. Ces algorithmes de compression ont été choisi car ils regroupent les principales méthodes de compression sans perte (entropique, dictionnaire, hybride). Ils ont été testés dans les mêmes conditions de fonctionnement pour calculer le taux de compression obtenu et le temps de traitement nécessaire pour compresser les données.

Chaque algorithme a été implémenté et adapté aux contraintes mémoires et CPU de l'électronique embarquée. Les tests expérimentaux ont permis de comparer la répétabilité des performances de compression avec des données réelles de consommation. La répétabilité des résultats a été effectuée à l'aide de l'écart interquartile, en anglais *Inter-Quartile Range* (IQR), qui permet de mesurer la différence entre le premier Q_1 et le troisième quartile Q_3. L'écart interquartile est utilisé pour construire de simples représentations graphiques (e.g., diagramme en boîte) d'une distribution de probabilité comme illustré sur la Fig. 2.10-a).

Le diagramme en boîte résume des caractéristiques de position telles que la médiane, le minimum, le maximum et les quartiles. Ceux-ci permettent de mesurer rapidement la variabilité des résultats. Pour les obtenir, l'étude a été menée dans plusieurs environnements différents (e.g. habitations domestiques, bâtiment tertiaire, campus universitaire). La Fig. 2.10-b) illustre le taux de compression calculé par (2.1) pour chaque algorithme testé. Les plages de dispersion illustrées dans cette figure sont représentatives des performances de compression dans un bâtiment tertiaire utilisé par environ 80 personnes dans la journée. Les algorithmes RLBE, Bzip2, LZW et Huffman conduisent à de meilleurs taux de compression par rapport aux autres algorithmes car ils n'encodent pas les différences de temps de la même manière. Les codes à longueur variable (Even-Rodeh, Exp-Golomb, Fibonacci) attribuent des mots de code plus longs aux entiers plus grands. La résolution qui a été initialement définie engendre des différences de temps plus grandes, ce qui réduit l'efficacité dans l'usage unique de codes à longueur variable.

Le taux de compression de l'algorithme Bzip2 varie fortement avec un écart de 20% entre Q_1 et Q_3. Ses performances ne sont pas toujours répétables et cet algorithme n'est donc pas considéré comme optimal. Les codages Even-Rodeh et Fibonacci fournissent les meilleures répétabilités mais leurs performances restent inférieures aux codages de Huffman, LZW ou encore RLBE. L'algorithme RLBE fournit les meilleures performances avec une médiane qui excède 65 % de taux de compression. Ce dernier se distingue également par ses capacités à minimiser les coûts de calculs comme illustré sur la Fig. 2.11. Les coûts de calcul sont mesurés par le temps de traitement requis pour compresser les données. La Fig. 2.11 montre que les algorithmes de Huffman, LZW et Bzip2 présentent des temps de traitement élevés et non homogènes, avec un

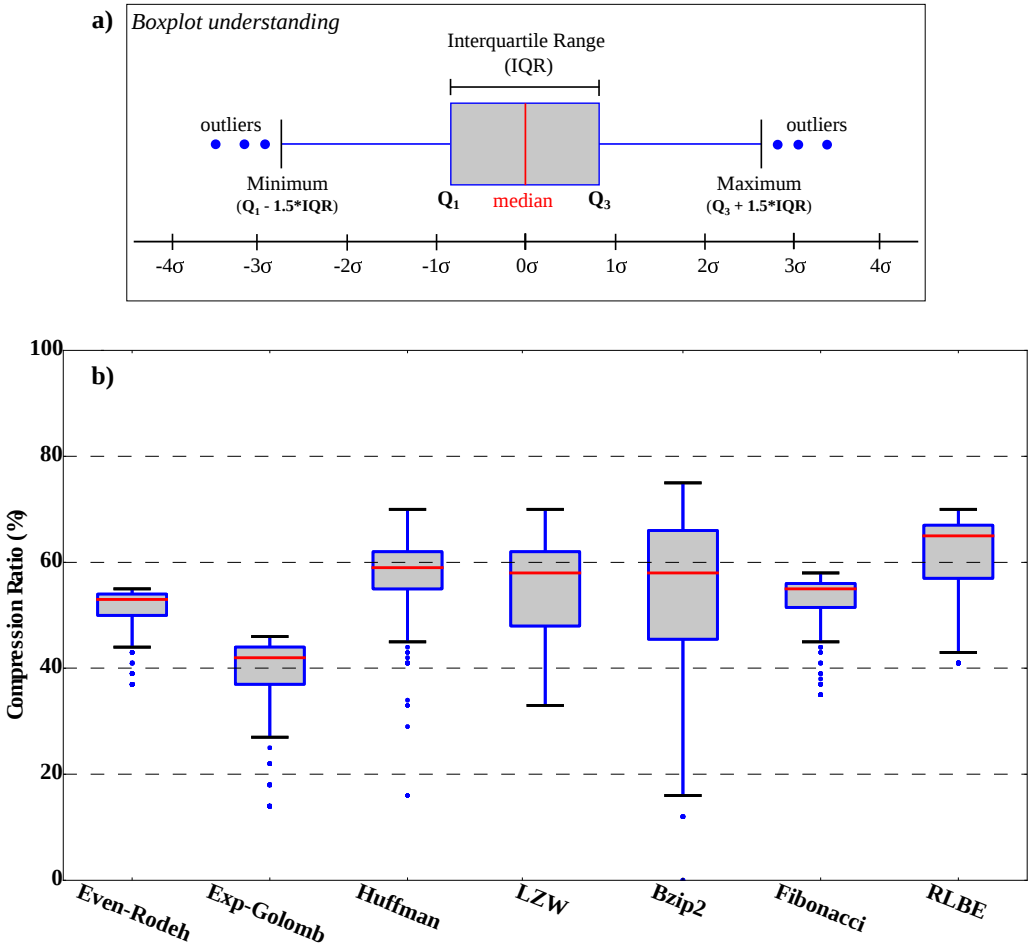


Figure 2.10 – Comparaison de la mesure de dispersion des taux de compression obtenus entre le codage RLBE et six autres algorithmes, a) principe général d'un diagramme en boîte, b) diagrammes en boite des sept algorithmes de compression

écart pouvant aller jusqu'à 700 μs. Les algorithmes exp-Golomb (k=0), Even-Rodeh, Fibonacci et RLBE sont parmi les plus rapides avec des temps de traitement inférieurs à 100 μs.

L'algorithme RLBE, qui utilise un codage de Fibonacci, encode les données plus rapidement qu'un codage de Fibonacci seul. Cela est due au fait que nous utilisons les capacités mémoires pour encoder des codes entiers. Les temps de calculs sont donc allégés. Les codes de Fibonacci correspondent aux redondances et sont directement stockées dans la mémoire flash de l'électronique embarquée. Cette technique ne peut être appliquée au codage de Fibonacci seul car celui-ci encode des valeurs différentielles δ_i nettement plus grandes. La représentation binaire des codes de Fibonacci est donc stockée en mémoire flash, avec une limite maximale de stockage. Au-delà de cette limite, une nouvelle plage de données est créée. Les Fig. 2.10 et Fig. 2.11 montrent que l'algorithme RLBE est généralement plus efficace que les autres algorithmes car il permet d'allier un fort taux de compression avec un temps de calcul très faible. Ces deux critères sont donc calculés pour l'encodage de chaque trame, ce qui représente un

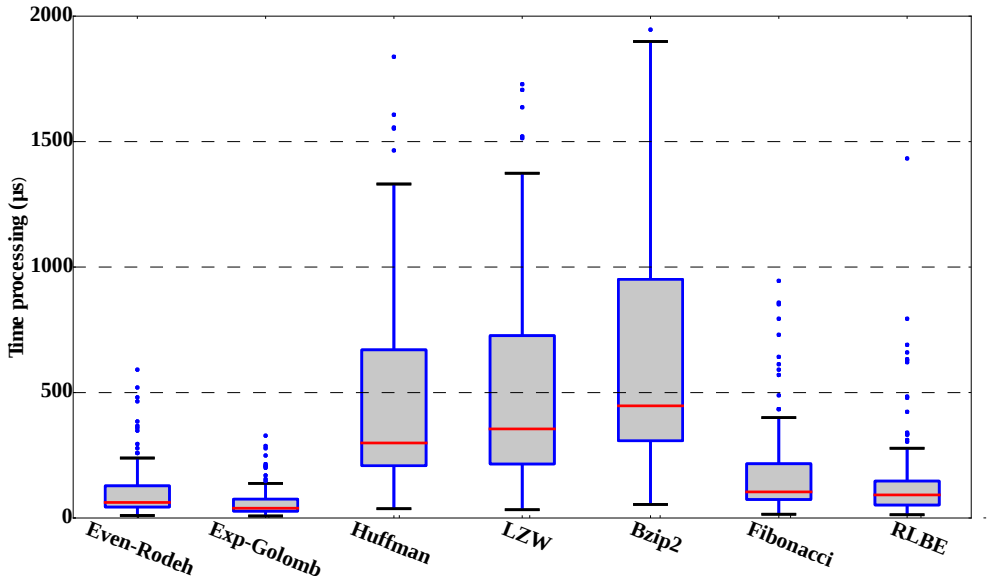


Figure 2.11 – Comparaison de la mesure de dispersion des temps de traitements entre le codage RLBE et six autres algorithmes

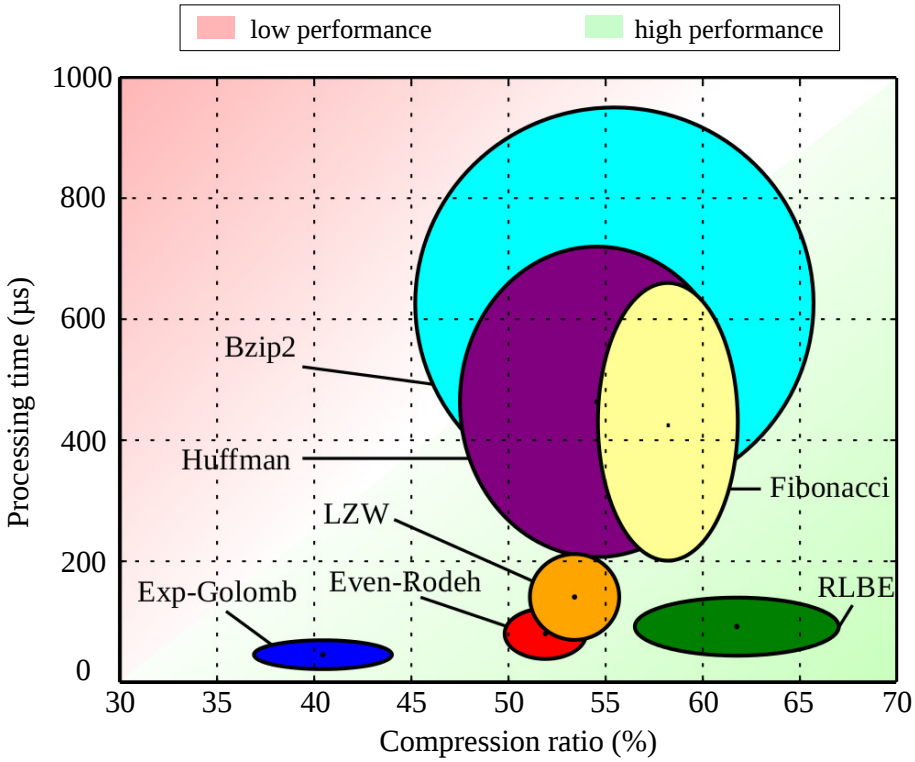


Figure 2.12 – Taux de compression et temps de traitement des algorithmes de compression testés

Tableau 2.12 – Bilan énergétique associé aux performances des algorithmes de compression sans perte étudiés pour un exemple de consommation d'eau dans un bâtiment tertiaire

Lossless Compression algorithm	Total data size per day (bytes)	Average number of encoded frame per day	Average processing time per encoded frame (μs)	Average power of data transmission per encoded frame (μW)
No compression	50 760	706	61	4.79
Even-Rodeh	22 614	413	179	2.33
Exp-Golomb	26 970	472	121	2.74
Static Huffman	19 595	389	758	2.08
LZW	21 653	406	818	2.25
Bzip2	18 940	380	1113	2.02
Fibonacci	20 872	408	242	2.21
RLBE	17 910	373	163	1.94

peu moins de 400 points de mesure acquises en 24h par un compteur mécanique DN32 dans un bâtiment tertiaire. Ces points de mesure ont été illustrées par des ellipses dans la Fig 2.12 afin d'identifier rapidement l'algorithme de compression optimal. Le taux de compression et le temps de traitement sont tout aussi important l'un que l'autre pour réduire la consommation d'énergie de l'électronique embarquée.

Un bilan énergétique a été effectué pour chaque algorithme de compression. Les performances de compression et l'énergie requise pour transmettre les données sont illustrées dans le Tableau 2.12. Les résultats ont été obtenus pour une consommation d'eau dans un bâtiment tertiaire composé d'environ 80 personnes. La méthode de compression proposée permet de réduire la puissance moyenne de transmission de 4,79 à 1,94 μW, soit plus de 60% de l'énergie consommée pour la transmission des données. Par conséquent, les coûts de traitements associés au processus de compression sont largement compensés par une durée de transmission plus courte.

Les coûts de transmission sont estimés avec l'utilisation d'un seul et même algorithme pour encoder toutes les données. Ces données contiennent des redondances différentes en fonction de la consommation de l'eau. Par exemple lorsque les données d'origine ne contiennent pas beaucoup de redondances, il est préférable d'utiliser un codage binaire à la suite d'un codage différentiel, plus rapide, plutôt qu'un autre algorithme, plus lent et plus coûteux en mémoire. L'utilisation d'un unique algorithme n'est donc pas optimale car les performances de compression peuvent parfois être impactées. Afin d'évaluer l'efficacité des algorithmes sur une période de temps connue, la longueur maximale L_f des messages n'a pas été considérée. En effet, ce paramètre influe sur le nombre de transmission car la quantité de données à transmettre varie en fonction de la performance de l'algorithme utilisé. Une étude expérimentale a été effectuée dans un cas domestique composé de trois personnes. Cette étude démontre que les performances de compression varient en fonction des données à encoder. L'efficacité des algorithmes a été évaluée à travers le taux de compression obtenu pour une journée typique de consommation.

Étant donné que L_f n'est pas considéré, l'intervalle de transmission est fixé à 5 minutes et n'est plus modifié. Nous avons choisi de ne pas changer le nombre de redondances, R_E reste donc égal à 6. Dans cet exemple, 288 trames sont envoyées en 24h. Pour chaque trame, l'algorithme retenu est celui avec le meilleur taux de compression. Nous avons évalué combien de fois chaque algorithme a été élu sur l'ensemble des paquets transmis sur une période de temps. Un taux d'utilisation exprimé en % a été défini pour chaque algorithme en 24h de consommation d'eau. La Fig. 2.13 montre le taux d'utilisation des huit algorithmes de compression utilisés pour transmettre la consommation dont la courbe de charge est également indiquée.

L'algorithme RLBE fournit les meilleures performances de compression dans 46.2% des cas. Le codage binaire et l'algorithme bzip2 suivent ensuite avec 28.2% et 23.4%. Les algorithmes de Huffman et LZW sont rarement optimaux et fournissent de meilleurs taux de compression pour 1.1% des trames encodées. L'algorithme bzip2, principalement utilisé pour la compression de fichier, fournit le meilleur taux de compression dans 23.4% des cas. Malheureusement le temps de traitement est trop important et coûte cher en énergie. Les autres algorithmes n'ont pas été considérées davantage dans cette étude car ils sont inefficaces pour des grands entiers lorsqu'ils sont employés seul. Dans la suite de l'étude, nous ne retiendrons que les algorithmes RLBE, Huffman, LZW et le codage binaire.

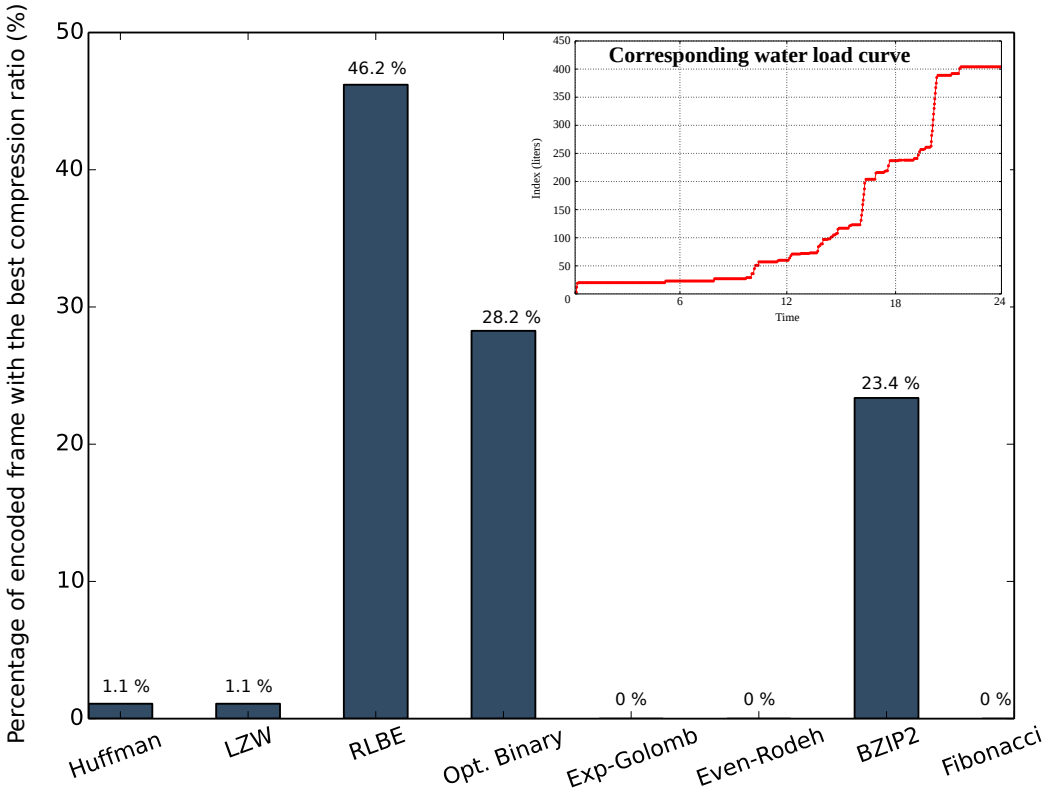


Figure 2.13 – Taux d'utilisation des algorithmes pour la compression des données d'une consommation domestique de trois personnes (taux moyen sur 24h)

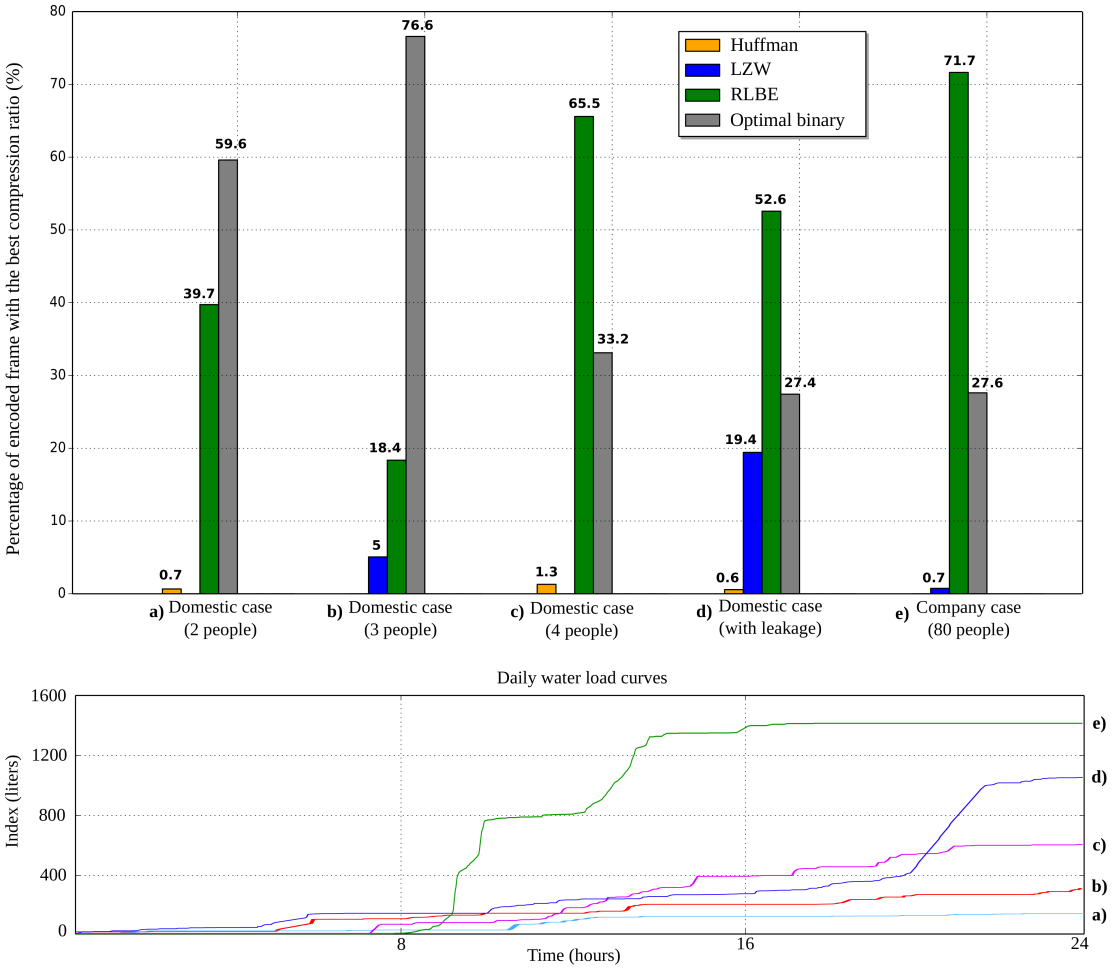


Figure 2.14 – Taux d'utilisation des algorithmes de compression testés pour l'encodage des données de consommation dans cinq environnements différents, illustrés par les courbes de charge a), b), c), d) et e)

L'étude comparative menée jusqu'ici a été menée dans cinq environnements différents. La Fig. 2.14 montre l'efficacité des codages RLBE, Huffman, LZW et binaire par rapport à des consommations différentes. Les trois premiers concernent des cas domestiques composés de deux, trois et quatre personnes. Le quatrième reprend le cas domestique de quatre personnes où figure une fuite d'eau importante. Le dernier cas correspond à la consommation d'eau d'un bâtiment tertiaire composé d'environ 80 personnes. Typiquement, le codage binaire satisfait les environnements où la consommation d'eau est très faible (cas domestiques de deux et trois personnes). Les redondances et la quantité de données à transmettre sont limitées. En revanche, les redondances d'information ont tendances à être plus nombreuses lorsque la consommation d'eau augmente. L'algorithme RLBE est plus optimal dans ce cas de figure. Le Tableau 2.13 regroupe les meilleurs algorithmes de compression pour ces cinq cas de consommation. Leurs performances permettent de maintenir un taux de compression moyen de 60%.

