Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

Projet WAVES : Des flux de données brutes et hétérogènes à l'information qualifiée

N° du contrat F1411006 Q Date de début 2 juin 2014 Durée 36 mois

Livrable D5.1
Dataset et Datastream Visualisation : Rapport d'étude

1. Statut

Niveau dissémination	Publique
Date d'échéance	Mois 12, 30/06/2015
Date de soumission	Mois 13, 03/07/2015
Work Package	5
Tâche T	Dataset et Datastream Visualisation
statut d'approbation	Draft
Version	0.1
Nombre de Pages	1
Nom du fichier	D5.1 - Datavis Rapport

2. Historique

Version	Date	Revu par
0.0	26/06/2015	-
0.1	03/07/2015	

3. Auteurs

Organisation	Nom	Contact
Data-Publica	Loïc Petit	loic.petit@data-publica.com

4. Résumé

Le but de ce livrable est de définir l'objectif de la visualisation et les critères qui définiront son choix dans notre cadre applicatif : la gestion de données et de **flux** de données.

Pour cela nous proposons une approche qui dresse les différentes catégories de visualisations ainsi que de dynamiques de données et comment ces deux s'intersectent. Enfin, nous proposons une approche architecturale qui permet de composer des tableaux de bords de visualisation de données dans notre cadre applicatif.

Contenu

5. Introduction

Le but de ce livrable est de définir l'objectif de la visualisation et les critères qui définiront son choix : que veut-on montrer, quel type d'interaction envisage-t-on, quel est le public ? Ce livrable doit aussi analyser les contraintes de cette visualisation dans notre cadre applicatif : prise en compte du format des données et prise en compte des contraintes de la dynamique des données.

La visualisation est devenue un passage obligatoire pour les applications qui exploitent des données afin de montrer à un public non averti scientifiquement l'étendue d'une étude. Il devient critique de s'armer d'outils pour produire des visualisations des données que nous avons dans notre application. Toutefois, dans notre cadre, les données évoluent dans le temps et ceci doit potentiellement être pris en compte pour la visualisation de données.

Dans ce livrable, nous identifierons premièrement les besoins de visualisations pour mieux appréhender notre problématique. Ensuite, nous analyserons l'impact de la dynamique des données sur ces applications. Nous continuerons par l'étude de l'architecture logicielle d'une application de visualisation en s'arrêtant sur les impacts lié à notre cadre. Enfin, nous finirons par exposer notre proposition de contribution technologique dans ce projet.

6. De l'impact de la visualisation

6.1. Intérêt de la visualisation

La visualisation de donnée (ou *dataviz*) telle qu'on l'entend actuellement ne doit jamais être faite de façon anodine. Elle est porteuse d'un message et c'est pour cela qu'elle reste avant tout un concept communicatif. Toutefois, elle peut être utilisée en tant que support aux scientifiques même si les outils utilisés peuvent différer car ils seront des fois plus spécifiques et plus techniques.

Pour rappel, la visualisation de donnée se différencie de façon majeure avec l'infographie sur une caractéristique principale : l'interaction. Une infographie conduit un message mais n'offre pas d'alternative sur son interprétation, une visualisation de donnée se démarque par l'interprétation en générale moins guidée et plus exploratoire.

De notre expérience, après avoir fournis de nombreuses visualisations pour nos clients en interne, la visualisation de donnée passe par la réponse à trois questions majeures qui doivent être posées dans cet ordre :

- **Quoi ?** – i.e. quelle donnée voulons-nous montrer, quelles sont ses caractéristiques ?
- **Pourquoi ?** – i.e. dans quel but voulons-nous exposer ces données, quel est le message principal qui doit passer à travers cette visualisation ?

- **Comment ?** – i.e. avec quel moyen le message va être véhiculé ?

Nous arguons qu'une fois ces questions correctement posées et répondues de façon argumentées, ensuite l'implémentation de la visualisation peut être appliquée.

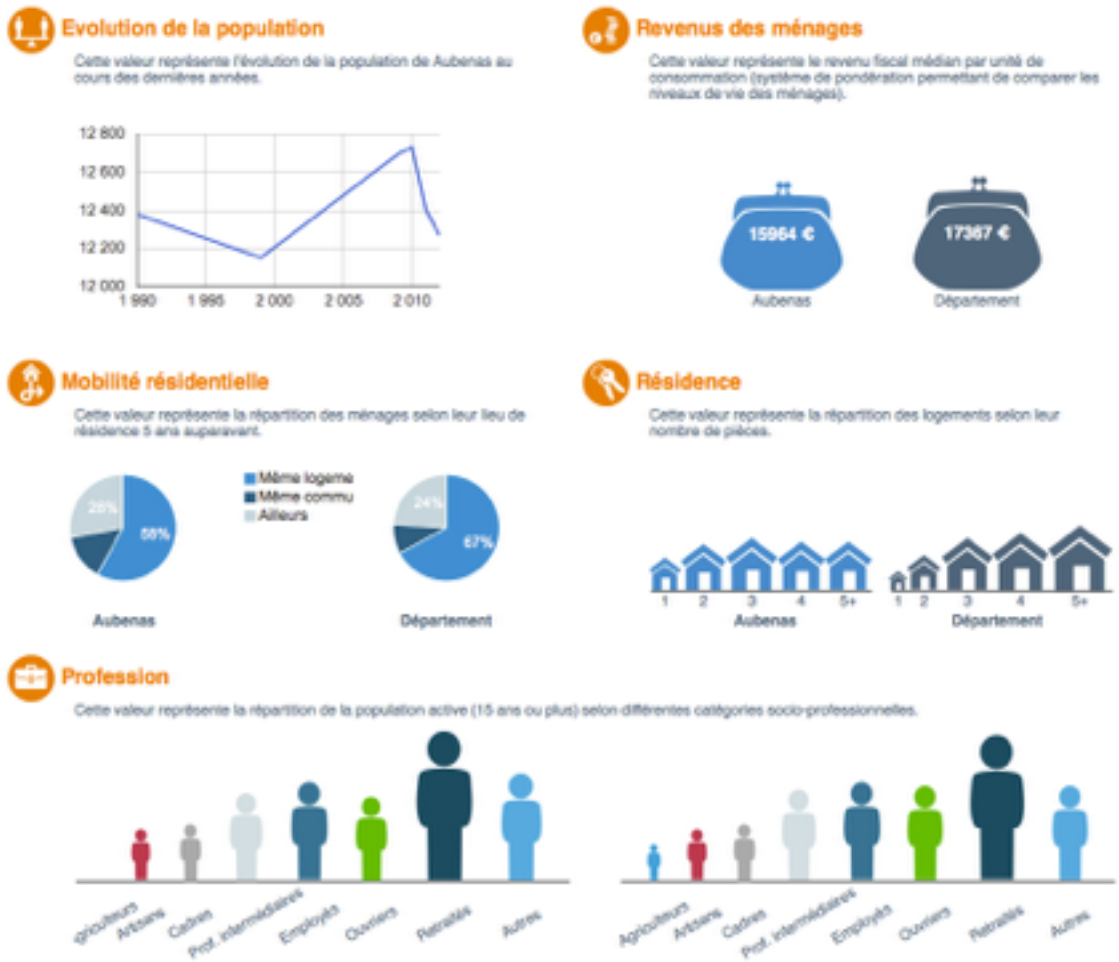
Sans rentrer dans les détails d'implémentations, il existe de nombreuses façon de représenter des données et nous exposerons différents composants graphiques au cours de ce document.

6.2. Catégories d'applications de visualisation

Les visualisations se composent en application pour ensuite former un tout cohérent qui permettra de faire passer un ou plusieurs messages. Dans cette section, nous exposons trois types applications de visualisations de données.