Tableau 2.13 – Corrélation entre la consommation d'eau et l'algorithme offrant les meilleurs performances de compression

Daily Consumption cases	Volume (liters)	Average length of daily uncompressed data (kilo bytes)	Most suited compression algorithm for energy optimization	Average compression ratio (%)
Domestic case with 2 people	265	9.2	Binary encoding	58.6
Domestic case with 3 people	483	17.3	Binary encoding	58.7
Domestic case with 4 people	519	18.7	RLBE encoding	61.0
Domestic case with a leakage detection	2521	90.3	RLBE encoding	61.8
Tertiary building with 80 people	1410	50.7	RLBE encoding	59.7

Un algorithme de compression est déterminé en fonction des redondances d'information et de la nature des données. La longueur des données d'origine influe souvent sur les redondances d'information et apporte une indication sur l'efficacité de l'algorithme. Cette indication est illustrée sur la Fig. 2.15. Nous avons comparé la performance des algorithmes de compression en fonction de la longueur des données d'origine. Le codage de Huffman et le codage binaire offrent les meilleurs taux de compression pour des longueurs de données allant jusqu'à 50 octets. Le codage de Huffman est préféré au codage binaire (code à longueur fixe) lorsque la séquence de données à compresser comprend des répétitions importantes de digits. En revanche, ce dernier offre de meilleurs performances pour des redondances de données extrêmement faibles. L'algorithme RLBE est plus efficace pour la compression des données originales comprises entre 50 et 300 octets. Pour des longueurs de données plus grande, le codage LZW tend à offrir de meilleurs taux de compression. Même si ce dernier peut être employé dès 100 octets, LZW est principalement adapté pour de longues séquences de données à compresser (> 600 octets). Compte tenu de la longueur maximale des trames L_f que nous avons défini précédemment (120 octets), l'algorithme LZW est très rarement optimal dans notre cas. En revanche, LZW est très intéressant pour des trames plus longues.

La comparaison des différents algorithmes a permis de valider les performances du codage RLBE. De manière générale, ce dernier est le plus adapté car il offre les meilleures performances de compression dans les conditions de fonctionnement souhaitées (L_f = 120 octets, R_E = 6, T_{max} = 5minutes). C'est pour ces raisons que nous l'avons implémenté dans le microcontrôleur du module radio décrit dans le Tableau 1.1 du chapitre 1.

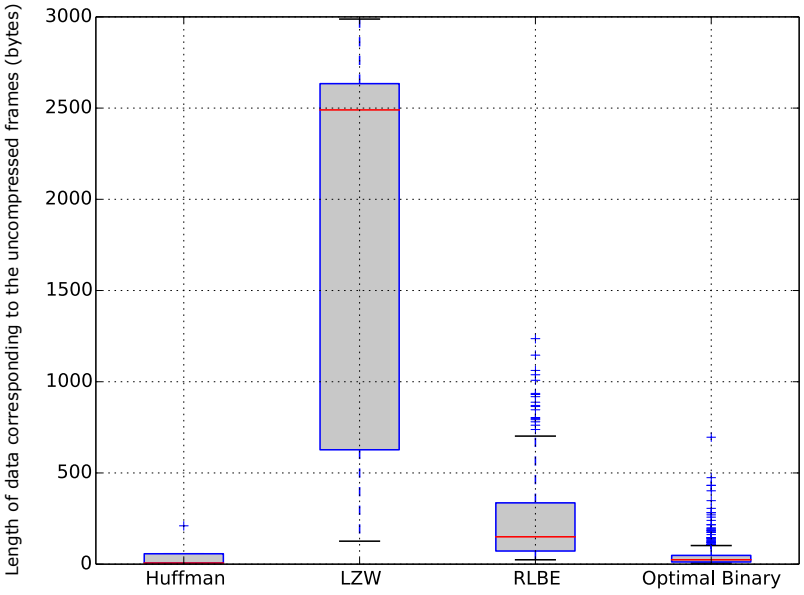


Figure 2.15 – Dispersion statistique sur les performances des algorithmes de compression en fonction de la longueur des données originales

2.5 Bilan

Les performances de compression de l'algorithme RLBE ont été validées par des prototypes installés dans des conditions réelles de fonctionnement. Ces prototypes permettent de collecter les données brutes détectées par le capteur et de les compresser sans perte d'information avec l'algorithme RLBE. La réception des données est assurée par la fenêtre glissante proposée dans la section 2.3.4. Cela permet de répéter les données compressées six fois dans six trames différentes tout en limitant les collisions pendant la transmission. Afin d'évaluer les coûts énergétiques dans le module radio, une étude expérimentale a été réalisée sur six mois de collecte de données dans trois cas de consommation différents :

- cas 1 : bâtiment domestique habité par deux personnes (compteur d'eau DN15) ;
- cas 2 : restaurant universitaire à l'IUT de Mulhouse (compteur d'eau DN15) ;
- cas 3 : arrivée d'eau principale à l'IUT de Mulhouse (compteur d'eau DN100).

Cette étude a pour objectif de comparer les performances énergétiques des modules radio commercialisés sans compression avec ceux qui intègrent l'algorithme de compression RLBE. A ce jour, les modules radio mobiles (Walk By/Drive By) permettent de collecter les données mesurées avec une résolution maximale d'environ 8 secondes. Cette cadence d'émission est dédiée aux réseaux mobiles et elle permet d'assurer la réception d'au moins une trame lorsqu'un véhicule circule à proximité. Dans cette configuration, les données transmises sont donc que très rarement réceptionnées. La consommation moyenne de transmission de ces modules radio

est d'environ 36 μW. En réseau fixe, la résolution est nettement plus faible due aux contraintes énergétiques de l'électronique embarquée. Les données sont transmises toutes les 15 ou 30 minutes ou même parfois une à deux fois par jour afin d'assurer la durée de vie du compteur. Cette résolution ne permet pas d'optimiser la gestion du réseau d'eau et d'analyser finement la consommation de l'eau. La stratégie de collecte de données proposée permet de compresser toutes les données brutes et de les transmettre toutes les 5 minutes maximum. Le Tableau 2.14 montre que l'approche de compression sans perte proposée (RLBE) a permis de réduire la consommation moyenne de transmission. L'énergie nécessaire pour compresser et transmettre les données relatives au profil de consommation est négligeable comparé à la consommation d'énergie des modules radios actuels. En effet, pour la compression et la transmission de données le module radio qui intègre nos contributions consomme entre 0,61 μW et 2,41 μW selon les cas de consommation contre 36 μW pour les modules radio mobiles actuels sans compression. Les prototypes développés permettent de réduire la longueur des messages de 61% à 64% en moyenne et montre la répétabilité des résultats dans trois cas de consommation différents sur six mois d'enregistrement. Comparés aux systèmes de communications actuels (Walk By/Drive By), la compression de données et la stratégie de transmission proposée permettent d'améliorer considérablement la consommation énergétique tout en fournissant toutes les données nécessaires à une analyse plus précise de la consommation de l'eau.

Tableau 2.14 – Analyse expérimentale et bilan énergétique des transmissions de données, compressées avec l'algorithme RLBE dans trois environnements différents

Experimental study over 183 days of data gathering from three different smart water meters		**Case 1: Domestic case (ND15)**	**Case 2: University restaurant (ND15)**	**Case 3: Whole IUTM (ND100)**
Total volume of water consumed (m^3)		49	132	5315
Volume of water consumed per day (liters)	Minimum	37	0	730
	Maximum	1502	9786	386610
	Average	268	739	29200
	Standard deviation	199	960	40410
Total number of frames received		59950	68859	93400
Number of frame received per day	Minimum	83	261	180
	Maximum	410	1055	1499
	Average	328	375	513
	Standard deviation	41	67	244
Total amount of received data (Mb)		1.7	2.3	6.6
Average amount of received data per day (kb)		9.3	12.6	36.1
Average power consumed for compression and data transmission (μW)		0.61	0.86	2.41
Data reduction with the RLBE algorithm compared to the uncompressed data (%)		64	63	61

2.6 Conclusion

Au cours de ce chapitre, plusieurs techniques de compression de données sans perte ont été mise en œuvre pour optimiser la consommation énergétique des compteurs d'eau communicants. Une stratégie d'économie d'énergie a été proposée. Cette stratégie est composée d'une nouvelle méthode de compression de données sans perte, *Run-Length Binary Encoding* (RLBE), qui permet de réduire la taille des messages à transmettre.

Les performances de l'algorithme RLBE ont été évaluées et comparées à celles des algorithmes de Huffman, Even-Rodeh, Exponential-Golomb, LZW, Fibonacci et à une version adaptée du codage hybride Bzip2. Ces différents algorithmes ont été testés avec des données réelles de consommation. Les inconvénients et les avantages de chaque algorithme ont été décrits. Il a été montré que le choix de l'algorithme doit prendre en compte les contraintes mémoires et CPU de l'électronique embarquée. Principalement adapté à des séquences successives de longueurs binaires identiques, l'algorithme RLBE offre les meilleurs compromis en termes de taux de compression et de temps de traitement.

Afin de valider les contributions proposées dans ce chapitre, l'algorithme RLBE a été implémenté dans le microcontrôleur de plusieurs module radio existants. Un bilan énergétique a été effectué pour confirmer l'efficacité de l'algorithme RLBE en conditions réelles de fonctionnement. Cette étude a permis de comparer les performances des systèmes actuels (sans compression) avec les contributions proposées. Il a été montré que la compression de données et la stratégie de transmission proposée permettent d'optimiser considérablement la consommation énergétique. L'énergie nécessaire pour compresser et transmettre les données varie entre 0,6 μW et 2,4 μW, contre environ 36 μW pour les systèmes de communication mobiles actuels. Nos contributions ont permis d'optimiser la consommation énergétique des compteurs communicants d'un facteur moyen environ égal à 20. Les contributions ne permettent pas uniquement de réduire la consommation d'énergie car les données collectées sont transmises plus régulièrement et avec une résolution nettement plus élevée (1 ms pour l'horodatage de chaque mesure).

La réception, l'exploitation et l'analyse des données de consommation seront abordées dans le prochain chapitre. Nous verrons tout le potentiel qu'offre une telle richesse d'information dans la collecte de données mise en place.

3 Exploitation des données

La stratégie de collecte de données proposée a permis de réduire la consommation d'énergie du module radio et de transmettre les données mesurées par le capteur avec une meilleure résolution temporelle. Le transfert des données brutes permet de déporter le traitement au niveau du serveur pour analyser plus précisément les consommations d'eau. Les données sont stockées dans une base de données pour assurer plusieurs services tels que l'archivage, l'analyse, la surveillance, la détection, le dépannage et la facturation.

Dans ce chapitre, nous nous intéresserons à :

- vérifier la fiabilité et la durabilité des communications entre les compteurs et le serveur sur du long terme ;
- décoder et valider la réception des données ;
- stocker les informations décodées dans la base de données ;
- évaluer la quantité de données collectées et stockées.

Cette stratégie de collecte de données permet de transmettre plus de données. Cela offre plus de précision dans les analyses de consommation et conduit au développement de nouvelles fonctionnalités.

Les fonctionnalités déportées sont décrites dans la section 3.1. Pour répondre aux objectifs décrits ci-dessus, une plate-forme expérimentale est proposée dans la section 3.2. Cette plate-forme est composée de plusieurs compteurs d'eau améliorés, de plusieurs récepteurs et d'une base de données hébergée sur un serveur. Les données compressées sont envoyées par les compteurs d'eau communicants. Le serveur les reçoit et les décode. Ce processus de décodage et de décompression est décrit dans la section 3.3. La section 3.4 évalue la quantité de données stockées dans le serveur. Les tenants et les aboutissements de l'exploitation des données collectées sont abordés dans la section 3.5. La reconstruction d'une courbe de charge, le calcul de courbes de charge moyenne, la détection de fuites sont montrés avec les données issues d'un cas d'usage. La section 3.6 est un bilan qui conclut ce chapitre.

3.1 Fonctionnalités déportées

La mesure et la collecte de données sont définies de deux manières différentes : par une mesure intrusive et non intrusive. Le système intrusif permet une analyse très précise car il est composé de capteurs installés sur chaque appareil individuellement (e.g., WC, douche, machine à laver). Ce système, appelé *Intrusive Appliance Load Monitoring* (IALM), génère des coûts importants en terme d'installation et de maintenance. Un système non intrusif, appelé *Non Intrusive Appliance Load Monitoring* (NIALM) permet de collecter les données seulement sur un seul point de mesure [102]. Les coûts d'installation sont donc largement réduits mais la précision est limitée [103], [104]. Les contributions proposées dans cette thèse permettent de collecter les données avec une meilleure résolution temporelle à partir d'un seul point de mesure. Les données brutes collectées permettent de déporter le traitement des données au niveau du serveur. L'analyse des données pourra être adaptée en fonction des besoins, sans pour autant modifier la collecte des données dans le module radio situé sur le compteur. Cette stratégie de collecte de données offre plus de flexibilité dans les analyses de consommation et plus de fonctionnalités, telles que :

- facturer l'eau en fonction de la consommation réelle ;
- reconstruire l'historique de la courbe de charge et du profil de consommation pour chaque compteur ;
- évaluer les anomalies dans le compteur (sous/sur-débits) ;
- détecter automatiquement les fraudes et les retours d'eau ;
- détecter et prévenir des éventuelles fuites ;
- optimiser la taille des compteurs et la dimension des canalisations ;
- détecter les dépôts dans le compteur (e.g., présence de sable, particules) ;
- analyser finement les données pour de la maintenance prédictive du réseau de distribution.

Pour répondre à tous ces besoins, nous avons mis en place une plate-forme expérimentale composées de plusieurs compteurs et de plusieurs capteurs environnementaux. Afin d'améliorer la distribution de l'eau, les données de consommation peuvent être corrélées avec d'autres informations environnementales (température extérieure, humidité, pression, etc.). En effet, certaines études ont relevé que les conditions météorologiques peuvent, dans certains cas, être des facteurs importants pour l'analyse de la consommation d'eau [105], [106].

3.2 Plate-forme expérimentale pour la collecte des données

3.2.1 Architecture de la plate-forme

Une plate-forme expérimentale a été conçue pour collecter les données de consommation dans des conditions réelles de fonctionnement. Afin de faciliter la mise en œuvre et pour des raisons de proximité, la plate-forme expérimentale a été installée sur le "*Campus des collines*" de l'Université de Haute-Alsace à Mulhouse. Cette plate-forme est illustrée sur la Fig. 3.1. Celle-ci

se compose de compteurs d'eau communicants, de récepteurs et d'un serveur. Les compteurs sont placés dans des endroits spécifiques pour surveiller efficacement le réseau de distribution du campus. La plate-forme est composée de deux types de compteurs : le premier concerne les compteurs existants réservés à la facturation ("*official meters*") ; le deuxième intègre les contributions proposées dans les chapitres précédents ("*enhanced meters*"). Les compteurs existants ne peuvent pas être modifiés, déplacés ou changés sans l'autorisation du distributeur. C'est pour cette raison que nous avons installé d'autres compteurs (*enhanced meters*) en série des compteurs existants. La collecte des données est effectuée à partir des nouveaux compteurs.

Le campus accueille plus de 1400 personnes chaque jour et combine diverses activités dans plusieurs bâtiments (Fig. 3.1). Nous nous sommes intéressés à collecter les données de consommation du restaurant universitaire, des bâtiments tertiaires réservés à l'enseignement et des logements de fonction situés sur le campus. Les compteurs sont installés à proximité des machines de production industrielle (e.g., l'usinage, la découpe au jet d'eau, l'injection de plastique), au niveau de la cuisine et à l'entrée de certains bâtiments couvrant des toilettes, des robinets et divers appareils. Afin d'élargir les cas d'utilisation, nous avons également installés des compteurs dans trois environnements domestiques situés à plusieurs kilomètres de l'université. Ces environnements ont été choisis pour leur différences en terme de consommation d'eau (pour un couple et pour des familles de trois et quatre personnes). Tous les nouveaux compteurs installés transmettent les données de consommation à des récepteurs. Ces derniers font office de passerelles entre les compteurs et le serveur. Les données collectées sont stockées dans une seule base de données SQL hébergée à l'université.

Cette plate-forme expérimentale permet d'obtenir des données de consommation qui sont représentatives des conditions réelles. La plate-forme est modulaire et peut intégrer d'autres compteurs à tout moment.

A ce jour, la plate-forme expérimentale est composée de:

- 5 compteurs d'eau officiels pour la facturation ;
- 7 nouveaux compteurs d'eau, intégrant les contributions du chapitre 2, pour étendre l'analyse et la collecte des données ;
- 5 récepteurs (Raspberry Pi 3 B+) pour renvoyer les trames sur le serveur ;
- 1 station météorologique et des capteurs environnementaux (développés sur un Arduino Uno) pour collecter des données de température, d'humidité, de pression et de luminosité ;
- 1 serveur SQL hébergé à l'université pour stocker les données décodées ;
- 1 système de surveillance pour la détection et l'analyse des fuites.

3.2.2 Aspects de communication

La réception des données transmises par les compteurs d'eau est effectuée à l'aide d'un module RF 868 MHz connecté sur un Raspberry Pi. La Fig. 3.2 montre le compteur d'eau, le module radio et le récepteur. Sur cette figure, le module radio est séparé du compteur d'eau, en fonctionnement

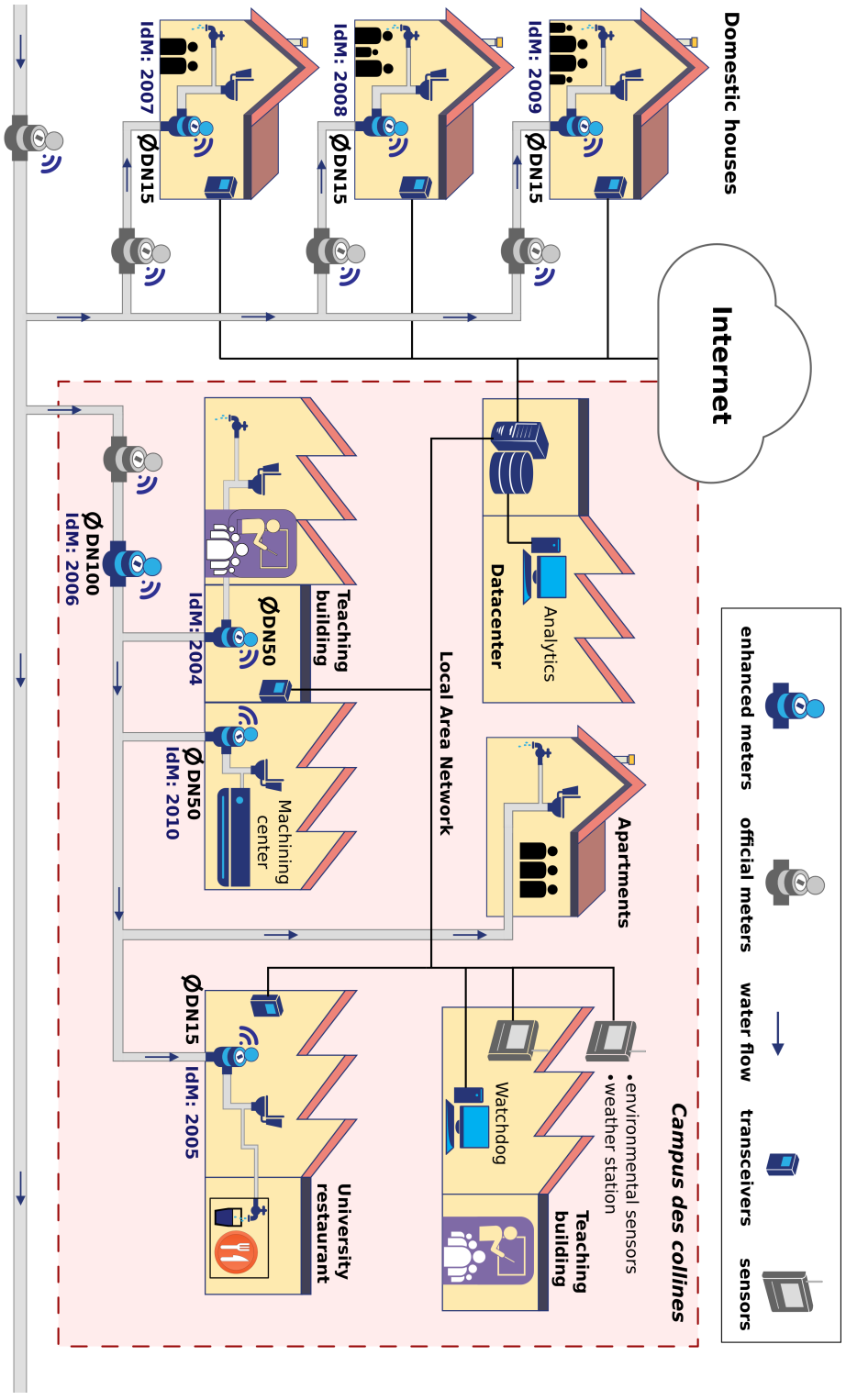


Figure 3.1 – Plate-forme expérimentale conçue pour collecter les données dans des environnements tertiaires, industriels et domestiques

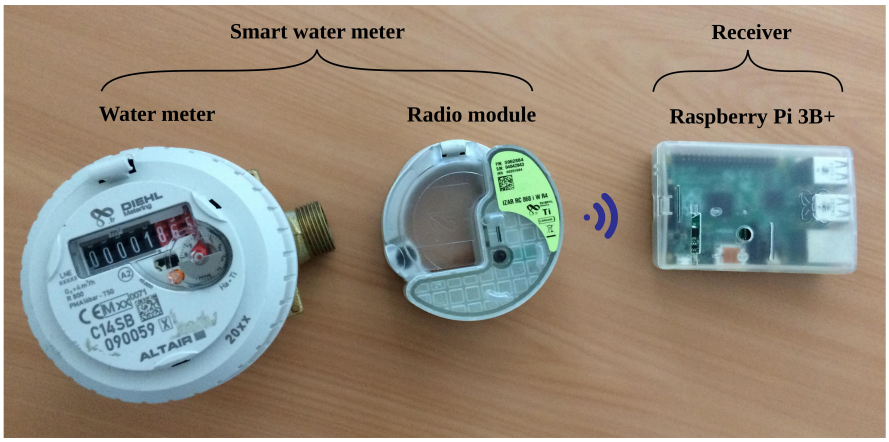


Figure 3.2 – Illustration d'un compteur d'eau mécanique, d'un module radio pour la transmission de données et d'un récepteur pour la réception des messages

normal il est clipsé sur le compteur d'eau mécanique. Dans un premier temps, des Raspberry Pi ont été choisis pour le prototypage car ils sont faciles et rapides à mettre en œuvre. Après une première étape de validation pour la réception des données, les Raspberry Pi seront remplacés par des systèmes à microcontrôleurs spécifiques intégrés dans les concentrateurs existants. Ces derniers sont conçus pour réceptionner les données sur de longues distances (plusieurs kilomètres).