6.2.1. Tableaux de bords

Les tableaux de bords sont une compositions de visualisation afin d'établir un bilan d'une situation sur un sujet composée de nombreuse dimensions. Ci-dessous, un extrait du tableau de bord des communes de Data-Publica pour une commune donnée. On y voit de nombreuse visualisation comparatives.



[Quoi] Les données en entrée sont issues de nombreuses sources et elles ont pour particularité d'être fortement couplées ce qui apporte une grande complexité sur le message à faire passer.

[Pourquoi] Les tableaux de bords ont pour but de faire des bilans et des rapports afin d'établir des diagnostiques sur un ensemble complexe.

[Comment] La complexité des données doit être compensée par des visualisations très simples (histogrammes par exemple) et une interaction simpliste pour mieux guider la personne. Dans quelques exceptions, des visualisations complexes pourront être mises en place telles que des représentations à base de tracé de graphes afin de visualiser des relations (ou des réseaux).

6.2.2. Visualisation Exploratoire

La visualisation exploratoire se caractérise par sa restriction en terme de nombre de jeux de données mais par une très grand interractivité. Ci-dessous, nous

pouvons retrouver un extrait d'une application d'exploration de sondage produite par Data-Publica.

Bien que ce soit délicat de parler d'interactivité dans un rapport écrit, nous pouvons compter 12 points d'interactions avec uniquement quelques visualisations.

Si nous résumons cette catégorie avec nos questions, nous avons :

[Quoi] Les données sont très restreintes. Seuls un ou deux jeux de données seront mis en avant. Le nombre d'attributs lui peut être très grand potentiellement, mais les liens entre données hétérogènes est plus limité.

[Pourquoi] Le but est de comprendre en profondeur un phénomène particulier. Ce genre d'outil n'est exposé qu'aux personnes qui sont plus à même d'interpréter les résultats.

[Comment] La visualisation exploratoire n'a pas de support privilégié car cela va majoritairement dépendre de ce que l'expert métier souhaite. Sa conception est du coup plus délicate puisque le concepteur de la visualisation aura plus de mal à interpréter le résultat. Toutefois, nous pouvons remarquer la plus grande présence d'interaction face au nombre de visualisations.

6.2.3. Supervision de système

La supervision permet de visualiser en un clin d'œil l'état d'un système qui est actualisé en temps réel. Ci-dessous, une application de supervision extraite de la présentation produit de CA Technologies[1].

[Quoi] Nous souhaitons représenter l'état d'un système (pas spécialement informatique) en temps réel.

[Pourquoi] Le but d'une application de supervision est de comprendre et réagir rapidement sur l'arrivée d'un événement particulier.

[Comment] Ce qui est intéressant et très spécifique à ce type d'application réside sur son instance qui ne contient que des visualisations extrêmement simplifiées. L'histogramme reste dans ce cadre la visualisation la plus complexe, il sera préférable d'utiliser des couleurs simples (vert, jaune, rouge), ou des jauges. De plus, l'historique local d'une métrique sous forme de graphique fait partie des visualisations courantes.

7. Impact de la vélocité des données

Nous avons revu trois types d'applications qui nous permettent de mieux cerner l'usage et les types de visualisations utilisées. Désormais, nous allons voir en détail

[1] Data-Publica possède de nombreuses visualisations de supervision, toutefois nous ne pouvons pas exposer des captures d'écrans qui révèlent systématiquement des informations internes.

l'impact du fait que les données évoluent en fonction du temps et les types de visualisation que nous pouvons appliquer.

7.1. Définitions des différentes dynamiques de données

Les données se classent en deux catégories dans leurs rapports avec le temps : froid ou chaud. Notons qu'une donnée ici n'est pas considéré comme atomique et qu'un n-uplet par exemple est considéré dans son ensemble.