Les trames provenant des compteurs installés sur le campus sont directement renvoyées par le récepteur sur le serveur web via le réseau informatique (*Local Area Network*). Concernant les compteurs situés à l'extérieur du campus (cas domestiques), les trames sont reçues en local, puis transmises sur le serveur via Internet. Une fois les données réceptionnées, elles sont décodées puis stockées pour pouvoir être analysées.

Personnaliser et optimiser la distribution de l'eau nécessitent plusieurs étapes. Celles-ci sont illustrées sur la Fig. 3.3. Les messages transmis par les compteurs sont reçus, déchiffrés et décodés avant d'être stockés dans la base de données. La caractérisation et la modélisation des consommations ne font pas l'objet de cette thèse. Le compteur amélioré, avec sa stratégie de transmission de données, permet d'envisager les fonctionnalités 3, 4, 5, 6 et 7 du diagramme de la Fig. 3.3, ce que ne permet pas un compteur actuellement commercialisé.

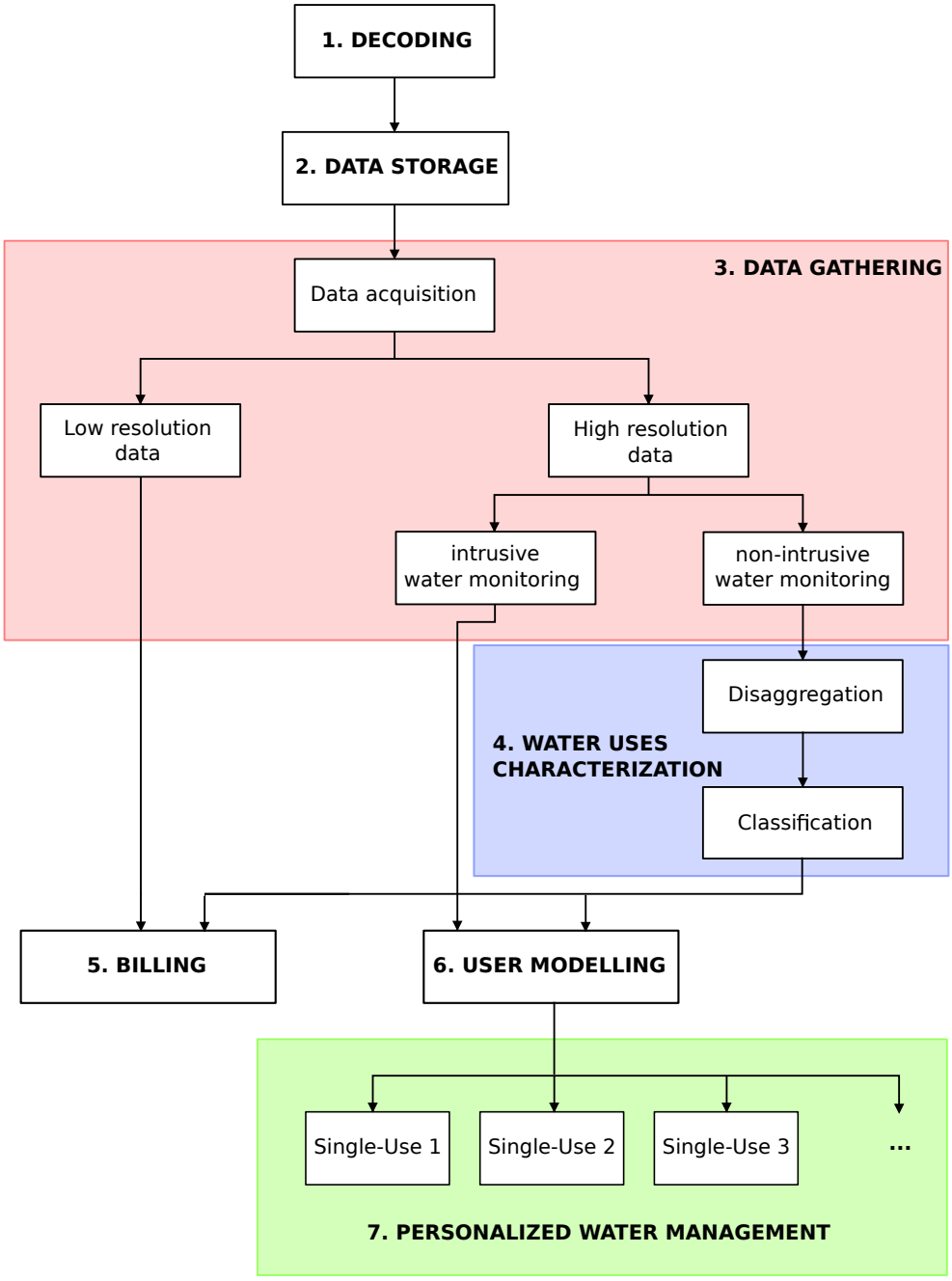


Figure 3.3 – Diagramme illustrant les différentes étapes de la réception des messages jusqu'à la gestion personnalisée de l'eau

3.3 Décodage des trames

Afin d'analyser les données de consommation, les trames reçues doivent être déchiffrées et décompressées. Le processus de décompression est illustré sur la Fig. 3.4 et applique l'opération inverse des étapes de compression définies dans le chapitre précédent. Le décodage est aussi important que le codage car il s'agit de la dernière étape dans la chaîne de communication.

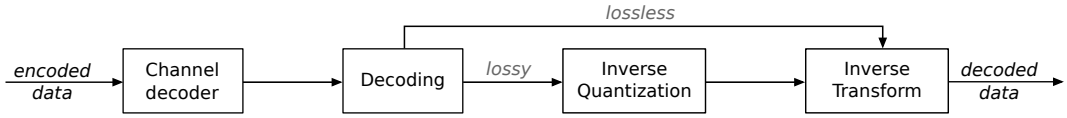


Figure 3.4 – Architecture générale de décompression

3.3.1 Suppression des redondances

Avant même de décompresser les données sur le serveur, la première étape consiste à filtrer les trames dans le récepteur. Le récepteur vérifie donc l'exactitude des trames reçues. Un contrôle de redondance cyclique (CRC) est effectué à l'aide des deux derniers octets de la trame. Cela permet de détecter les erreurs de transmission et s'assurer que les trames reçues soient valides. La Fig. 3.5 illustre la structure générale d'une trame vérifiée et validée.

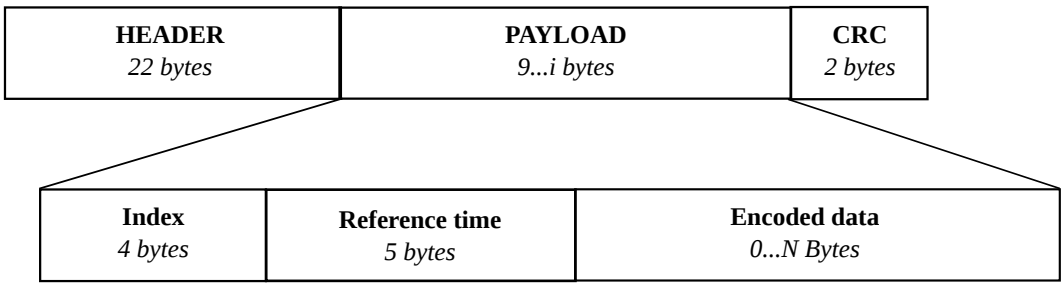


Figure 3.5 – Décomposition des trames réceptionnées pour le décodage de source

De plus, les trames reçues peuvent contenir plusieurs fois la même information due à la fenêtre glissante présentée dans le chapitre précédent. Les trames reçues dans le serveur sont donc filtrées en amont par le récepteur. Ce processus de filtrage a été mis en place pour supprimer les informations répétées, qui sinon augmenterait considérablement la quantité de données à stocker. Pour cela, deux informations sont nécessaires : le nombre de redondance (R_E) et le compteur de message indiquant le numéro de la trame. Ces deux champs sont encodés dans l'en-tête de la trame (*HEADER* sur la Fig. 3.5).

Un exemple d'application illustre le principe de fonctionnement du filtrage proposé sur la Fig. 3.6. Toutes les trames numérotées (n°122 à n°134) sont envoyées par le compteur. Supposons que certaines d'entre elles n'ont pas été correctement réceptionnées à cause de problèmes de transmission. Le récepteur filtre les trames reçues en utilisant le nombre de redondance (R_E = 6) et le numéro de trame. Dans ce cas, le récepteur retransmet une trame sur six au serveur.

Par ailleurs, les données brutes sont des instants définis à partir de l'horloge interne de chaque compteur. Même si toutes les horloges sont correctement réglées, elles peuvent dériver. Pour corriger la dérive des horloges, une synchronisation est nécessaire. La synchronisation n'a pas fait l'objet de cette thèse car elle peut être gérée de différentes manières [107], [108]. Celle-ci devra être effectuée en permanence sur le serveur pour profiter de la puissance de calcul mise à disposition.

Pour une transmission radio unidirectionnelle, nous proposons d'émettre quelques fois dans la journée (e.g., deux ou trois fois) le temps de réception t_{Rec} lorsqu'une trame est reçue par le récepteur (Fig. 3.7). Cette proposition permet de corriger les dérives d'horloges directement sur le serveur. Plusieurs méthodes pourront être utilisées par la suite pour calculer les différences de temps et corriger la dérive dans l'horodatage des données.

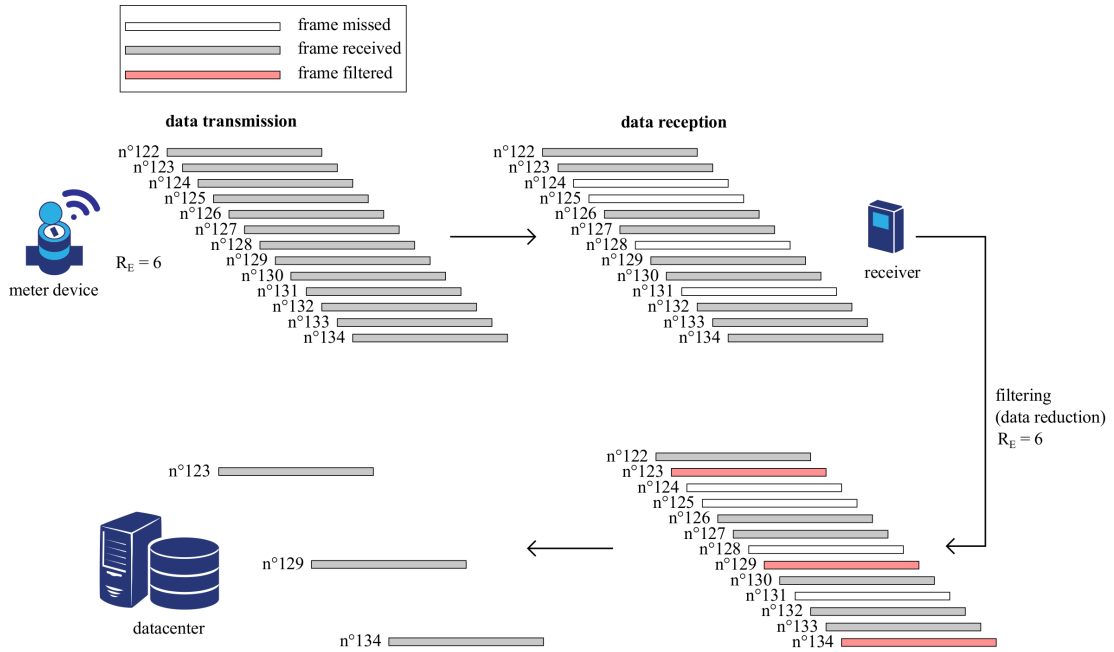


Figure 3.6 – Filtrage mis en place dans le récepteur pour réduire la quantité de données avant les processus de décodage et de stockage

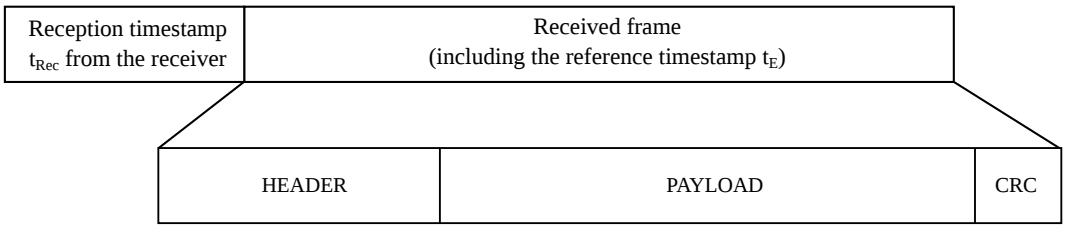


Figure 3.7 – Trame envoyée par le récepteur avec l'horodatage de réception pour synchroniser les dérives d'horloge

3.3.2 Extraction des données

A leur réception sur le serveur, les trames sont directement décodées. La longueur des trames varie entre 33 et 120 octets en fonction des données brutes compressées. L'index et l'horodatage de référence sont lus avant le décodage de la source.

Afin de décrire le processus de décompression, nous considérons la réception de la trame compressée f_E suivante : f_E = {*0xaf 0x63 0x90 0x76 0x87 0xb9 0x6d 0xeb 0x6e 0x5b 0x7a 0xbb 0x1a 0xc9 0x36 0xe5 0xb8 0x29 0x2e 0xd2 0xda 0x0*}.

La trame f_E est représentée au format binaire afin de décomposer les différents champs. La chaîne de bits doit être décodée conformément à la structure de l'algorithme RLBE. Le décodage de source est le processus inverse du codage RLBE. La Fig. 3.8 illustre la décomposition de chaque champs correspondant à la structure RLBE : S_{RLBE}= {H_1, R_1, D_1, H_2, R_2, D_2, ..., H_n, R_n, D_n}, avec n le nombre de plages. Chaque plage est décodée en lisant 5 bits fixe pour H_n, k_n bits pour R_n et α_n bits pour D_n. Les k_n bits correspondent aux redondances des données successives D_n. Ces redondances sont décodées avec la séquence de Fibonacci en ajoutant les termes de la séquence associée aux symboles "1". Ensuite, α_n et H_n sont convertis en décimal pour détecter chaque code binaire D_n. Le Tableau 3.1 illustre le processus de décodage complet relatif à la structure RLBE. En cas de retour d'eau, un délimiteur "00000" est utilisé. Ce délimiteur indique que les données qui suivent correspondent à des retours d'eau. Le décodage de ces données reste comparable au décodage des données positives. Dans l'exemple donné, f_E ne contient pas de retour d'eau.

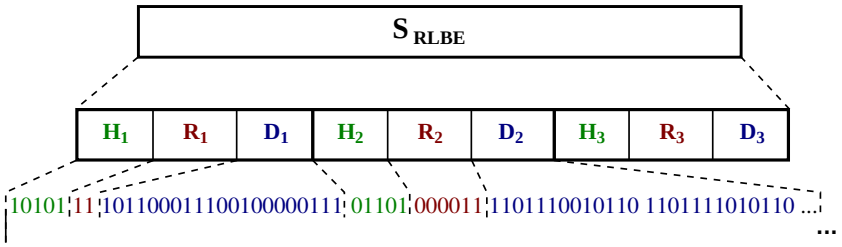


Figure 3.8 – Décomposition de la trame f_E selon la structure RLBE

Une fois que chaque plage est décodée, les codes binaires sont convertis pour obtenir les valeurs différentielles δ_i. Ces valeurs correspondent à la différence de temps entre deux événements horodatées. La valeur horodatée t_i est obtenue suite à l'opération inverse du codage différentiel. Chaque valeur horodatée est calculée comme suit, avec $i > 0$ et t_E l'horodatage de référence :

$$t_{i-1} = t_i - \delta_i. \tag{3.1}$$

La transformation inverse permet de reconstituer les données au cours du temps. Comme le montre la Fig. 3.9, cette transformation reconstruit parfaitement les données d'origine. Les

Tableau 3.1 – Décodage complet des données compressées selon la structure S_{RLBE}

Sequence n	1			2			3		
$\mathbf{S_{RLBE}}$	$\mathbf{B_1}$	$\mathbf{R_1}$	$\mathbf{D_1}$	$\mathbf{H_2}$	$\mathbf{R_2}$	$\mathbf{D_2}$	$\mathbf{H_3}$	$\mathbf{R_3}$	$\mathbf{D_3}$
RLBE decoding	10101	11	10110001110010000111	01101	000011	1101110010110 1101111010110 1101110010110 1101111010101 1101100011010 1100100100110 1101110010110 1110000010100	10010	11	101101001011011010
Values	L_{D_1}=21	1x	1x21 bits	L_{D_2}=13	8x	8x13 bits	L_{D_3}=18	1x	1x18 bits
Decimal conversion of binary codes δ_i	1456391			7062, 7126, 7062, 7125, 6938 6438, 7062, 7188			185050		

données encodées dans le module radio sont parfaitement reconstituées, selon la résolution initialement définie (1 ms pour chaque événement horodatée).

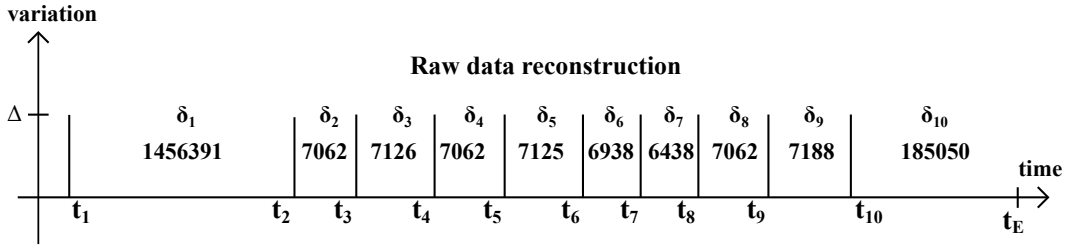


Figure 3.9 – Reconstruction des événements horodatés à partir des données compressées

3.4 Architecture proposée pour le stockage des données

Le stockage des données décompressées est une étape importante pour le traitement et l'analyse de la consommation. Les données collectées peuvent rapidement représenter une quantité importante à stocker. Par ailleurs, cette quantité a tendance à augmenter en raison de l'émergence de données complémentaires (e.g., la température, la pression). Il est essentiel de concevoir une architecture de stockage efficace pour optimiser l'analyse et la gestion des réseaux de distribution. Les données décodées peuvent être stockées de plusieurs manières, mais la stratégie de stockage doit prendre en compte le temps des traitements et les capacités de stockage.

Une architecture de stockage a été mise en place pour évaluer et optimiser le volume de données généré. Les données sont stockées dans la base de données SQL hébergée sur le serveur de l'université. L'architecture proposée est organisée en cinq tables différentes pour :

- regrouper l'ensemble des trames filtrées par les récepteurs, y compris les temps d'émission et les temps de réception pour une éventuelle synchronisation des horloges ;
- différencier facilement la provenance des trames reçues à l'aide d'un identifiant (IdM) pour chaque compteur ;
- organiser les données décodées pour reconstruire l'historique de consommation, analyser et détecter les anomalies ;
- définir les droits d'accès pour la gestion de la plate-forme expérimentale ;
- décrire les événements détectés par les utilisateurs (fuites, interruptions de service, périodes de maintenance, etc.).

L'architecture de la base de données est présente en annexe A.4. Les données décodées précédemment sont, pour l'instant, simplement stockées dans la table *myRawData*. A long terme, il n'est pas concevable de stocker les données sans les compresser. Pour cette raison, nous avons évalué l'espace de stockage nécessaire en compressant les données brutes décodées dans des fichiers zip (algorithme de compression *DEFLATE* [94]).

Des données réelles ont été collectées et compressées pour une consommation journalière

Tableau 3.2 – Comparaison de la quantité de données stockées pour 24h de consommation pour trois environnements domestiques différents

	Domestic cases		
	with 2 people	with 3 people	with 4 people
Daily volume (liters)	290	380	609
Number of received frames by the receiver (per day)	341	355	368
Amount of data received by the receiver and per day (kb)	11.8	13.0	15.1
Amount of data stored in zip files per day (bytes)	711	1005	1200
Extrapolation of data stored for 15 years (Mb)	3.89	5.51	6.57

typique, dans trois environnements domestiques différents. Ces environnements représentent trois foyers composés de deux, trois et quatre personnes avec des usages et des consommations d'eau différents. Le Tableau 3.2 montre que la longueur des données compressées dépend fortement du volume d'eau consommé. Nous pouvons constater que la consommation moyenne fluctue entre 120 et 150 litres par jour et par personne, comme indiqué dans [109]. Extrapolée à la durée de vie moyenne d'un compteur (15 ans), la taille des données compressées représente quelques Mega-octets (Tableau 3.2).

Chaque fichier compressé comprend les données d'une journée complète avec l'index absolu au début de la journée, la variation Δ de la grandeur mesurée, l'horodatage du premier événement décodé, et les valeurs différentielles décodées en millisecondes. L'horodatage du premier événement décodé sert de référence pour toutes les valeurs différentielles. Ces valeurs correspondent à la différence de temps entre deux événements. L'ensemble des valeurs différentielles est défini comme une *série temporelle* représentant l'évolution de la consommation au cours du temps. Un jeu de données est montré en annexe A.5. Ce jeu de données correspond à une journée complète de consommation dans un environnement domestique de deux personnes.

Un autre test expérimental a été réalisé à partir des trois environnements présentées dans le Tableau 2.14. Les mesures ont été effectuées pour une consommation journalière moyennée sur 183 jours. Le Tableau 3.3 illustre la quantité de données stockées dans des conditions réelles de fonctionnement. Ces données prennent en compte les nuits, les week-ends, les vacances scolaires et les périodes de non-consommation. Ces résultats reflètent des conditions de consommation réelles.

Le stockage et le traitement des données sont réalisés concrètement en combinant différentes technologies avec plusieurs langages de programmation tels que Matlab, Python et PHP. La Fig. 3.10 illustre la centralisation des données sur le serveur. À l'avenir, les conteneurs web, les

[1]Stored in the Table "myEncodedRawFrames" as mentioned in Fig. A.5
[2]Stored in the Table "myRawData" as mentioned in Fig. A.5

Tableau 3.3 – Stockage des données décompressées pour trois environnements différents sur une journée moyenne de consommation calculée à partir des six mois de données présentées dans le Tableau 2.14

	Domestic case with 2 people	University restaurant	Whole IUTM (main meter)
Identifier Meter (IdM)	2007	2005	2006
Average volume of water consumed per day (liters)	268	739	29200
Average number of frame received by the receiver (per day)	328	375	513
Average amount of data received by the receiver and per day (kb)[1]	9.1	12.9	36.0
Average amount of data stored in zip files per day (bytes)	425	1101	4381
Average processing time for decoding (per frame in ms)	1.8	8.8	124.1
Average processing time for compressing and storing all data (in ms)	42	118	458
Extrapolation of data stored for 15 years (Mb)[2]	2.33	6.03	24.0

machines virtuelles (VM) ou Docker pourront être utilisés pour réduire les temps de traitements et optimiser l'utilisation des ressources [110]. La compression de données sur le serveur doit pouvoir être effectuée directement dans la base de données. Dans le futur, les données compressées pourront être stockées efficacement dans une base de données spécifique aux séries temporelles. Connu sous la dénomination de *Time Series DataBase* (TSDB), ce type de base est très répandu dans l'Internet des Objets (IoT). Une TSDB permet de stocker une grande quantité de données compressées avec des fichiers objets appelés BLOB (en anglais *Binary Large OBjects*) [111]. Bien que les BLOB servent à stocker du code source binaire, ils peuvent également contenir des fichiers tels que des images ou des vidéos. Les fichiers zip pourront donc être instanciés en tant que BLOB.

3.5 Analyse des données à travers une étude de cas

La courbe de charge et le profil de consommation introduits dans le chapitre 1 ont été calculés à partir des données décodées et stockées sur le serveur. Ces outils d'analyse sont illustrés sur la Fig. 3.11, et mettent en évidence la richesse des informations. Chaque fluctuation dans le profil fournit une information utile sur la consommation et sur le flux de l'eau dans le réseau de distribution.

Pour effectuer nos premières analyses, nous nous sommes intéressé à la consommation du restaurant universitaire. Les courbes de charge et les profils de consommation d'eau obtenus

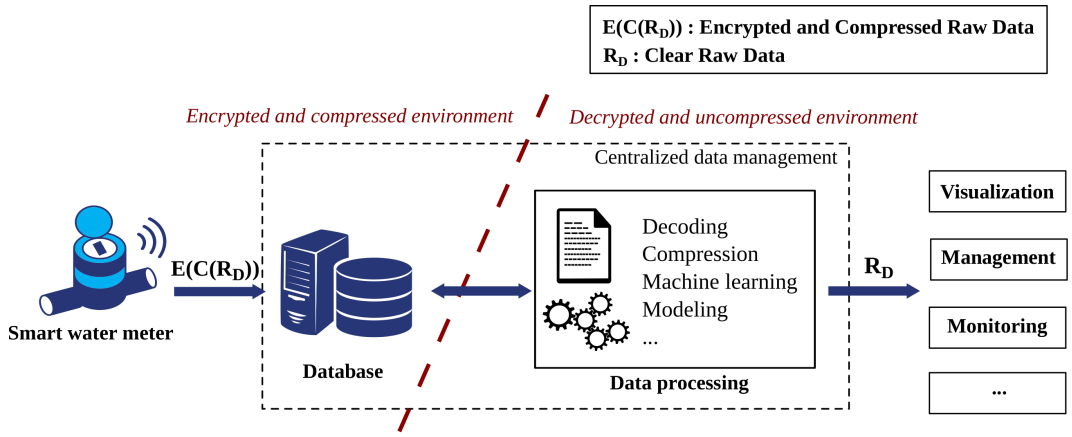


Figure 3.10 – Centralisation des données sur le serveur pour l'analyse de la consommation

sont utilisés pour comparer chaque jour de consommation. La Fig. 3.12 illustre six jours successifs, week-end inclus. Pour pouvoir comparer chaque jour ouvré, les données ont été ré-échantillonnées avec une résolution à la minute. Cela conduit à cinq courbes journalières de 1440 points (Fig. 3.14). Cette résolution (1 min) permet de détecter facilement les grosses fuites sans effectuer de traitements complémentaires (Fig. 3.15). Cette figure met en évidence la fuite présente entre mercredi 15h00 et jeudi 9h00.

Afin de détecter automatiquement les fuites et les anomalies dans le réseau de distribution, un système de surveillance a été mis en place. Ce système analyse en permanence les données décodées sur le serveur. Des scripts d'analyse et de détection ont été implémentés en Python et en Matlab. Dès qu'une anomalie est détectée, un message d'alarme est envoyé à l'utilisateur (e.g., le technicien de maintenance à l'université ou encore le consommateur final). Grâce à ce système de surveillance, une rupture de canalisation a été détectée automatiquement à l'université. La fuite est clairement visible sur la Fig. 3.13. Le message d'alarme envoyé au technicien de maintenance a permis de réparer la fuite deux heures plus tard.

Une étude complémentaire permettrait de caractériser et modéliser le comportement dans le réseau de distribution afin de prédire les éventuelles anomalies. La caractérisation des utilisations de l'eau peut être réalisée en utilisant des algorithmes d'apprentissage automatique à partir des données décodées et stockées. Ces derniers ont pour objectif de décrire et de prédire l'utilisation de l'eau dans le réseau de distribution. Plusieurs études suggèrent l'utilisation de techniques d'apprentissage tels que les réseaux de neurones artificiels ou encore les modèles de Markov [112], [113]. Les tendances des recherches futures visent à mettre en place des techniques d'apprentissage automatique pour analyser les données collectées tout en s'adaptant aux usages et aux usagers. Ces techniques permettront de personnaliser la gestion de l'eau et de définir des modes de consommation précis et fiables, utiles pour les services des eaux. L'utilisation de techniques d'apprentissage permet d'étudier plus en détail le comportement des compteurs dans le temps comme illustré sur la Fig. 3.16 où le profil de consommation est identifié et divisé comme un sous-ensemble d'événements. Les compteurs d'eau communicants

pourront être utilisés dans les technologies de l'*Active Assisted Living* (AAL) [114] afin d'offrir davantage de sécurité. Associée à l'intégration d'autres capteurs, cette nouvelle stratégie de collecte fournit de nouvelles solutions et ouvre des perspectives en terme d'utilisation.

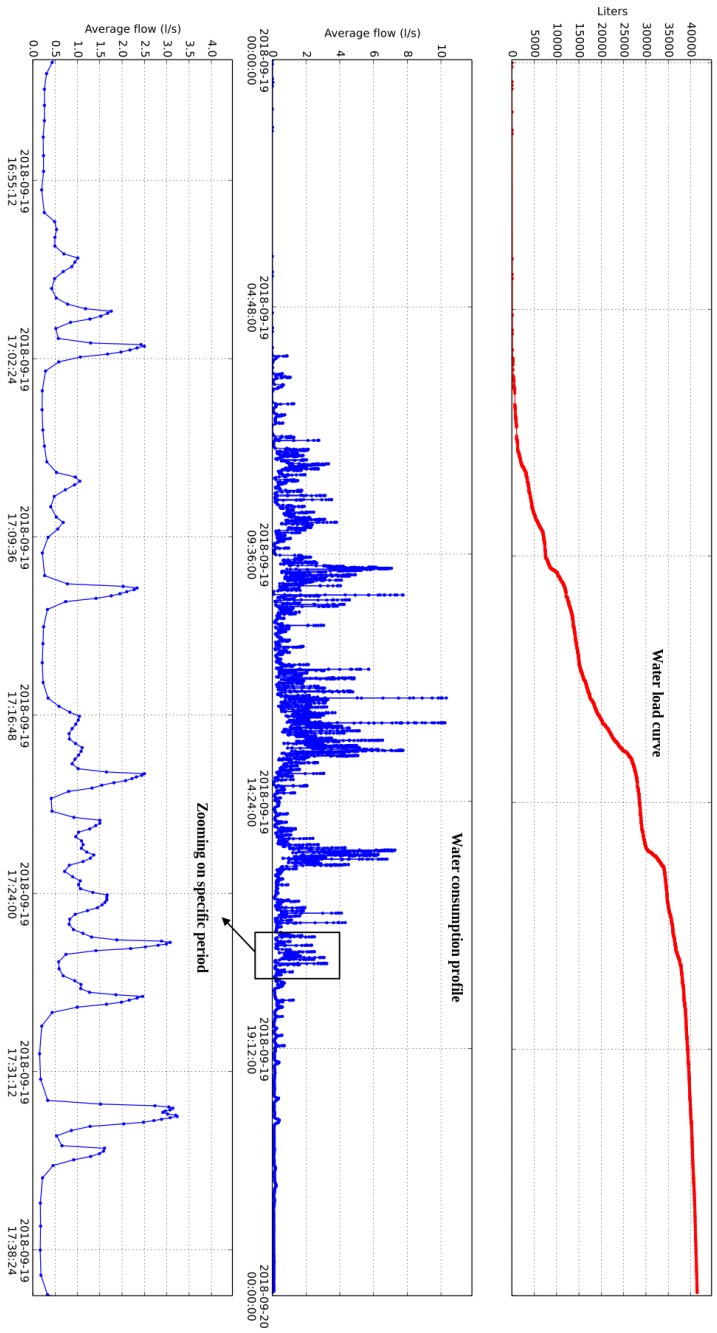


Figure 3.11 – Courbe de charge et profil de consommation illustrant la richesse des informations collectées pour une journée typique de consommation à l'université (DN100)

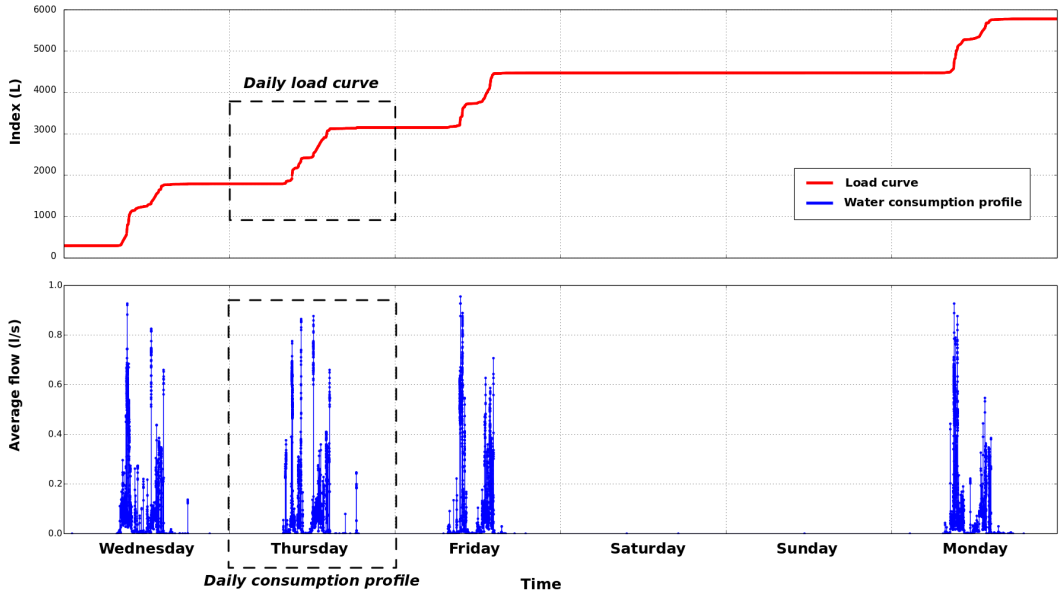


Figure 3.12 – Courbe de charge et profil de consommation illustrant la consommation sur six jours successifs au sein du restaurant universitaire à l'IUT de Mulhouse

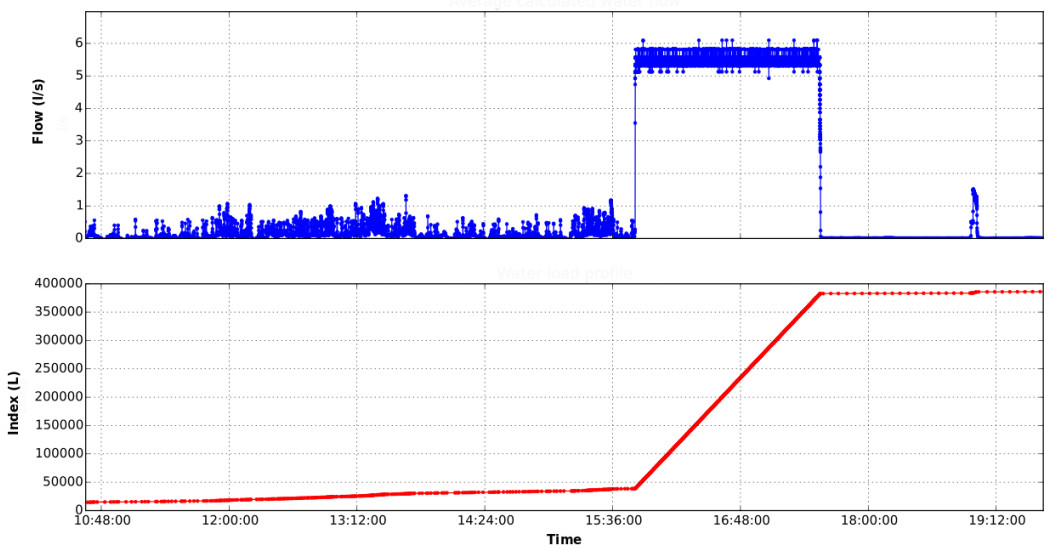


Figure 3.13 – Rupture de canalisation détectée par le système de surveillance le 19 février 2018 sur le compteur principal de l'université

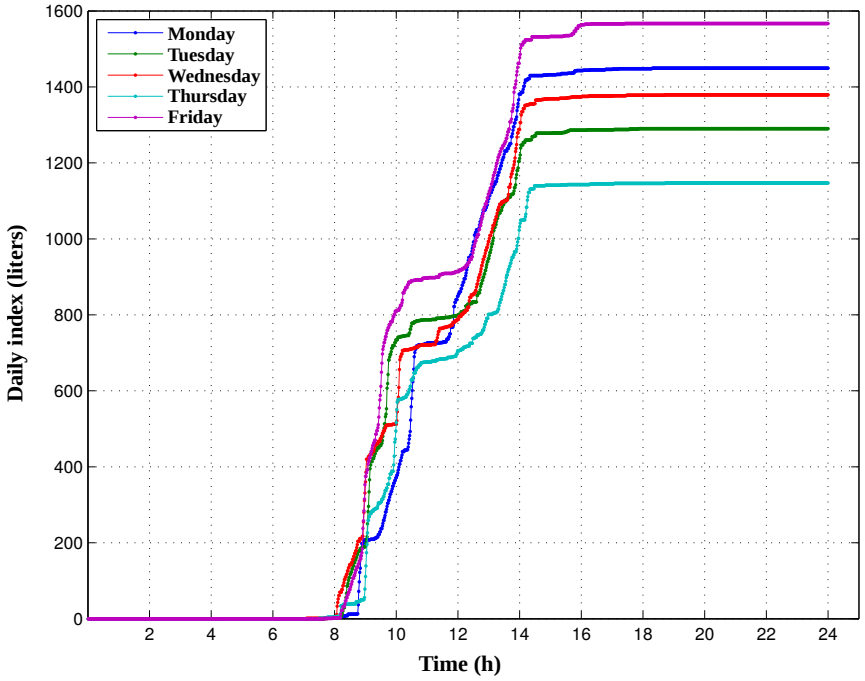


Figure 3.14 – Comparaison des courbes de charge journalières pour une semaine typique de consommation d'eau au restaurant universitaire

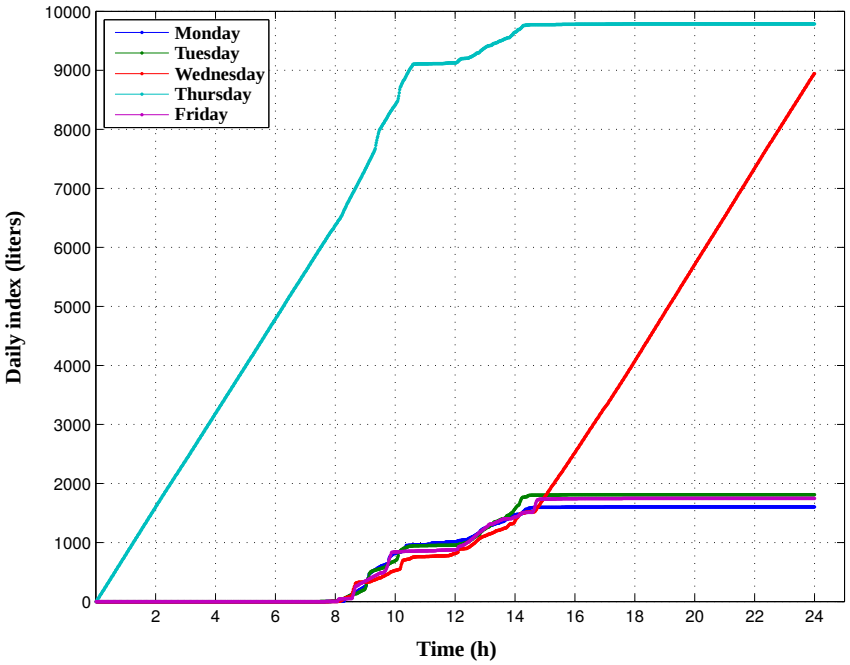


Figure 3.15 – Détection de fuite au restaurant de l'université à partir des courbes de charge échantillonnées entre Mercredi 14h30 et Jeudi 9h00

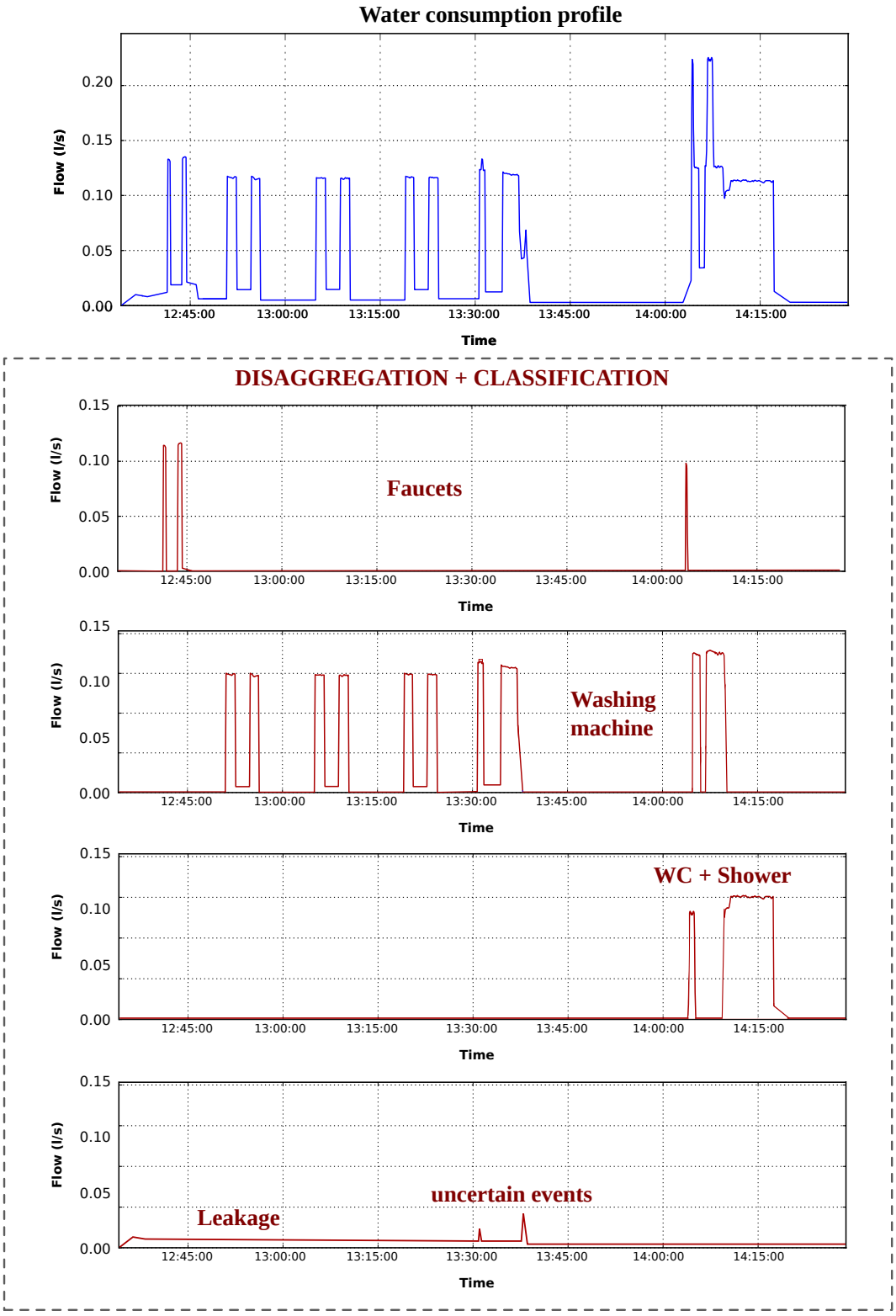


Figure 3.16 – Caractérisation des utilisations de l'eau avec un exemple de désagrégation et de classification

3.6 Conclusion

Dans ce chapitre, une plate-forme expérimentale a été mise en place dans des cas réels de fonctionnement pour valider le fait que :

- les prototypes soient capables de fonctionner sur une longue période ;
- les communications soient fiables entre les compteurs d'eau et le serveur ;
- les données soient correctement décodées et stockées dans la base de données ;
- les données brutes soient fidèlement reconstruites ;
- le système de surveillance soit capable de détecter les anomalies et les fuites pour en informer automatiquement l'utilisateur final ;
- la collecte de données offre une analyse précise de la consommation.

Cette plate-forme expérimentale a permis de collecter les données de comptage dans le but de les analyser. Dans ce chapitre, le processus de décodage est complètement décrit, de la réception des données à la reconstruction complète du profil de consommation. Une architecture de stockage a été proposée afin de limiter le volume de données générées, réduire les temps de calcul dans le serveur et permettre de faire des analyses. Les résultats ont montré qu'il faut prévoir en moyenne environ 11 Mo pour stocker 15 ans d'enregistrement d'un seul compteur, avec une résolution d'une milliseconde pour chaque mesure horodatée. La taille des données à stocker varie en fonction du volume de données généré. Par ailleurs, un système de surveillance a été mis en place pour détecter automatiquement les fuites et pour effectuer de premières analyses de consommation. A ce jour, la plate-forme qui utilise les technologies proposées génère les données au fur et à mesure des consommations et ceci de manière autonome. Un historique des données est ainsi constitué. L'ensemble de ces données permettra de caractériser et de modéliser les usagers et les usages en terme de consommation d'eau.

La stratégie complète et proposée ne s'applique pas uniquement au comptage de l'eau c'est pourquoi dans le prochain chapitre nous nous intéresserons à la généralisation de la collecte des données. L'objectif est de s'affranchir des contraintes de transmission et de proposer un encodage et une compression de données appropriée à tous types de mesure.