7.1.1. Données froides

Les données froides sont définies par leur ignorance du concept de temps a priori. Il reste toutefois trois sous catégories à celle-ci.

Premièrement, les données descriptives sont des données qui ne varient pas dans le temps de par leur nature. Par exemple, l'identifiant d'un capteur ou sa marque ou sa nature sont des données statiques. La plupart des visualisations se basent donc sur des représentations textuelles et/ou colorées telles que du texte brut, une étiquette ou une couleur.

De manière similaire, nous pouvons retrouver les données qui sont des résultats spontanés. Ces données ne sont pas bruts car elles sont calculées mais gardent la propriété d'être figée dans le temps. Toutefois, ces données ont été générées à un instant dans le temps, qui reste inconnu a priori. Les modes de visualisations liées restent dépendant de la nature mais les modes de visualisations exploratoires spécifiques sont directement applicable ici.

Enfin, nous pouvons citer les données dites **stables**. Ces données sont des données descriptives mais qui ne sont toutefois pas immuables et peuvent subir des modifications. Leur mode de représentation est entièrement dépendant de la nature des données mais pour donner des exemples : la carte de répartition des capteurs ou la fiche détaillée d'un produit n'est pas statique mais seul un instantané est représenté.

7.1.2. Données chaudes

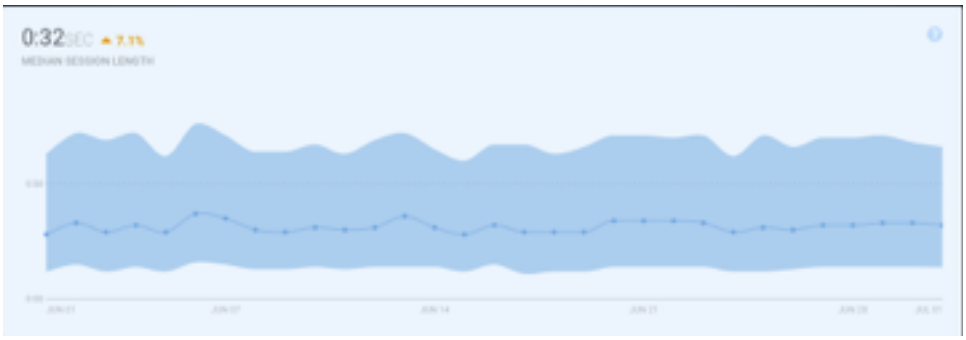
Les données chaudes ont pour propriété d'avoir la notion du temps intégrée à leur nature. Comme la section précédente, deux sous-catégories existent.

En premier, nous retrouvons la donnée de relevée. Ceci correspond à un capteur ou un autre dispositif qui produit une ou plusieurs métriques périodiquement. Sa visualisation varie en fonction du "Pourquoi" dans les questions de visualisation.

- Si seule la dernière valeur est pertinente alors une jauge pourra être utilisée
- Si une fenêtre sur quelques dizaines de périodes est pertinente alors un histogramme ou un graphe en temps réel pour être utilisé.

- Si une fenêtre sur des centaines de périodes est pertinente alors un histogramme, graphe ou un graphe de centiles sera utilisé.
- Dans le cadre où plusieurs périodes sont utilisées, une jauge de dérivée pour montrer l'évolution des métriques peut être intéressante.

Ci-dessous une visualisation de durée de session par les outils de fabric.io avec une valeure instantanée résultat d'un calcul (médiane, texte brut), une variation sur une période représenté par une jauge de dérivée, et enfin un histogramme de percentile pour représenter son évolution au cours du temps.



Ensuite, nous pouvons remarquer l'émergence d'une donnée qui n'est pas régie par une période comme la relevée : les événements. Cette donnée n'a pas de métrique à proprement parler mais est une donnée sémantique ou à interpréter.