4 Généralisation de la collecte des données

Nous avons proposé et validé au cours des chapitres précédents une stratégie complète de collecte de données ; de l'acquisition jusqu'à la compression des données de mesures en vue d'une transmission peu coûteuse en énergie à des fins d'exploitation. Bien que cette stratégie ait été développée dans le cadre des compteurs d'eau, elle ne se limite pas à ce seul fluide. En effet, nous pouvons aisément remplacer un volume d'eau par un volume de gaz, une consommation électrique, une température ou toute autre grandeur physique. Les contraintes liées à l'exploitation de ces grandeurs peuvent varier en fonction des souhaits de l'utilisateur.

Afin de s'adapter à de nouvelles problématiques, nous proposons dans ce chapitre de généraliser la collecte des données en introduisant plus de souplesse dans le choix des algorithmes de compression, de la résolution, de la quantité de données à transmettre et de la technologie utilisée pour envoyer les données.

Une structure universelle de données sera proposée dans la section 4.1. Celle-ci sera composée d'un bloc de données de longueur variable, facilement adaptable en fonction des besoins. Une étude de cas sera illustrée pour valider la faisabilité de la stratégie pour d'autres dispositifs. La section 4.2 propose une méthode de compression dynamique sans perte qui permet de tenir compte des redondances dans les données d'origine. Cette méthode permettra de sélectionner l'algorithme de compression optimal et d'adapter l'encodage des données. Les performances de la compression dynamique seront comparées à la compression sans perte RLBE dans la section 4.3. Enfin, les avantages et les inconvénients de la compression dynamique seront discutés.

4.1 Proposition d'une structure universelle de données

La stratégie proposée jusqu'ici permet de collecter le profil de consommation d'eau avec une résolution temporelle maximale. La qualité des données collectées sur le serveur est identique à celle fournie par le capteur ; le processus de collecte et de transmission n'altère pas l'information. Cette stratégie peut être utilisée pour tout type de mesures (e.g., électricité, gaz, pression). Pour collecter diverses autres grandeurs, nous proposons de généraliser la collecte de données avec une structure universelle. Cette généralisation permet d'encoder efficacement les données en fonction de la résolution souhaitée, du mode de transmission (liaison filaire ou sans fils), de la quantité de données à transmettre, etc. Les opérations d'encodage et de transmission seront traitées de façon indépendante, ce qui permet de s'affranchir des contraintes de transmission.

La fenêtre glissante proposée dans le chapitre 2 a permis de fiabiliser la transmission en répétant les informations plusieurs fois, mais pour assurer la réception des informations, d'autres méthodes existent, telles que l'utilisation d'un code correcteur Reed-Solomon pour essayer de reconstruire les données erronées pendant la transmission, ou bien une communication bidirectionnelle avec accusé de réception pour valider la transmission des messages. Or, pour généraliser la collecte de données, l'ajout de redondances est une contrainte de transmission qui doit être effectuée indépendamment de l'encodage. De plus, il n'est pas toujours nécessaire de répéter plusieurs fois la même information car cela dépend du bilan de liaison. C'est pourquoi, les informations à transmettre sont mises en forme dans un bloc de données composé d'un certain nombre de paramètre commun à toute grandeur physique (la nature des données d'origine, l'algorithme de compression, la résolution souhaitée, etc.). Ces différents paramètres seront décrits dans la section suivante. Le Tableau 4.1 illustre les différences entre la collecte de données développée dans le chapitre 2 pour le comptage de l'eau et la collecte de données généralisée à tout type de mesures.

Tableau 4.1 – Comparaison des paramètres utilisés pour la collecte de données développée dans le chapitre 2 pour le comptage de l'eau et la collecte de données généralisée à tout type de mesures

	Collecte de données proposée dans le chapitre 2	Collecte de données généralisée
Grandeur mesurée	volume d'eau	au choix
Variation Δ	1 litre	au choix
Résolution temporelle souhaitée	résolution 1 ms	au choix
Algorithme de compression	un seul algorithme (RLBE)	au choix
Type de compression	sans perte	avec ou sans perte selon l'algorithme de compression
Nature des données brutes	différences de temps (δ_i et δ'_i)	
Mode de transmission	RF avec fenêtre glissante ($R_E = 6$)	libre

4.1.1 Définition et caractérisation

Afin de décorréler l'encodage de la transmission, un bloc universel de données est proposé afin de regrouper toutes les informations requises pour reconstruire un profil relatif à une grandeur physique. L'encodage d'un profil concerne tout type de capteurs, avec des événements qui peuvent être générés de différentes manières (e.g., impulsions, valeurs numériques). Les événements sont encodés de façon relative à un instant de référence t_R (équivalent à t_E dans le le chapitre 2 tout en s'affranchissant de la transmission). L'instant de référence t_R correspond à une référence définie à la création du bloc de données avant la détection des événements. Comme le montre la Fig. 4.1, les événements détectés sont horodatés et un codage différentiel est appliqué à partir de t_R.

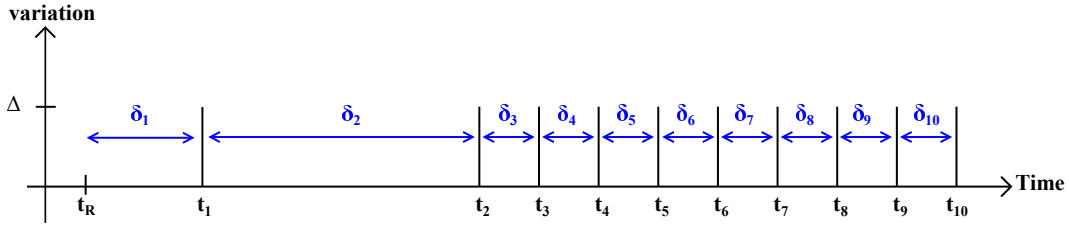


Figure 4.1 – Caractérisation de l'horodatage de référence t_R et codage différentiel des événements horodatés

La longueur des données encodées est définie par la première limite atteinte entre la longueur maximale L_{max} du bloc de données (en octets) et la fenêtre de temps maximale T_{max}. T_{max} correspond à l'intervalle de temps maximum entre deux temps de référence consécutifs t_{R_i} et $t_{R_{i+1}}$. Les paramètres L_{max} et T_{max} sont utilisés pour encoder les blocs de données individuellement. Pour illustrer le rôle de ces paramètres, nous avons considéré deux cas différents :

- La longueur du bloc n'excède pas la longueur maximale tolérée L_{max}. La fin du bloc est déterminée par le paramètre T_{max} afin de garantir une collecte fréquente de données (Fig. 4.2). T_{max} est considéré comme le temps de référence $t_{R_{i+1}}$ pour le prochain bloc.

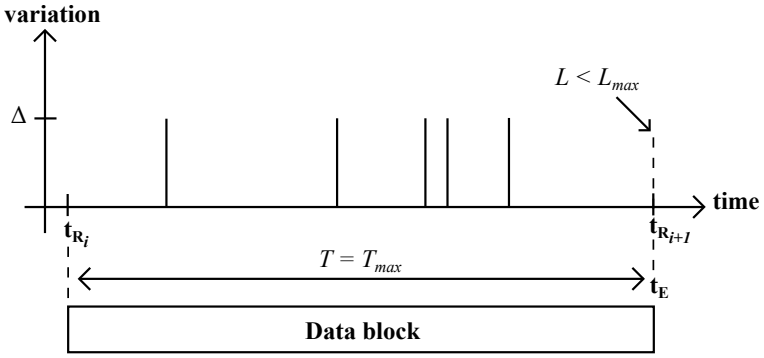


Figure 4.2 – Exemple d'un bloc de données limité par la fenêtre d'enregistrement maximale T_{max}

- La longueur du bloc est limitée par la longueur maximale tolérée L_{max}. Dans ce cas, la fenêtre d'enregistrement T est par conséquent plus petite que T_{max} (Fig. 4.3).

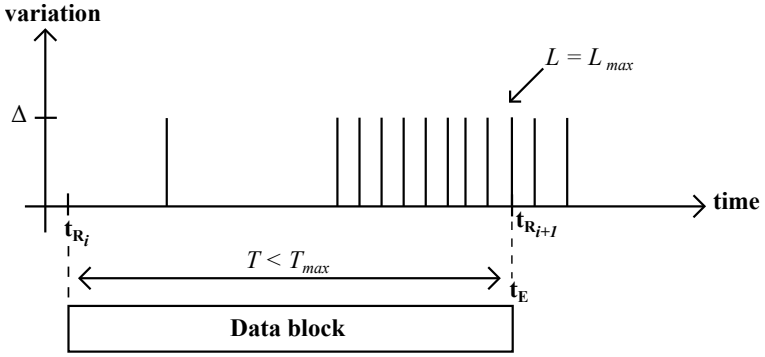


Figure 4.3 – Exemple d'un bloc de données redimensionné pour respecter la longueur maximale tolérée L_{max}

Ces deux paramètres permettent d'encoder les blocs en fonction de la quantité de données initiales (Fig. 4.4). Le nombre d'encodage va dépendre du volume de données à compresser, de la fenêtre d'enregistrement T_{max} et de la longueur maximale L_{max}.

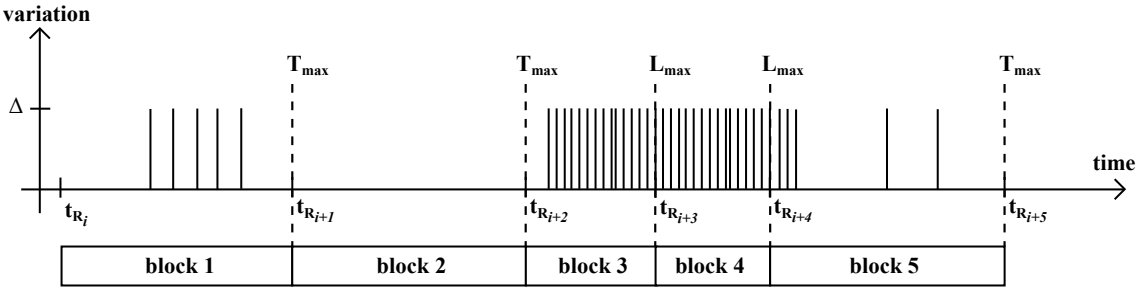


Figure 4.4 – Concaténation de plusieurs blocs en fonction des paramètres T_{max} et L_{max}

Afin d'encoder toute grandeur physique, le bloc de données proposé est toujours construit avec les informations suivantes :

- la longueur du bloc de données ;
- le descripteur indiquant la structure du bloc ;
- la valeur absolue de la grandeur mesurée à l'instant de référence t_R ;
- l'instant de référence t_R ;
- les données brutes compressées.

Le bloc de données proposé est illustré sur la Fig. 4.5. Les quatre premiers champs (longueur, descripteur, valeur absolue de la grandeur mesurée, instant de référence) sont chacun encodé avec un code universel *Variable Length Quantity* (VLQ), utilisé dans le format de fichier MIDI [115]. Ce code est utilisé pour offrir le maximum de flexibilité en terme d'utilisation. La structure générale d'un VLQ est définie dans le Tableau 4.2. Un VLQ utilise un octet où le bit de poids fort A est réservé pour indiquer si un autre octet de ce même code suit. Si A est égal à 0, alors il

Data block				
Length	Descriptor pointer	Start value	Reference time t_R	compressed raw data
1...n bytes	*1...m bytes*	*1...x bytes*	*3...y bytes*	*0...z bytes*

Figure 4.5 – Structure générale du bloc de données universel

s'agit du dernier octet du code VLQ. En revanche, si A est à 1, le prochain octet correspond à l'extension du code VLQ. B_n est un code de 7 bits [0x00, 0x7F] et n est la position de l'octet dans le code VLQ, avec B_0 le moins significatif.

Tableau 4.2 – Structure générale du codage VLQ

VLQ byte							
7	6	5	4	3	2	1	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
A	B_n						

Le champ *length* est totalement variable en fonction de la longueur des données compressées. Lorsqu'il n'y a pas de données détectées entre t_{R_i} et $t_{R_{i+1}}$, le champ relatif aux données brutes est vide. Comme le montre la Fig. 4.5, le bloc de données proposé ne peut être encodé en dessous de 6 octets.

Le champ *descriptor pointer* définit la structure du bloc de données. Il s'agit d'un pointeur dans une table où chaque valeur pointe sur un ensemble de paramètres. Le Tableau 4.3 montre des exemples descripteur, composé des paramètres suivants :

- la variation Δ détectée par le capteur ;
- la résolution r_t de tous les événements horodatés dans le bloc ;
- la résolution du temps de référence t_R ;
- l'algorithme de compression utilisé.

Le nombre de paramètre est variable et peut être différent dans chaque ensemble. Si un nouveau paramètre est ajouté dans la table, la valeur attribuée à *descriptor pointer* sera le plus petit entier non utilisé.

Le champ *start value* indique la valeur absolue mesurée par le capteur à l'instant t_R (e.g., index pour la consommation d'eau). Comme illustré par le Tableau 4.4, le bit de poids fort A est utilisé pour adapter l'encodage en fonction de la valeur absolue. Par exemple, si la valeur absolue est égale à 99546, celle-ci nécessite 3 octets.

Tableau 4.3 – Exemple d'un tableau de descripteur illustrant des ensembles de paramètres utilisés pour différencier plusieurs grandeur physique mesurée

Descriptor pointer	Variation	Timestamp resolution	Reference time resolution	Compression algorithm	...
1	1 litre	15.625 ms			...
2	10 litres	1 ms	1 ms	RLBE	...
3	0.1 °C	1 s	1 min	Binary	...
4	0.1 litre	1 ms	1 ms	LZW	...
5	30 mbars	100 ms	1 s	Huffman	...
...					

Tableau 4.4 – Limites du code VLQ selon le nombre d'octets utilisés

MS Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	LS Bit 0	No. bytes	No. combination
A	B_0							1	127
A	B_1							2	16 383
A	B_2							3	2 097 151
A	B_3							4	268 435 455
...									
A	B_i							$i+1$	∞

Le champ *reference time* t_R est la référence de temps absolue nécessaire pour reconstruire toutes les différences de temps δ_i et δ'_i encodées dans le champ *compressed raw data*. Cet horodatage de référence est encodé en fonction de la résolution définit dans le tableau de descripteur. La résolution de l'horodatage de référence t_R ne doit pas être confondue à celle réservée pour l'horodatage de chaque événement (r_t). Nous avons défini une résolution minimale de 1 heure, ce qui conduit à une longueur minimale de 3 octets (e.g., une résolution d'une heure pour 20 ans est égale à 175 200 combinaisons). L'encodage de ce champ est identique aux trois précédents : *length*, *descriptor pointer* et *start value*.

4.1.2 Étude de cas

Pour une meilleure compréhension, deux exemples concrets sont illustrés ci-après, pour la mesure de la consommation d'eau et pour la mesure de la pression dans la canalisation. On peut constater que le principe de fonctionnement reste similaire et que le bloc de données décrit dans la section précédente permet d'englober les informations nécessaires pour reconstruire le profil de consommation relatif à la grandeur mesurée.

Exemple 1 : Mesure de la consommation d'eau
Nous considérons les paramètres suivants :

- variation Δ détectée par le capteur : 1 litre ;
- résolution r_t des événements horodatés : 1 milliseconde ;
- résolution du temps de référence t_R : 1 seconde ;
- valeur absolue initiale à t_R : 25560 litres ;
- horodatage de référence t_R : 2018-09-19 10h26min45s ;
- différence de temps δ_i relative à une variation positive : 40000, 4120, 1003 ms;
- différence de temps δ'_i relative à une variation négative : aucun.

Dans ce cas, la valeur absolue initiale (ici il s'agit de l'index du compteur d'eau) est de 25560 litres à 10h26min45s (t_R). Le premier événement détecté est apparu 40000 millisecondes après t_R. La valeur absolue s'incrémente de 1 litre à 10h27min25s000ms. Un deuxième événement est apparu 4120 millisecondes après le premier. La valeur absolue devient 25562 à 10h27min29s120ms. Enfin, le dernier événement est apparu 1003 millisecondes après le deuxième. La valeur absolue s'incrémente de 1 litre à 10h27min30s123ms et devient 25563 litres.

Exemple 2 : Mesure de la pression dans la canalisation
Nous considérons les paramètres suivants :

- variation Δ détectée par le capteur : 100 mbars ;
- résolution r_t des événements horodatés : 1 seconde ;
- résolution du temps de référence t_R : 1 seconde ;
- valeur absolue initiale à t_R : 3500 mbars ;
- horodatage de référence t_R : 2018-09-19 10h40min33s ;
- différence de temps δ_i relative à une variation positive : 633, 1800 seconds;
- différence de temps δ'_i relative à une variation négative : 1533 seconds.

Dans ce cas, la valeur absolue initiale (ici il s'agit de la pression mesurée dans la canalisation) est de 3500 mbars à 10h40min33s (t_R). Le premier événement détecté correspond à une variation Δ positive, avec une différence de temps δ_i de 633 secondes par rapport à t_R. La valeur absolue augmente donc de 100 mbars à 10h51min06s. Le deuxième événement correspond à une variation Δ négative. La valeur absolue diminue de 100 mbars à 11h06min06s et est égale à 3500 mbars. Enfin, la dernière variation est positive avec une différence de temps de 1800 secondes par rapport au premier événement. La valeur absolue devient 3600 mbars à 11h21min06s.

Dans ces deux exemples, le bloc de données n'est pas encodé de la même manière et la longueur du bloc dépend des événements détectés. La nature des données détermine toujours la quantité de données à compresser car le nombre d'événements peut impacter la redondance d'information dans les données d'origine. Par conséquent, il y a des cas où il n'est pas toujours souhaitable de compresser les données avec le même algorithme. C'est pourquoi nous nous intéressons à la compression dynamique.

4.2 Vers une compression de données dynamique sans perte

Il a été montré dans le chapitre 2 que les performances de compression dépendent en partie de la nature des données originales. La Fig. 2.14 du chapitre 2 montre le taux d'utilisation des algorithmes de compression offrant les meilleures performances en 24h de consommation dans cinq cas différents. On peut voir que l'algorithme de compression RLBE est en moyenne le plus adapté à la compression des données relatives au comptage de l'eau. En revanche, dans des cas particuliers (répétitions successives de valeurs identiques ou des cycles récurrents), d'autres algorithmes peuvent être plus efficaces. Dans ces cas-là, la compression de données dynamique sans perte prend tout son sens.

Une compression de données dynamique sans perte consiste à choisir l'algorithme adéquat en fonction des redondances dans les données originales. L'objectif est d'adapter la manière de compresser pour optimiser les performances de compression, limiter le nombre de transmissions et réduire la consommation d'énergie.

Nous proposons une nouvelle stratégie de compression dynamique, basée sur la méthode de compression sans perte de la section 2.3. Les étapes de calculs et de compression sont illustrées sur la Fig. 4.6. Cette stratégie se concentre uniquement sur le codage de source puisque le codage de canal, quant à lui, est lié au mode de transmission. Une étape supplémentaire est nécessaire afin de décrire les redondances d'information et de sélectionner l'algorithme de compression optimal.

4.2.1 Description et exploitation des redondances

La description des redondances est l'étape qui précède l'encodage des données. Cette étape est nécessaire afin de pouvoir choisir l'algorithme de compression adapté aux données à compresser. Le choix de l'algorithme est donc déterminé dynamiquement en fonction des redondances d'information dans les données entrantes. Afin de limiter les temps de traitements et pour pouvoir implémenter la compression dynamique dans des systèmes de faibles puissances, une stratégie de compression dynamique est proposée. La compression dynamique s'appuie sur une libraire de plusieurs algorithmes dont la sélection se fait à l'aide du pointeur "*descriptor pointer*" décrit précédemment dans le tableau de correspondance (section 4.1). Comme le montre la Fig. 4.7, cinq codages différents sont utilisés pour améliorer le taux de compression : RLBE, RLE, Huffman, LZW et le codage binaire. Ces codages sont regroupés dans une libraire qui peut être étendue à tout moment pour utiliser d'autres algorithmes. Notons que l'ajout d'un algorithme induit une nouvelle valeur de pointeur dans le tableau de descripteur (Tableau 4.3).

Le choix de l'algorithme est déterminé par la description des redondances dans les différences de temps δ_i et δ'_i. Les redondances sont décrites par plusieurs critères afin d'extraire le maximum d'informations disponibles.

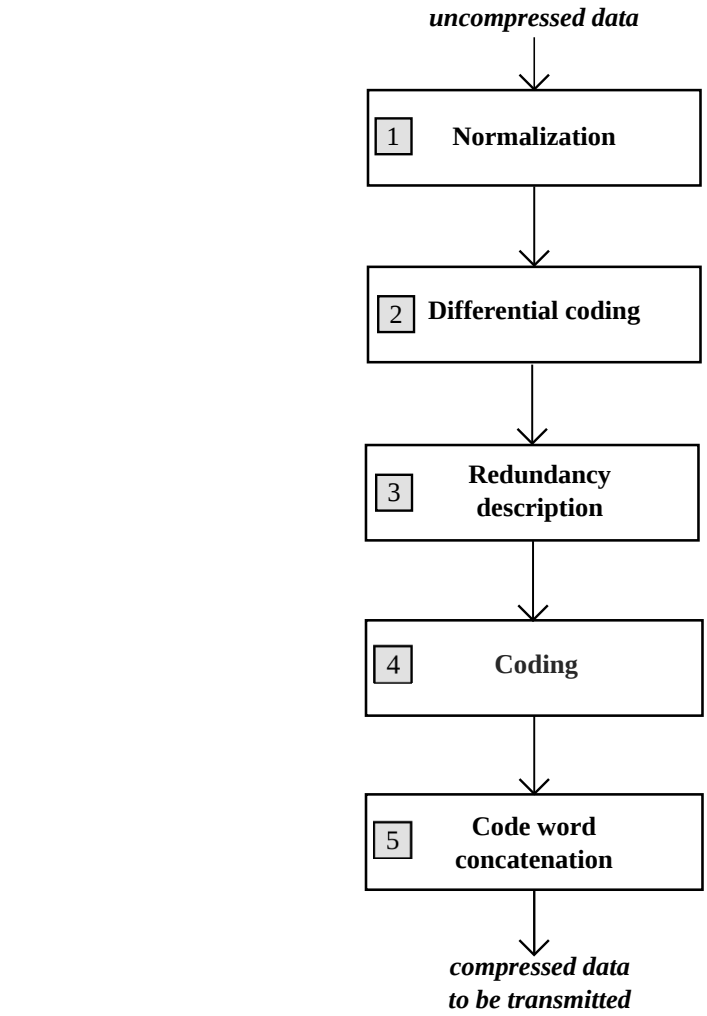


Figure 4.6 – Organigramme de la méthode proposée pour la compression de données dynamique sans perte

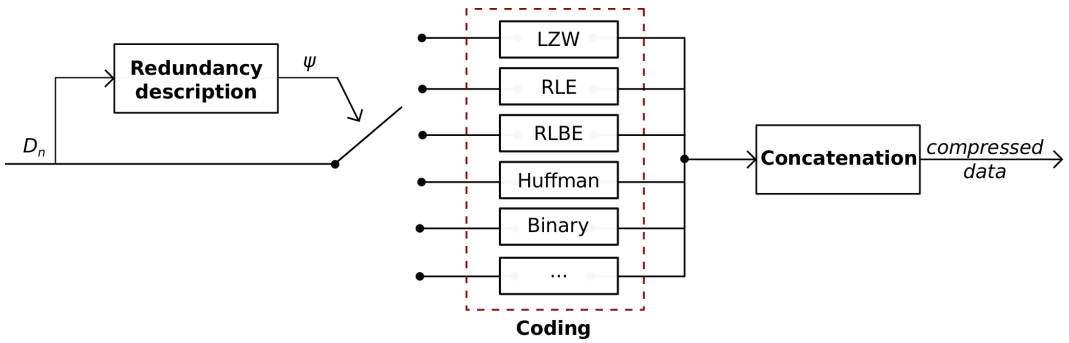


Figure 4.7 – Schéma bloc illustrant la sélection de l'algorithme de compression en fonction des redondances décrites dans les données d'origine

Afin de quantifier la dispersion des données, l'écart-type σ_x permettrait de définir rapidement le comportement des redondances lié au profil de consommation. En effet, une grande valeur de σ_x indique de faibles redondances dans les différences de temps δ_i. En revanche, il est difficile de définir un seuil de décision qui permettrait de sélectionner l'algorithme adéquat. Nous proposons alors d'utiliser les quatre critères suivants pour décrire les redondances :

- L_D , la longueur des données originales (δ_i, δ'_i) à compresser ;
- ζ , le rapport entre le nombre de données identiques successives N_{data} et le nombre total de données N_{total} : $\zeta = \frac{N_{data}}{N_{total}}$ (exprimé en %) ;
- λ , le rapport entre le nombre de code binaire successif ayant la même longueur N_{binary} et le nombre total de données N_{total} : $\lambda = \frac{N_{binary}}{N_{total}}$ (exprimé en %) ;
- P_S , la probabilité du symbole ayant la plus grande fréquence d'apparition.

Le premier critère proposé est la longueur L_D des données d'origine. Cette longueur donne une indication sur la quantité d'information dans les différences de temps à compresser. La longueur L_D permet d'estimer le nombre de redondances. Comme le montrent les tests expérimentaux effectués dans la section 2.4, une longueur L_D supérieure à 500 octets est très favorable à l'utilisation d'un codage par dictionnaire LZW. En dessous de cette valeur, d'autres descripteurs doivent être employés.

Les critères ζ et λ analysent le nombre de répétitions successives dans les différences de temps δ_i. ζ est le rapport entre le nombre de données identiques successives et le nombre de données total. λ est basé sur le même principe mais ce dernier utilise la représentation binaire. Ces ratios sont exprimés en pourcentage et fournissent une indication sur l'efficacité des algorithmes RLE et RLBE. De toute évidence, une valeur élevée de ζ engendre une valeur élevée de λ. Lorsque ces deux rapports sont petits, une description au niveau du symbole est nécessaire.

Le critère P_S s'intéresse aux redondances dans les symboles rencontrés. P_S donne la probabilité du symbole ayant la plus grande fréquence d'apparition. Ce paramètre est utilisé pour déterminer l'efficacité du codage de Huffman.

4.2.2 Sélection de l'algorithme de compression

La description et l'exploitation des redondances dans les données d'origine permettent de sélectionner l'algorithme de compression optimal. Le choix de l'algorithme de compression n'a pas été l'objectif principal de notre étude, mais les critères présentés précédemment donnent une indication sur la nature des redondances.

Afin de sélectionner l'algorithme adéquat parmi les cinq codages décrits précédemment, des seuils ont été déterminés expérimentalement. Ces seuils pourront être amenés à varier avec l'ajout d'autres algorithmes. Plusieurs tests expérimentaux ont été effectués dans des conditions réelles de fonctionnement afin de déterminer les seuils associés à chaque critère.

Au final, les seuils suivants sont utilisés pour sélectionner l'algorithme :
a) Si $\zeta \geq 70\%$ alors le codage RLE est sélectionné et $\psi = 1$;
b) Si $\zeta < 70\%$, $\lambda \geq 50\%$ et $L_D < 1000$ octets alors le codage RLBE est sélectionné et $\psi = 2$;
c) Si $\lambda < 50\%$, $P_S \geq 50$ % et $L_D < 200$ octets alors le codage Huffman est sélectionné et $\psi = 3$;
d) Si $L_D > 500$ octets alors le codage LZW est sélectionné et $\psi = 4$;
e) Dans tous les autres cas le codage binaire est sélectionné et $\psi = 5$.

Les redondances dans les données non compressées D_n sont analysées afin de déterminer la valeur de ψ. Cette valeur permet de sélectionner l'algorithme de compression adéquat. La Fig. 4.8 illustre la description des redondances et la sélection de l'algorithme de compression pour quatre séquences de données originales différentes $D_n = \{\delta_i, \delta'_i\}$, où n est le numéro de la séquence de données. Le choix de l'algorithme LZW (ψ=4) n'est pas représenté car ce dernier est plus adapté aux longues séquences de données.

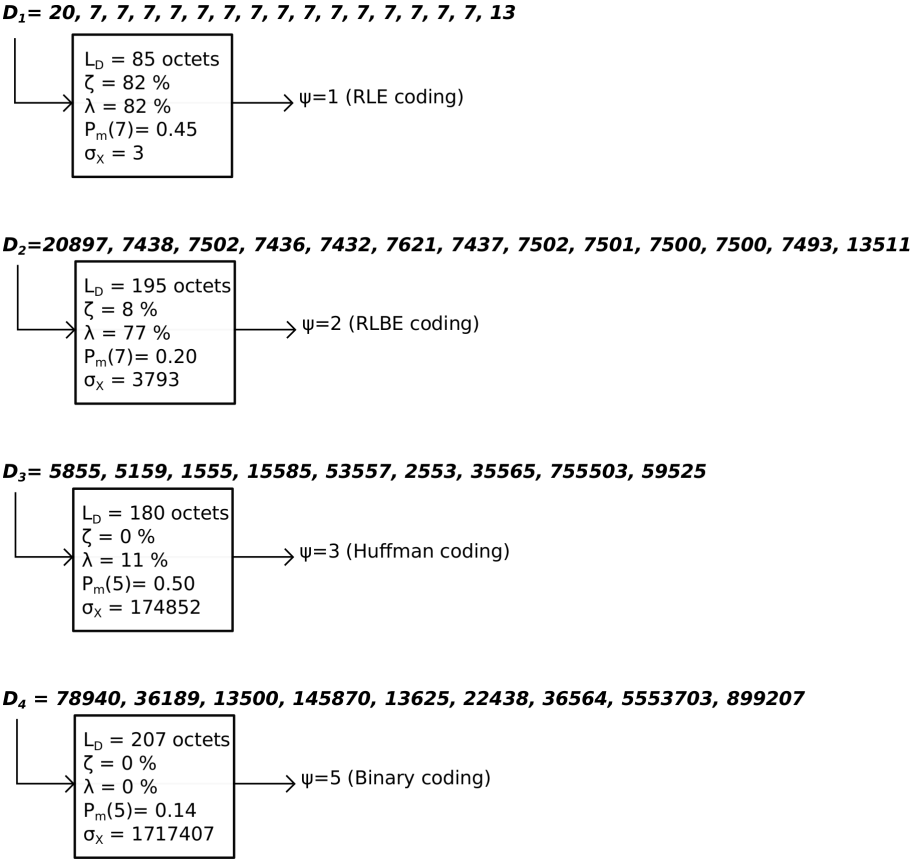


Figure 4.8 – Exemples de valeurs numériques des descripteurs servant à sélectionner l'algorithme de compression pour quatre séquences de données différentes D_1, D_2, D_3 et D_4

4.3 Performances de compression

Les performances de la compression dynamique sans perte ont été évaluées avec des données réelles de consommation d'eau. L'objectif de cette étude est de comparer l'efficacité de la compression dynamique avec celle de la compression de données RLBE, toutes les deux étant sans pertes. Les données compressées correspondent uniquement au champ "*compressed raw data*" du bloc de données présenté dans la section 4.1. Les résultats expérimentaux sont obtenus dans un premier temps avec T_{max} égal à 5 minutes et L_{max} égal à 120 octets pour les mêmes raisons que celles définies dans la section 2.4.1. Les contraintes liées à la transmission et la répétition des messages n'ont pas été considérées. Nous avons testé la compression dynamique dans quatre cas de consommations différents :

- cas 1 : un bâtiment domestique habité par deux personnes ;
- cas 2 : un bâtiment domestique habité par trois personnes ;
- cas 3 : un bâtiment tertiaire où travaillent environ 80 personnes ;
- cas 4 : un banc d'essai qui illustre un cas bien spécifique pour tester les compteurs d'eau chez Diehl Metering.

Les résultats expérimentaux relatifs à chaque cas de consommation sont illustrées dans les Tableaux 4.5, 4.6, 4.7 et 4.8. On voit que la compression dynamique offre un meilleur taux de compression. Ce dernier est toujours plus élevé que celui obtenu avec l'algorithme RLBE. En revanche, la compression dynamique induit de la complexité et davantage de calculs dans le microcontrôleur. Dans les cas domestiques et tertiaires, l'optimisation du taux de compression est négligeable. Dans ces cas-là, une compression RLBE suffit et est largement adaptée à l'utilisation. En revanche, lors de faibles consommations d'eau, la quantité de données à compresser est dérisoire et la redondance d'information est quasiment nulle. Par conséquent, les performances de compression du RLBE ne sont pas optimales. Dans ce cas le codage RLBE est remplacé par un simple codage binaire. Ce dernier réduit les temps de traitements et garantit un taux de compression supérieur ou égal à celui du codage RLBE. Enfin, nous avons répété les résultats avec T_{max} = 10 minutes et L_{max} = 500 octets pour modifier le nombre de redondances dans les données originales. La différence est négligeable. L'intervalle de transmission T_{max} et la longueur maximale des données compressées L_{max} ne modifient pas les performances de compression.

Dans des cas spécifiques, la compression dynamique est optimale et prend tout son sens. Typiquement, une consommation industrielle (cas 4) ayant des cycles répétés de consommation est plus favorable à un codage par dictionnaire. En effet, la quantité de données est plus élevée et les redondances d'information sont plus adaptées au codage LZW. La compression dynamique engendre plus de calculs mais le taux de compression est optimisé d'environ 19 % par rapport au RLBE. A l'inverse des cas domestiques et tertiaires, un intervalle de transmission T_{max} plus élevé influe sur l'efficacité de compression. Plus l'intervalle est élevé, plus la compression par dictionnaire est efficace. On voit qu'avec T_{max} = 10 minutes, le taux de compression est amélioré à 65 %.

Tableau 4.5 – Comparaison entre la compression dynamique et la compression RLBE pour une journée typique de consommation d'eau dans une habitation occupée par 2 personnes

Domestic case (197 litres in 24h)	T_{max} (min)	L_{max} (bytes)	Total data length (bytes)	Total processing time (ms)	Compression ratio (%)
No compression	5	120	1026	3.4	–
RLBE			435	3.8	57.60 %
Dynamic compression			412	5.9	59.80 %

Tableau 4.6 – Comparaison entre la compression dynamique et la compression RLBE pour une journée typique de consommation d'eau dans une habitation occupée par 3 personnes

Domestic case (423 litres in 24h)	T_{max} (min)	L_{max} (bytes)	Total data length (bytes)	Total processing time (ms)	Compression ratio (%)
No compression	5	120	2274	3.6	–
RLBE			977	5.2	57.0 %
Dynamic compression			910	7.7	60.0 %

Tableau 4.7 – Comparaison entre la compression dynamique et la compression RLBE pour une journée typique de consommation d'eau dans un bâtiment tertiaire où travaillent environ 80 personnes

Domestic case (1410 litres in 24h)	T_{max} (min)	L_{max} (bytes)	Total data length (bytes)	Total processing time (ms)	Compression ratio (%)
No compression	5	120	6078	4.8	–
RLBE			3008	8.9	50.5 %
Dynamic compression			2869	10.8	52.8 %

Tableau 4.8 – Comparaison entre la compression dynamique et la compression RLBE pour une consommation journalière de type industrielle expérimentée sur un banc d'essai

Test bench case (25763 litres in 24h)	T_{max} (min)	L_{max} (bytes)	Total data length (bytes)	Total processing time (ms)	Compression ratio (%)
No compression	5	120	31920	21.7	-
RLBE	5	120	16630	37.5	47.9
		500			
	10	120	16369		48.7
		500			
Dynamic compression	5	120	13228	152.5	58.6
		500			
	10	120	10899	160.1	65.9
		500			

4.4 Conclusion

Dans ce chapitre, la stratégie de collecte de données a été généralisée à toute grandeur physique. La généralisation permet de gérer l'encodage indépendamment de la transmission car il a été montré que les contraintes de transmission diffèrent d'une technologie à une autre (e.g., Sigfox, LoRa, NB-IoT). Par ailleurs, la stratégie proposée ne s'applique pas uniquement au comptage de l'eau et peut être étendue à d'autres mesures.

Un bloc de données universel a donc été proposé afin d'offrir de la flexibilité dans la collecte des données. Ce bloc est encodé de manière adaptative en fonction de :

- la nature des données collectées ;
- la résolution des événements horodatées ;
- l'algorithme de compression utilisé ;
- la répétition des informations dans la transmission.

Il a été montré que dans des cas particuliers les données collectées ne peuvent pas être compressées de la même manière. C'est pourquoi une nouvelle méthode de compression dynamique a été proposée. Elle permet d'adapter la compression des données sans perte afin d'optimiser le taux de compression. Les performances de la compression dynamique ont été comparées aux performances obtenues avec la méthode de compression sans perte RLBE. Les résultats ont montré que la compression dynamique garantit un meilleur taux de compression que celui obtenu avec l'algorithme RLBE seul. La quantité de données à transmettre est par conséquent réduite. En revanche, elle engendre très souvent des calculs complémentaires qui peuvent être assez importants d'un point de vue énergétique. Des tests expérimentaux effectués dans le comptage de l'eau ont permis de montrer que l'algorithme de compression sans perte RLBE est plus adaptée aux consommations domestiques et tertiaires. Cet algorithme n'est pas toujours optimal mais il offre le meilleur compromis entre le taux de compression et les temps de calculs nécessaires. En revanche, la compression dynamique permet de réduire la longueur des messages dans des cas bien spécifiques (cas industriels, fuites d'eau importante, etc.).

Cette méthode de compression offre de la flexibilité car d'autres algorithmes de compression peuvent être utilisés, ce qui permet d'adapter la compression à l'évolution de tous types de système embarqué.

Conclusion et perspectives

Conclusion

Les développements sur les compteurs d'eau communicants visent notamment à automatiser de plus en plus le relevé de la consommation à distance. Cependant, transmettre les données plus régulièrement, avec une meilleure résolution temporelle et de manière totalement autonome, est devenu une des principales priorités pour les distributeurs d'eau. Ces derniers requièrent plus d'information sur la consommation, souhaitent développer davantage de fonctionnalités et tendent vers un système hautement sécurisé. Des informations supplémentaires (e.g., la durée de la fuite, le volume d'eau perdu, le débit moyen entre deux émissions) sont indispensable pour analyser plus précisément la consommation de l'eau. L'ajout d'information nécessite de transmettre davantage de données, ce qui augmente le temps de transmission et la consommation d'énergie du module radio clipsé sur le compteur d'eau. En effet, le module radio est un système embarqué qui est limité en terme d'énergie disponible.

Afin de pallier ces problématiques, une nouvelle stratégie de collecte de données a été proposée et validée dans cette thèse. Cette stratégie a été décomposée en plusieurs étapes allant de la collecte des données dans le compteur jusqu'à l'exploitation de l'information utile dans le serveur. Les objectifs de cette thèse ont été :

- d'évaluer la quantité d'énergie qu'il est possible de récupérer dans un compteur d'eau ;
- de collecter davantage de données de consommation ;
- d'augmenter la résolution et déporter les analyses dans le serveur ;
- de réduire la consommation d'énergie du module radio.

Le chapitre 1 a présenté un état de l'art sur le comptage de l'eau. Le fonctionnement général des compteurs d'eau actuels a été décrit, allant du mesureur jusqu'à la transmission des données. Cette description a permis de définir les enjeux actuels dont les contraintes énergétiques. Dans un premier temps, nous avons cherché à récupérer l'énergie dans l'environnement des compteurs d'eau afin de répondre à la question énergétique. Deux prototypes différents ont été réalisés et testés. L'un permet de récolter l'énergie cinétique provenant du flux d'eau, l'autre permet de convertir les variations de pression dans la canalisation en énergie électrique. Les résultats ont montré que l'énergie cinétique provenant du flux d'eau est la principale source d'énergie

disponible. Celle-ci n'est pas toujours suffisante et ne permet pas d'assurer une autonomie énergétique totale. Nous avons donc complété cette approche avec une nouvelle stratégie de collecte de données dans le but d'économiser l'énergie dans la transmission. La stratégie proposée consiste à transmettre l'horodatage de chaque variation détectée par le capteur. Cela permet de collecter les données de comptage avec une meilleure résolution afin de fournir une information plus précise sur la consommation de l'eau. Cependant, une meilleure résolution engendre plus de données. Les messages deviennent plus longs et le temps de transmission augmente drastiquement. Il a été proposé de compresser les données collectées pour réduire la taille des messages, limiter le nombre de transmission et économiser l'énergie dans le compteur.

Dans ce contexte, une étude sur les algorithmes de compression sans perte a été menée dans le chapitre 2. Les algorithmes Huffman, Even-Rodeh, Exponential-Golomb, LZW, Fibonacci et une version adaptée du codage hybride Bzip2 ont été testés dans le contexte du comptage de l'eau et dans des conditions réelles de fonctionnement. Les inconvénients et les avantages de chaque algorithme ont été décrits. Afin de pallier les principaux inconvénients, une nouvelle méthode de compression sans perte a été proposée. Cette nouvelle méthode de compression, que nous avons appelée, Run-Length Binary Encoding (RLBE) est considérée comme un algorithme hybride car trois codages différents y sont regroupés. Les performances de l'algorithme RLBE ont été évaluées et comparées aux autres algorithmes. Principalement adapté à des séquences successives de longueurs binaires identiques, l'algorithme RLBE offre les meilleurs compromis entre le taux de compression et le temps de traitement. Celui-ci a été implémenté dans plusieurs modules radio existants. Par ailleurs, un bilan énergétique a été effectué pour comparer les performances des systèmes actuels avec ceux qui intègrent la nouvelle stratégie de collecte de données. Il a été montré que nos contributions permettent d'optimiser d'un facteur 20 la consommation énergétique d'un compteur d'eau communicant. L'énergie nécessaire pour compresser et transmettre les données varie entre 0,6 μW et 2,4 μW, contre environ 36 μW pour les systèmes de communication mobiles actuels. La stratégie proposée ne permet pas uniquement de réduire la consommation d'énergie ; les données collectées sont transmises avec une résolution nettement plus élevée (1 ms avec notre prototype). Cela permet de disposer de données plus précises et d'optimiser la gestion du réseau de distribution d'eau.

Dans le chapitre 3, le décodage des trames est complètement décrit, de la réception des données à la reconstruction complète du profil de consommation. Une plate-forme expérimentale a été mise en place dans des cas réels de fonctionnement. Elle consiste en une dizaine de compteurs d'eau modifiés capables de transmettre les consommations d'eau au litre prêt. Les données compressées et transmises sont collectées, décompressées et stockées sur un serveur. Cette plate-forme a permis de valider la faisabilité de la stratégie de collecte de données proposée. Les données brutes sont correctement réceptionnées et fidèlement reconstruites. Une architecture de stockage a été proposée afin de limiter le volume de données générées et d'analyser les données dans le serveur. Pour évaluer le volume de données à stocker, nous avons effectués des tests dans trois situations différentes, à savoir la consommation domestique de deux personnes, la consommation du restaurant universitaire et la consommation générale de l'IUT de Mulhouse. En moyenne, il faut prévoir environ 11 Mo pour stocker 15 ans d'enregistrement, avec une

résolution d'une milliseconde pour chaque mesure horodatée. Des premières analyses de détection de fuites ont été testées. Les données collectées peuvent être analysées pour décrire certains comportements dans le réseau de distribution. Cette nouvelle stratégie de collecte de données pourra être utilisée dans les technologies de l'*Active Assisted Living* (AAL) afin d'offrir davantage de sécurité et de services.

La stratégie de collecte de données a été généralisée à d'autres mesures dans le chapitre 4. La généralisation permet de gérer l'encodage indépendamment de la transmission car les contraintes de transmission diffèrent d'une technologie à une autre. Par ailleurs, il a été montré que la stratégie proposée ne s'applique pas uniquement au comptage de l'eau mais qu'elle peut être étendue à d'autres grandeurs physiques (température, pression, volume de gaz, etc.). Basée sur le processus de compression présenté dans le chapitre 2, une nouvelle méthode de compression dynamique a été proposée. Celle-ci consiste à adapter la manière de compresser en sélectionnant l'algorithme de compression adéquat en fonction des redondances dans les données originales. Les performances de la compression dynamique ont été évaluées avec des données réelles de consommation d'eau. De manière générale, dans les cas domestiques, l'amélioration du taux de compression est négligeable (en moyenne 6%) par rapport à celui obtenu avec l'algorithme RLBE, et ce en dépit d'un temps de traitement plus élevé. Par conséquent, dans ce type de cas, l'utilisation de l'algorithme RLBE seul est plus favorable que la compression dynamique d'un point de vue énergétique. En revanche, dans des cas bien spécifiques (fuites, cycles récurrents, etc.), la compression dynamique est plus favorable qu'un codage RLBE seul car le taux de compression peut être bien meilleur (en moyenne 25%). La quantité de données à transmettre est donc fortement réduite ainsi que l'énergie consommée par tout le système embarqué. Cette méthode de compression est flexible car d'autres algorithmes de compression peuvent être utilisés, ce qui permet d'adapter la compression à l'évolution de tous types de systèmes embarqués.

Perspectives

La nouvelle stratégie de collecte de données qui a été développée constitue une base solide pour analyser plus précisément la consommation de l'eau dans le réseau de distribution. Cette stratégie fournit de nouvelles solutions et ouvre des perspectives en terme d'utilisation. Dans le futur, nous nous intéresserons à :

- intégrer aux compteurs d'eau d'autres capteurs pour mesurer la température de l'eau, sa pression, sa conductivité, etc. ;
- implémenter la méthode de compression dynamique pour optimiser les performances de compression dans les cas particuliers ;
- caractériser et modéliser le comportement dans le réseau de distribution à l'aide de techniques d'apprentissage automatique afin de prédire les éventuelles anomalies ;
- étendre la stratégie de collecte de données à d'autres compteurs utilisés dans les applications domestiques et industrielles.