La visualisation d'une telle donnée peut être différente suivante sa nature. Étant donné qu'il n'y a pas de présence de métrique a priori, il n'y aura pas de graphe. Par contre :

- Il est possible de faire une analyse fréquentiste de ce flux de donnée et dans ce cas, nous revenons à de la donnée de relevée.
- Il est possible d'afficher un historique de ce flux sous forme de liste et éventuellement afficher ses antécédents si cette opération est possible.
- Si le flux est interprété. Cet événement correspond à une alerte avec un niveau de gravité.
    - Un encart de notification avec une information résumée ou non peut être affichée.
    - Des alertes appliquées sur les éléments descriptifs du système peut être appliqué (par exemple des icones sur une carte)

7.2. Croisements des dynamiques

L'intérêt et la richesse des dynamiques de données ce sont leur croisement. Dans un paradigme d'interrogation autorisant les requêtes continues, il est possible d'interroger tous les types de dynamiques et la production de la requête aura sa propre dynamique.

Quelques exemples :

- Si une requête instantanée est faite, la donnée sera froide (de type résultat spontané)
- Si une requête de fenêtre est faite, la donnée est froide et stable a priori mais cela reste soumis à interprétation en fonction de la dynamique des données d'entrée.
- Si une requête à flux à cardinalité stable est produite alors nous sommes en présence d'un flux périodique.
    - A contrario, si la cardinalité est instable, alors nous avons des données événementielles

Ainsi, si nous partons d'une donnée stable telle que la localisation d'un capteur fixe. En exécutant la requête qui produit le flux des nouvelles valeurs de cette donnée. Alors nous aurons produit le flux événementiel des opérations de déplacement d'un capteur dans le réseau.

Comme précisé précédemment, ces types de dynamiques sont dépendantes de l'interprétation que l'utilisateur fait de ses données.

7.3. Applications aux exemples de données de Ondeo Systems

Reprenons le cas applicatif majeur qui nous intéresse dans le cadre du projet Waves. Et appliquons notre formalisme pour repérer les données et les visualisations à appliquer

Donnée	Dynamique	Visualisation
Relevés de compteurs	Périodique	Dépendant de la période d'observation : jauge, jauge de dérivée, histogramme temps réel ou graphe historique
Structure du réseau	Stable	Carte descriptive
Caractéristique d'un capteur	Statique	Fiche détaillée, Étiquette
Alerte de fuite	Événementielle	Notification, Icone appliquée sur la carte descriptive, Information résumée
Résultat d'un calcul de corrélation	Spontané	Visualisation exploratoire

La fréquence de relevé n'est pas extrême dans le cadre de cette application ce qui rend certaines visualisations non applicable et en terme d'implémentation cela peut avoir des implications majeures.

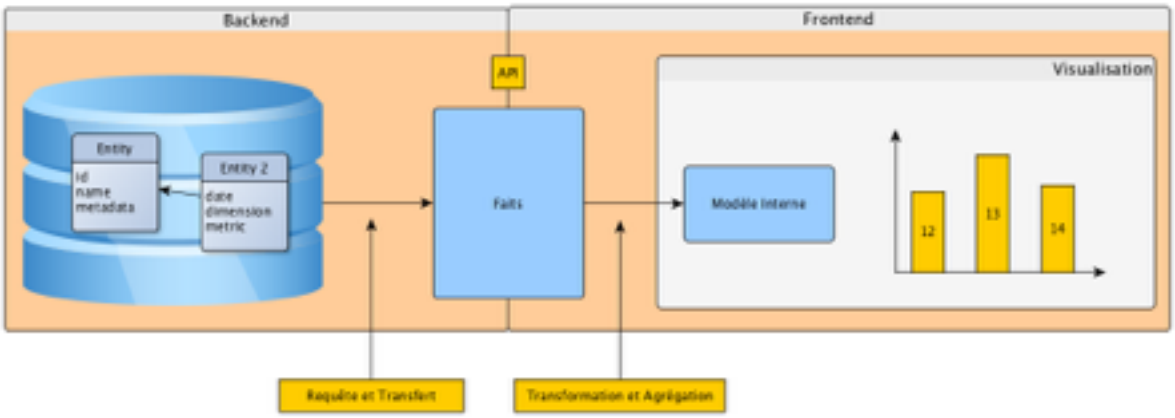
8. Architecture logicielle

Nous avons désormais présenté l'ensemble des aspects théoriques du rapport entre la visualisation et le temps. Nous allons voir les impacts de la dynamique des données sur l'architecture logicielle.