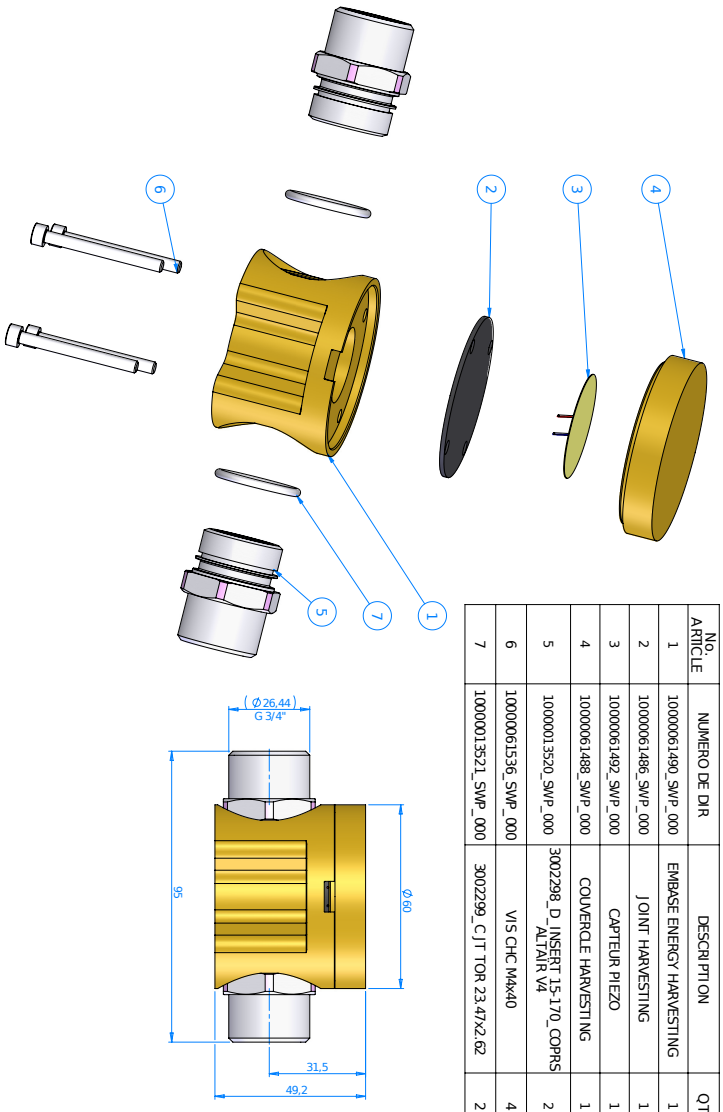
A Annexes

A.1 Liste des acronymes

AAL	Active Assisted Living
AES	Advanced Encryption Standard
AMR	Automated Meter Reading
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Network
ASCII	American Standard Code for Information Interchange
AWWA	American Water Works Association
BWT	Burrows-Wheeler Transform
BLOB	Binary Large OBjects
CRC	Cyclic Redundancy Check
CPU	Central Processing Unit
EPDM	Ethylène-propylène-Diène Monomère
FLC	Fixed-Length Code
FSPL	Free Space Path Loss
HMM	Hidden Markov Model
HPEH	Hydraulic Pressure Energy Harvesting
IALM	Intrusive Appliance Load Monitoring
IoT	Internet of Things
IQR	Inter-Quartile Range
LZW	Lempel-Ziv Welch
MEMS	Micro Electro Mechanical System
MID	Measuring Instruments Directive
MPE	Maximum Permissible Error
MTF	Move-To-Front
MUC	Multi-Utility Controller
ND	Diamètre Nominal
NIALM	Non-Intrusive Appliance Load Monitoring
NRW	Non-Revenue Water
PDF	Probability Density Function
RGPD	Règlement Général sur la Protection des Données
RLE	Run-Length Encoding
RLBE	Run-Length Binary Encoding
SCD	State Change Detection
TSDB	Time Series DataBase
UHF	Ultra High Frequency
VLC	Variable Length Code
VLQ	Variable Length Quantity

A.2 Récupération d'énergie à partir d'un transducteur piézoélectrique

Les caractéristiques du transducteur piézoélectrique pour récupérer l'énergie hydraulique dans un compteur d'eau sont indiquées par la Fig. A.1. Ce générateur produit de l'électricité à partir des variations de pression dans la canalisation. Ses performances ont été évaluées dans des conditions réelles de fonctionnement.



No. ARTICLE	NUMERO DE DIR	DESCRIPTION	QTE
1	10000061490_SWP_000	EMBASE ENERGY HARVESTING	1
2	10000061486_SWP_000	JOINT HARVESTING	1
3	10000061492_SWP_000	CAPTEUR PIEZO	1
4	10000061488_SWP_000	COUVERCLE HARVESTING	1
5	10000013520_SWP_000	3002298_D_INSERT 15-170_COPRS ALTAIR V4	2
6	10000061536_SWP_000	VIS CHC M4x40	4
7	10000013521_SWP_000	3002299_CJT TOR 23.47x2.62	2

Figure A.1 – Illustration du prototype permettant de convertir l'énergie hydraulique en énergie électrique

A.3 Récupération d'énergie à partir d'un générateur électromagnétique

Le générateur électromagnétique couplé à la turbine d'un compteur d'eau mécanique a été testé en conditions réelles pour trois environnements différents qui sont :

- un bâtiment tertiaire qui accueille environ 80 personnes ;
- une maison domestique dans laquelle habitent 2 personnes ;
- une maison domestique dans laquelle habite une personne seule.

Les puissances de sortie générées en fonction des consommations sur une période de 24h dans ces trois environnements sont illustrées respectivement par les Fig. A.2, Fig. A.3 et Fig. A.4.

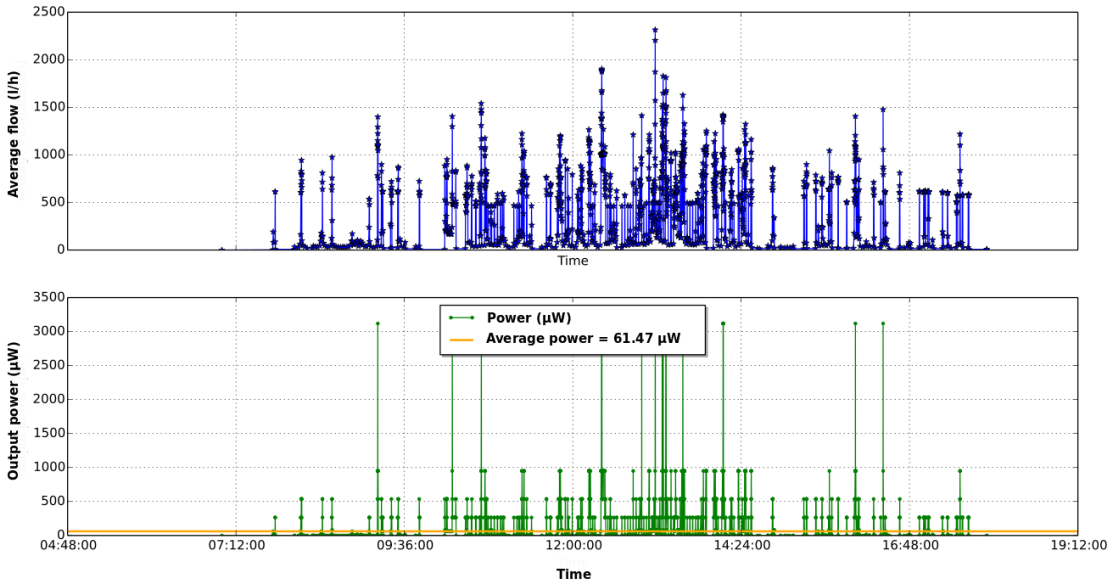


Figure A.2 – Énergie récupérée à partir d'une consommation d'eau journalière dans un bâtiment tertiaire qui accueille environ 80 personnes

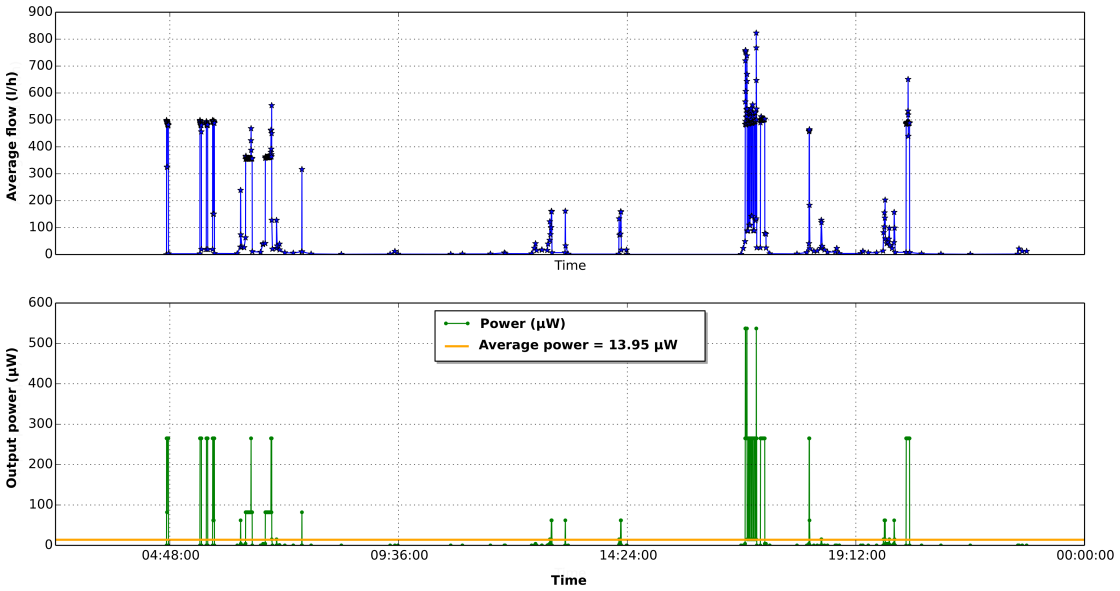


Figure A.3 – Énergie récupérée à partir d'une consommation d'eau journalière dans une maison domestique dans laquelle habitent 2 personnes

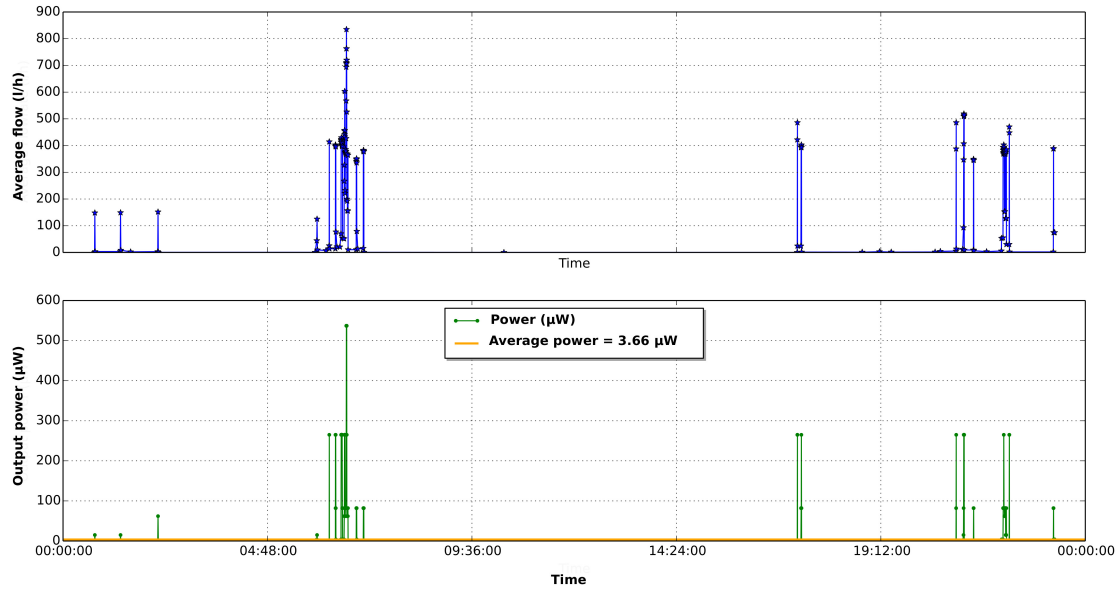


Figure A.4 – Énergie récupérée à partir d'une consommation d'eau journalière dans une maison domestique dans laquelle habite une personne seule

A.4 Modèle conceptuel de données proposé pour le stockage des données collectées

Un modèle conceptuel de données a été proposé pour le stockage des données collectées. Ce modèle est illustré sur la Fig. A.5 et est composé des cinq tables suivantes :

- *myEncodedRawFrames*: Cette table contient toutes les trames encodées et filtrées ;
- *myMeters*: Chaque trame codée est associée à un IdM qui différencie chaque compteur. Cette table contient l'IdM de chaque compteur, avec d'autres informations, telles que la description, le mot de passe et les informations personnelles ;
- *myUsers*: Cette table contient les utilisateurs qui gèrent les droits d'accès ;
- *myComments*: Cette table contient les événements détectés par les utilisateurs, e.g., les fuites, les interruptions du service d'eau, les périodes de maintenance ;
- *myRawData*: Cette table contient toutes les données décodées correspondant à l'historique de la consommation d'eau pour chaque compteur d'eau.

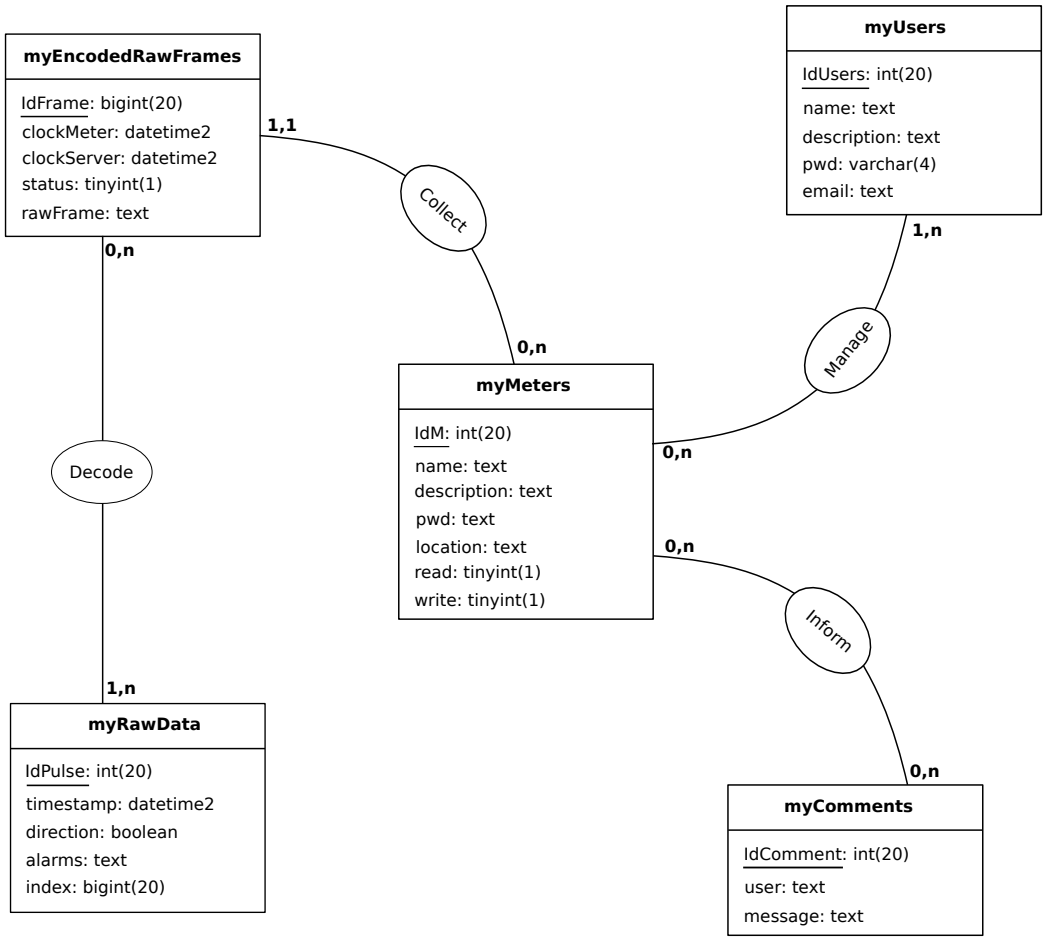


Figure A.5 – Proposition d'un modèle conceptuel de données pour stocker les données collectées dans la plate-forme expérimentale

A.5 Jeu de données décompressées et stockées sur le serveur

Dans le but de fournir un exemple concret de données réelles transmises au cours d'une journée complète par un compteur d'eau à un serveur, les mesures d'une journée type ont été enregistrées. Elles sont données ci-dessous. Les valeurs différentielles (flow) correspondent aux temps écoulés entre les événements mesurés par le compteur d'eau. Ces durées sont exprimées en millisecondes. Un événement correspond à un litre. La date de référence de la journée stockée est le 18 mai 2018 à 00h 56mn 52s 375ms et correspond à l'horodatage du premier événement. La courbe de charge et le profil de consommation correspondant sont illustrés sur la Fig. A.6. Le volume d'eau consommé dans la journée est de 290 litres. Aucun retour d'eau n'a été mesuré et transmis. Ces données constituent un jeu de test pour évaluer les différents algorithmes de compression (ceux étudiés et proposés dans cette thèse ainsi que tout algorithme de compression à venir ultérieurement).

```
% index :
idx = 28122

% variation détectée : 1 litre
delta = 1

% horodatage absolu de référence : 2018-05-18 00:56:52.375
tRef = 1526597812375

% différence de temps relatives aux variations positives :
flow = [8063, 8062, 8000, 28586375, 8563, 7562, 5750, 6875, 8437, 8438,
474188, 7500, 6187, 7250, 8375, 414938, 1536812, 8562, 7376, 7500,
8437, 8500, 8437, 964063, 8250, 8250, 8250, 8250, 8250, 8187, 7313,
6500, 7250, 8313, 8250, 8187, 190875, 8312, 8188, 8312, 8250, 8188,
8250, 8188, 7937, 8000, 8063, 49437, 1115000, 9563, 360750, 313750,
8374, 7750, 32188, 504812, 85188, 8125, 8063, 388374, 7063, 11625,
27812, 7126, 7187, 7125, 7062, 451376, 10750, 10750, 52250, 13500,
13312, 13688, 5263374, 378313, 244563, 20874, 33626, 406687, 7437,
7250, 7313, 146187, 13813, 5813, 5812, 6188, 10500, 9562, 225125,
61375, 8875, 8875, 8875, 8937, 8938, 8875, 8937, 8876, 8812, 8812,
8876, 36874, 67813, 1376875, 1246062, 274563, 2687, 2750, 2688, 2688,
1376124, 84063, 8250, 8250, 8313, 3354624, 3688, 2812, 2750, 2750,
2688, 78375, 29063, 10062, 9750, 9688, 9624, 9563, 9625, 9562, 9626,
1431562, 185312, 605376, 457812, 12000, 189688, 8187, 8125, 11424250,
2760000, 10562, 10813, 12625, 50062, 64750, 18438, 43688, 107000,
437250, 424687, 11750, 165625, 8250, 8250, 1746250, 238625, 3995625,
8125, 8250, 1220375, 9500, 9375, 9375, 9375, 9313, 9250, 9374, 9250,
9313, 9437, 9313, 9375, 9375, 9250, 9437, 9188, 9250, 9438, 9312,
9438, 9374, 9313, 9313, 9250, 9312, 9250, 9312, 9313, 9375, 9312,
9376, 9374, 9313, 9375, 9312, 9376, 9312, 9375, 8875, 8812, 8813,
8875, 8875, 8875, 8875, 8875, 8812, 8876, 8874, 8750, 8876, 8812,
8812, 37876, 10187, 10125, 10062, 10188, 9938, 10062, 10000, 10062,
10063, 10125, 10062, 10126, 10124, 10000, 10000, 10000, 10000, 10063,
10125, 10125, 10000, 10125, 9938, 10062, 10062, 10126, 10062, 7250,
7312, 7313, 12187, 25313, 38750, 8750, 8750, 8750, 8813, 8750, 8750,
8874, 8876, 8874, 193250, 11063, 353937, 25500, 9438, 5438, 4374, 4313,
4313, 4250, 4250, 6312, 7750, 8438, 8437, 8437, 8500, 81750, 8126, 8124,
8063, 8063, 8000, 8062, 8000, 7938, 1674187, 25437, 25938, 25812]

% différence de temps relatives aux variations négatives :
backflow = []
```

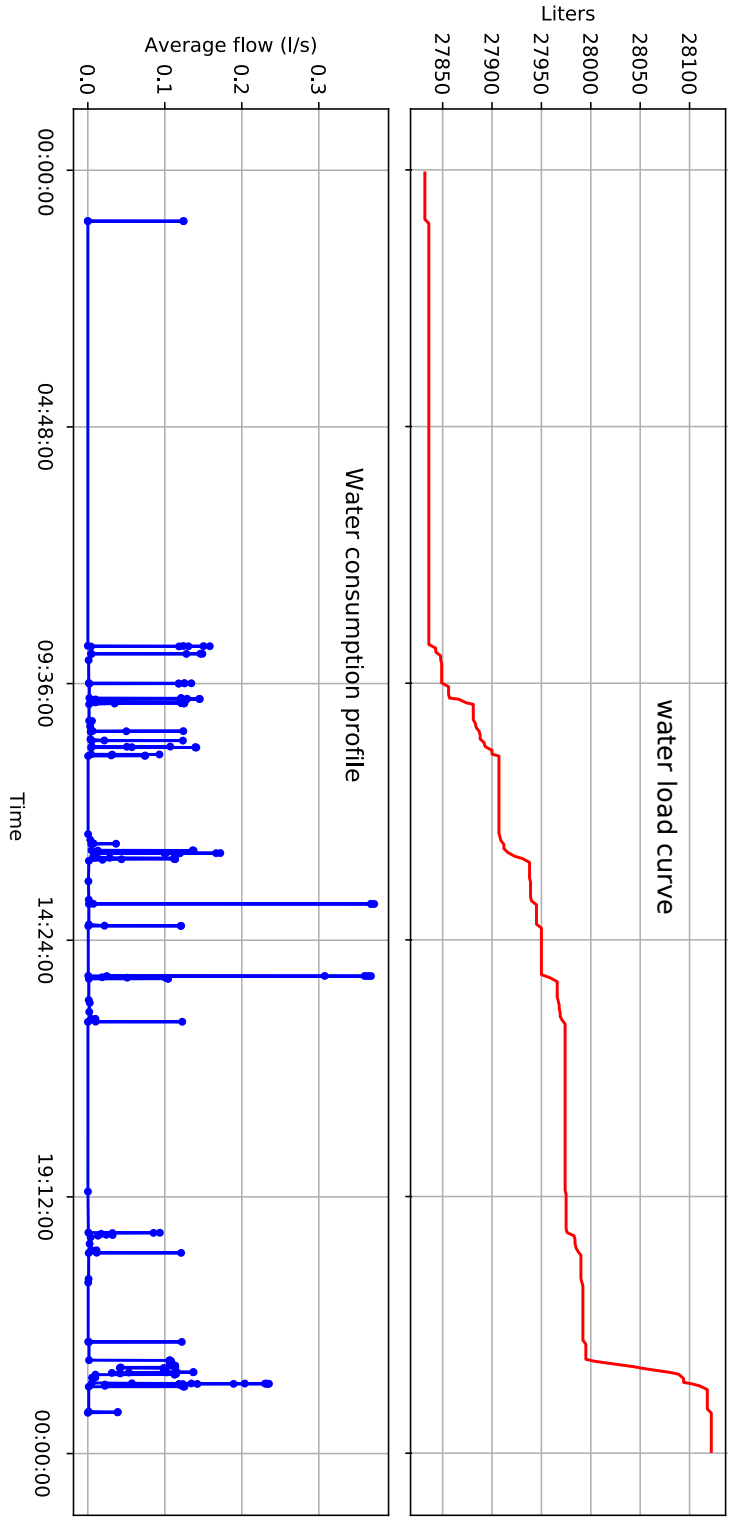


Figure A.6 – Courbe de charge et profil de consommation illustrant une consommation réelle de deux personnes en 24h

Bibliographie

[1] Oracle Utilities, "Smart metering for water utilities," An Oracle White Paper, Tech. Rep., 2009.

[2] L. Sastny, L. Franek, and P. Fiedler, "Wireless communications in smart metering," *IFAC Proceedings Volumes*, vol. 46, no. 28, pp. 330–335, 2013, 12th IFAC Conference on Programmable Devices and Embedded Systems.

[3] Y.-W. Lee, S. Eun, and S.-H. Oh, "Wireless digital water meter with low power consumption for automatic meter reading," 09 2008, pp. 639 – 645.

[4] M. P. McHenry, "Technical and governance considerations for advanced metering infrastructure/smart meters: Technology, security, uncertainty, costs, benefits, and risks," *Energy Policy*, vol. 59, pp. 834 – 842, 2013.

[5] J. Spiegel, G. Hermann, and P. Wira, "La récolte d'énergie et son application dans le comptage de l'eau," in *Congrès National de la Recherche en IUT (CNRIUT 2017), Auxerre*, 2017.

[6] J. Spiegel, P. Wira, and G. Hermann, "Energy efficiency optimization in fluid flow metering," in *2018 IEEE International Conference on Industrial Technology (ICIT 2018)*, Lyon, Feb 2018, pp. 1940–1945.

[7] J. Spiegel, G. Hermann, and P. Wira, "Energy harvesting and its application in water metering: A state-of-the-art," *Sensors*, submitted in 2018.

[8] ——, "Towards autonomous smart water meters," in *Poster presented at Journée des Ecoles Doctorales à l'Université de Haute Alsace, Mulhouse*, 2018.

[9] ——, "Toward autonomous smart water meters," in *Upper Rhine Cluster for Sustainability Research - International Conference (URCforSR 2018)*, 2018, Conference Proceedings.

[10] ——, "A comparative experimental study of compression algorithms for enhancing energy efficiency in smart meters," in *IEEE 16TH International Conference of Industrial Informatics (INDIN 2018)*, Porto, 2018, Conference Proceedings.

[11] A. Boudhaouia, J. Spiegel, and P. Wira, "Recueil et exploitation de données issues de capteurs connectés dans un contexte de télémonitoring pour batiments intelligents," in *Congrès National de la Recherche en IUT (CNRIUT 2018), Aix-en-Provence*, 2018, p. 3.

[12] Diehl Metering SAS, "Izar RC i G4," Portfolio - Software and System Components, Tech. Rep., 2018.

[13] P. Monica and M. Crainic, "A short history of residential water meters part II water meters with no moving parts," in *Installations for Buildings and Ambiental Comfort Conference XXI- edition*, Timisoara, Romania, 2012, pp. 36–43.

[14] F. Lo Castro, M. De Luca, and S. Iarossi, "Simulation of an ultrasonic flow meter for liquids," in *Sensors*, D. Compagnone, F. Baldini, C. Di Natale, G. Betta, and P. Siciliano, Eds. Cham: Springer International Publishing, 2015, pp. 397–402.

[15] A. Hamouda, O. Manck, M. L. Hafiane, and N.-E. Bouguechal, "An enhanced technique for ultrasonic flow metering featuring very low jitter and offset," *Sensors*, vol. 16, no. 7, 2016.

[16] "Awwa calls for water infrastructure investment," *Filtration Industry Analyst*, vol. 2003, no. 5, pp. 2–3, 2003.

[17] E. Standards, "Measuring Instruments Directive (MID)," 2004/22/EC, Tech. Rep., 2006.

[18] R. R. Mohassel, A. Fung, F. Mohammadi, and K. Raahemifar, "A survey on advanced metering infrastructure," *International Journal of Electrical Power and Energy Systems*, vol. 63, pp. 473–484, 2014.

[19] M. Kuzlu, M. Pipattanasomporn, and S. Rahman, "Review of communication technologies for smart homes/building applications," in *2015 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, Nov 2015, pp. 1–6.

[20] L. Šastný, L. Franek, and P. Fiedler, "Wireless communications in smart metering," *IFAC Proceedings Volumes*, vol. 46, no. 28, pp. 330–335, 2013, 12th IFAC Conference on Programmable Devices and Embedded Systems.

[21] S. Squartini, L. Gabrielli, M. Mencarelli, M. Pizzichini, S. Spinsante, and F. Piazza, "Wireless m-bus sensor nodes in smart water grids: The energy issue," *2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP)*, pp. 614–619, 06 2013.

[22] K. Zeman, P. Masek, J. Krejčí, A. Ometov, J. Hosek, S. Andreev, and F. Kröpfl, "Wireless m-bus in industrial iot: Technology overview and prototype implementation," *European Wireless 2017; 23th European Wireless Conference*, 2017.

[23] ICT and communication, "Communication systems for meters," CEN – EN 13757, Tech. Rep., 2013.

[24] M. El Guedri, "Caractérisation aveugle de la courbe de charge électrique : Détection, classification et estimation des usages dans les secteurs résidentiel et tertiaire," Theses, Université Paris Sud - Paris XI, Nov. 2009.

[25] C. Pan, M. Xie, and J. Hu, "Maximize energy utilization for ultra-low energy harvesting powered embedded systems," in *2017 IEEE 23rd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, Aug 2017, pp. 1–6.

[26] A. Hande, T. Polk, W. Walker, and D. Bhatia, "Indoor solar Energy Harvesting for Sensor Network router nodes," *Microprocessors and Microsystems*, vol. 31, no. 6, pp. 420–432, September 2007.

[27] R. Calio, U. Rongala, D. Camboni, M. Milazzo, C. Stefanini, G. de Petris, and C. Oddo, "Piezoelectric Energy Harvesting Solutions," *Sensors*, vol. 14, no. 3, pp. 4755–4790, March 2014.

[28] Y. Uzun, "Design and implementation of rf energy harvesting system for low-power electronic devices," *Journal of Electronic Materials*, vol. 45, no. 8, pp. 3842–3847, Aug 2016.

[29] S. LeBlanc, "Thermoelectric generators: Linking material properties and systems engineering for waste heat recovery applications," *Sustainable Materials and Technologies*, vol. 1-2, pp. 26–35, 2014.

[30] G. Ye and K. Soga, "Energy Harvesting from Water Distribution Systems," *Journal of Energy Engineering*, vol. 138, no. 1, pp. 7–17, 2012.

[31] H. Ostaffe, "RF Energy Harvesting Enables Wireless Sensor Networks – Sensors Mags," 2009.

[32] J.-W. Zhang, Y. Huang, and P. Cao, "An Investigation of Wideband Rectennas for Wireless Energy Harvesting," *Wireless Engineering and Technology*, vol. 05, no. 04, pp. 107–116, 2014.

[33] C. Mikeka and H. Arai, "Design Issues in Radio Frequency Energy Harvesting System," in *Sustainable Energy Harvesting Technologies - Past, Present and Future*, Y. K. Tan, Ed. InTech, dec 2011.

[34] R. Vyas, H. Nishimoto, M. Tentzeris, Y. Kawahara, and T. Asami, "A battery-less, energy harvesting device for long range scavenging of wireless power from terrestrial TV broadcasts," in *Microwave Symposium Digest (MTT), 2012 IEEE MTT-S International*. IEEE, 2012, pp. 1–3.

[35] M. Pinuela, P. D. Mitcheson, and S. Lucyszyn, "Ambient RF Energy Harvesting in Urban and Semi-Urban Environments," *IEEE Transactions on Microwave Theory and Techniques*, vol. 61, no. 7, pp. 2715–2726, July 2013.

[36] E. A. Kadir, A. P. Hu, M. Biglari-Abhari, and K. C. Aw, "Indoor WiFi energy harvester with multiple antenna for low-power wireless applications," *DeepDyve*, June 2014.

[37] A. Mouapi, N. Hakem, and G. Y. Delisle, "A new approach to design of rf energy harvesting system to enslave wireless sensor networks," *ICT Express*, 2017.

[38] M. Nalini, J. V. N. Kumar, R. M. Kumar, and M. Vignesh, "Energy harvesting and management from ambient rf radiation," in *2017 International Conference on Innovations in Green Energy and Healthcare Technologies (IGEHT)*, March 2017, pp. 1–3.

[39] L. Simeone, M. G. Tehrani, and S. J. Elliott, "Design of an electromagnetic-transducer energy harvester," *Journal of Physics: Conference Series*, vol. 744, no. 1, p. 12084, 2016.

[40] P. Becker, B. Folkmer, R. Goepfert, D. Hoffmann, A. Willmann, and Y. Manoli, "Energy autonomous wireless water meter with integrated turbine driven energy harvester," *Journal of Physics: Conference Series*, vol. 476, no. 1, pp. 012–06, 2013.

[41] D. Hoffmann, A. Willmann, R. Gopfert, P. Becker, B. Folkmer, and Y. Manoli, "Energy Harvesting from Fluid Flow in Water Pipelines for Smart Metering Applications," *Journal of Physics Conference Series*, vol. 476, pp. 012–014, dec 2013.

[42] C. Wei and X. Jing, "A comprehensive review on vibration energy harvesting: Modelling and realization," *Renewable and Sustainable Energy Reviews*, vol. 74, pp. 1–18, 2017.

[43] P. Vasan, S. Kirubaveni, and B. Sreeja, "Vibration energy harvesting for low power devices," in *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, Nov 2016, pp. 1–4.

[44] D.-A. Wang and N.-Z. Liu, "A shear mode piezoelectric energy harvester based on a pressurized water flow," *Sensors and Actuators A: Physical*, vol. 167, no. 2, pp. 449–458, jun 2011.

[45] C. Mo, L. J. Radziemski, and W. W. Clark, "Experimental validation of energy harvesting performance for pressure-loaded piezoelectric circular diaphragms," *Smart Materials and Structures*, vol. 19, no. 7, jul 2010.

[46] K. A. Cunefare, E. A. Skow, A. Erturk, J. Savor, N. Verma, and M. R. Cacan, "Energy harvesting from hydraulic pressure fluctuations," *Smart Materials and Structures*, vol. 22, no. 2, pp. 025–036, Feb 2013.

[47] G. V. Merrett and A. S. Weddell, "Supercapacitor leakage in energy-harvesting sensor nodes: Fact or fiction?" in *2012 Ninth International Conference on Networked Sensing (INSS)*, June 2012, pp. 1–5.

[48] M. Nelson, *The Data Compression Book*. New York, NY, USA: Henry Holt and Co., Inc., 1991.

[49] A. Moffat and A. Turpin, *Compression and Coding Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 2002.

[50] D. Salomon and G. Motta, *Handbook of Data Compression*, 5th ed. Springer Publishing Company, Incorporated, 2009.

[51] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Technical Journal*, vol. 27, pp. 623–656, 1948.

[52] H. Yamamoto, "Rate-distortion theory for the shannon cipher system," *IEEE Transactions on Information Theory*, vol. 43, no. 3, pp. 827–835, May 1997.

[53] G. K. Wallace, "The jpeg still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, Feb 1992.

[54] K. Brandenburg, "Mp3 and aac explained," in *Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding*, Sep 1999.

[55] C. Steinruecken, "Lossless data compression," Ph.D. dissertation, University of Cambridge, 2014.

[56] F. Benatti, S. K. Oskouei, and A. S. D. Abad, "Quantum entropy and complexity," *CoRR*, vol. abs/1705.09449, 2017.

[57] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*. New York, NY, USA: Cambridge University Press, 2002.

[58] R. Gray, *Entropy and Information Theory*, ser. SpringerLink : Bücher. Springer US, 2011.

[59] S. Foldes and N. Singhi, "On instantaneous codes," *Journal of Combinatorics, Information and System Sciences*, vol. 31, 01 2006.

[60] C. Heuberger, D. Krenn, and S. Wagner, "Canonical trees, compact prefix-free codes and sums of unit fractions: A probabilistic analysis," *SIAM J. Discrete Math.*, vol. 29, no. 3, pp. 1600–1653, 2015.

[61] D. Salomon, *Variable-length Codes for Data Compression*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[62] K. V. Kaipa, "An improvement of the asymptotic elias bound for non-binary codes," *CoRR*, vol. abs/1705.07785, 2017.

[63] D. Adjeroh and F. Nan, "Suffix-sorting via shannon-fano-elias codes," *Algorithms*, vol. 3, no. 2, pp. 145–167, 2010.

[64] J. Rissanen, "Complexity of strings in the class of markov sources," *IEEE Transactions on Information Theory*, vol. 32, no. 4, pp. 526–532, Jul 1986.

[65] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 653–664, May 1995.

[66] H. Björklund, J. Björklund, and N. Zechner, "Compression of finite-state automata through failure transitions," *Theoretical Computer Science*, vol. 557, pp. 87–100, 2014.

[67] Q. Hibon and L. C. Paulson, "Source coding theorem," *Archive of Formal Proofs*, oct 2016.

[68] R. Gallager and D. van Voorhis, "Optimal source codes for geometrically distributed integer alphabets (corresp.)," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 228–230, March 1975.

[69] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the Institute of Radio Engineers*, vol. 40, no. 9, pp. 1098–1101, September 1952.

[70] D. R. McIntyre and M. A. Pechura, "Data compression using static huffman code-decode tables," *Commun. ACM*, vol. 28, no. 6, pp. 612–616, jun 1985.

[71] D. E. Knuth, "Dynamic huffman coding," *J. Algorithms*, vol. 6, no. 2, pp. 163–180, jun 1985.

[72] X. Ruan and R. Katti, "Reducing the length of shannon-fano-elias codes and shannon-fano codes," in *Proceedings of the 2006 IEEE Conference on Military Communications*, ser. MILCOM'06. Piscataway, NJ, USA: IEEE Press, 2006, pp. 810–816.

[73] P. G. Howard and J. S. Vitter, "Practical implementations of arithmetic coding," in *IN IMAGE AND TEXT*. Kluwer Academic Publishers, 1992, pp. 85–112.

[74] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Commun. ACM*, vol. 30, no. 6, pp. 520–540, jun 1987.

[75] M. Gao, "An arithmetic coding scheme for blocked-based compressive sensing of images," *CoRR*, vol. abs/1604.06983, 2016.

[76] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the h.264/avc video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620–636, July 2003.

[77] F. Aulí-Llinàs, "Context-adaptive binary arithmetic coding with fixed-length codewords," *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 1385–1390, Aug 2015.

[78] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 13, no. 7, pp. 560–576, jul 2003.

[79] A. Unterweger and D. Engel, "Resumable load data compression in smart grids," *IEEE Transactions on Smart Grid*, vol. 6, no. 2, pp. 919–929, March 2015.

[80] P. Elias, "Universal codeword sets and representations of the integers," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 194–203, March 1975.

[81] S. Even and M. Rodeh, "Economical encoding of commas between strings," *Commun. ACM*, vol. 21, no. 4, pp. 315–317, apr 1978.

[82] I. Richardson, *The H.264 Advanced Video Compression Standard.* Wiley, 2010.

[83] S. Nakatsuka, T. Hamabe, Y. Takeuchi, and M. Imai, "An efficient lossless data compression method based on exponential-golomb coding for biomedical information and its implementation using asip technology," in *2013 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2013, pp. 382–385.

[84] A. F. Horadam, *Zeckendorf Representations of Positive and Negative Integers by Pell Numbers.* Dordrecht: Springer Netherlands, 1993, pp. 305–316.

[85] C. M. Campbell, E. F. Robertson, and R. M. Thomas, *Fibonacci Numbers and Groups.* Dordrecht: Springer Netherlands, 1988, pp. 45–60.

[86] F. P. Miller, A. F. Vandome, and J. McBrewster, *Levenshtein Distance: Information Theory, Computer Science, String (Computer Science), String Metric, Damerau?Levenshtein Distance, Spell Checker, Hamming Distance.* Alpha Press, 2009.

[87] L. Groupe, *Code Préfixe: Alphabet Morse, Liste Des Indicatifs Téléphoniques Internationaux Par Indicatif, Utf-8, Code Génétique, Codage de Huffman.* General Books, 2010.

[88] R. Sugiura, Y. Kamamoto, N. Harada, and T. Moriya, "Optimal golomb-rice code extension for lossless coding of low-entropy exponentially distributed sources," *IEEE Trans. Information Theory*, vol. 64, no. 4, pp. 3153–3161, 2018.

[89] R. Malvar, "Adaptive run-length / golomb-rice encoding of quantized generalized gaussian sources with unknown statistics," in *Data Compression Conference.* Institute of Electrical and Electronics Engineers, Inc., March 2006.

[90] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 337–343, May 1977.

[91] ——, "Compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. 24, no. 5, pp. 530–536, September 1978.

[92] J. A. Storer and T. G. Szymanski, "Data compression via textual substitution," *J. ACM*, vol. 29, no. 4, pp. 928–951, oct 1982.

[93] J. Ström and P. Wennersten, "Lossless Compression of Already Compressed Textures," in *Eurographics/ ACM SIGGRAPH Symposium on High Performance Graphics*, C. Dachsbacher, W. Mark, and J. Pantaleoni, Eds. ACM, 2011.

[94] P. Deutsch, "Deflate compressed data format specification version 1.3," IETF, United States, Tech. Rep. Request for Comments (RFC) 1951, 1996.

[95] T. A. Welch, "A technique for high-performance data compression," *Computer*, vol. 17, no. 6, pp. 8–19, June 1984.

[96] H. Zhang, X. p. Fan, S. q. Liu, and Z. Zhong, "Design and realization of improved lzw algorithm for wireless sensor networks," in *International Conference on Information Science and Technology*, March 2011, pp. 671–675.

[97] B. Alik and N. Lukač, "Chain code lossless compression using move-to-front transform and adaptive run-length encoding," *Image Commun.*, vol. 29, no. 1, pp. 96–106, jan 2014.

[98] T. Gagie and G. Manzini, "Move-to-front, distance coding, and inversion frequencies revisited," *Theoretical Computer Science*, vol. 411, no. 31, pp. 2925–2944, 2010.

[99] D. Adjeroh, T. Bell, and A. Mukherjee, *The Burrows-Wheeler Transform: Data Compression, Suffix Arrays, and Pattern Matching*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[100] R. Patel, *Parallel Lossless Data Compression on the GPU.* University of California, Davis, 2012.

[101] R. Ye, A. Boukerche, H. Wang, X. Zhou, and B. Yan, "Recodan: An efficient redundancy coding-based data transmission scheme for wireless sensor networks," *Computer Networks*, vol. 110, 10 2016.

[102] A. Cominola, M. Giuliani, D. Piga, A. Castelletti, and A. Rizzoli, "Benefits and challenges of using smart meters for advancing residential water demand modeling and management: A review," *Environmental Modelling & Software*, vol. 72, pp. 198–214, 2015.

[103] G. W. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, Dec 1992.

[104] O. Parson, "Unsupervised training methods for non-intrusive appliance load monitoring from smart meter data," PhD thesis, University of Southampton, 2014.

[105] M. Bakker, H. van Duist, K. van Schagen, J. Vreeburg, and L. Rietveld, "Improving the performance of water demand forecasting models by using weather input," *Procedia Engineering*, vol. 70, pp. 93–102, 2014, 12th International Conference on Computing and Control for the Water Industry, CCWI2013.

[106] J. Duran-Encalada, A. Paucar-Caceres, E. Bandala, and G. Wright, "The impact of global climate change on water quantity and quality: A system dynamics approach to the us–mexican transborder region," *European Journal of Operational Research*, vol. 256, no. 2, pp. 567–581, 2017.

[107] J. Levine, "An algorithm for synchronizing a clock when the data are received over a network with an unstable delay," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 63, no. 4, pp. 561–570, April 2016.

[108] V. Autefage, S. Chaumette, and D. Magoni, "Comparison of time synchronization techniques in a distributed collaborative swarm system," in *2015 European Conference on Networks and Communications (EuCNC)*, June 2015, pp. 455–459.

[109] A. Euzen, "Que se cache-t-il derrière les courbes de consommation d'eau ? L'exemple de Paris." in *15èmes Journées Scientifiques de l'Environnement - Usages de l'eau : synergies et conflits*, ser. Journées Scientifiques de l'Environnement, D. Thevenot, Ed., vol. JSE-2004, no. 8. Créteil, France: HAL, May 2004, 11 p.

[110] A. Kipf, W. Brunette, J. Kellerstrass, M. Podolsky, J. Rosa, M. Sundt, D. Wilson, G. Borriello, E. Brewer, and E. Thomas, "A proposed integrated data collection, analysis and sharing platform for impact evaluation," *Development Engineering*, vol. 1, pp. 36–44, 2016.

[111] T. Dunning, E. Friedman, M. Loukides, and R. Demarest, *Time Series Databases: New Ways to Store and Access Data*. O'Reilly Media, Incorporated, 2014.

[112] L. Pastor-Jabaloyes, F. J. Arregui, and R. Cobacho, "Water end use disaggregation based on soft computing techniques," *Water*, vol. 10, no. 1, 2018.

[113] D. Walker, E. Creaco, L. Vamvakeridou-Lyroudia, R. Farmani, Z. Kapelan, and D. Savić, "Forecasting domestic water consumption from smart meter readings using statistical methods and artificial neural networks," *Procedia Engineering*, vol. 119, pp. 1419–1428, 2015, computing and Control for the Water Industry (CCWI2015) Sharing the best practice in water management.

[114] P. Klein, "Non-intrusive information sources for activity analysis in ambient assisted living scenarios," Thèse de Doctorat, Université de Haute Alsace, Mulhouse, 2015.

[115] B. Meudic, "Automatic meter extraction from MIDI files," in *JIM*, Marseille, France, Jun. 2002, pp. 1–1.