Nous insistons ici sur le faire que la visualisation n'est qu'une projection d'un ensemble de données. Toutefois, cet ensemble de données peut évoluer au fur et à mesure du temps[2]. La première ambigüité à lever vient du terme "ensemble de données" qui reste à préciser.

8.1. Les trois modèles principaux de la visualisation

Dans une application de visualisation de donnée, les données transitent entre trois modèles avant de se retrouver projetés sur l'écran.



Représentation du transit des données jusqu'à la visualisation

Les données sont premièrement dans la base métier qui est structurée selon les besoins du concepteur de l'application. Aucune restriction n'est appliquée ici, l'environnement peut être complexe, relationnel, multidimensionnel ou autre.

Une requête est issue sur le système de base de donnée pour produire une "table" des faits[3]. Dans la pratique cette table peut être constitué de documents plutôt que de n-uplets au sens strict. Ceci constitue l'ensemble des données envoyées à l'interface pour affichage. Les données sont structurées sous forme de liste de points sur lequel l'interface pourra naviguer.

À partir de cet ensemble de faits, nous souhaitons transformer cette table des faits dans la projection voulue pour la visualisation (par exemple, l'agrégation d'une

[2] Très exactement en terme formels, une visualisation est une projection graphique d'une relation temporelle.

[3] Encore une fois, dans le contexte d'un système de gestion de flux de données, ceci peut être assimilé à une relation temporelle et la requête est continue

métrique groupée par une colonne en particulier). Cette étape est la pierre angulaire de l'interactivité de l'application. Cette transformation peut être paramétrable et rend la réactivité de chaque action très efficace vu qu'il n'y a pas d'appel réseau, seul la table des faits est interrogée. La structure du modèle interne dépend entièrement de la visualisation. Par exemple, pour un histogramme, nous aurons un ensemble de point agrégés avec un ensemble de catégories. Il est intéressant de noter que certaines données stables (textuelle par exemple) pourront être collectées soit avec la table des faits soit par un second appel réseau qui a son propre cycle de vie.

8.2. Navigation dans un flux

Lors de l'observation d'un flux, l'ensemble des faits envoyés à l'interface est en réalité une fenêtre sur ce flux. Cette fenêtre se défini dans les formalismes des systèmes de gestions de flux de données. Par exemple :

- S[LAST] = Dernière donnée
- S[RANGE 5min] = Cinq dernières minutes
- S[ROWS 20] = Vingt dernières données

Toutefois, la sémantique des fenêtres peut être très complexe puisque le taux de rafraîchissement, l'évolution des bornes voire même le fait de pouvoir revenir dans le passé est une possibilité en terme d'usage.

Par exemple : si nous prenons une application de supervision. Lorsqu'un soucis est constaté, il faut remonter l'historique d'un ou plusieurs flux pour trouver l'origine du problème.

Le modèle au sens large du composant est similaire à : $\mathcal{S}(\mathcal{D}(S[W]))$ avec

- *S* un opérateur de flux (*IS* ou *RS* dans le formalisme STREAM)
- *D* un opérateur temporel permettant de figer le temps ou non
- *W* une description de fenêtre

Dans le domaine des flux, mais lorsque l'historique devra être récupéré, il faudra interroger une autre source de donnée (en l'occurrence l'historique) qui pourrait être local au composant.

8.3. Impacts architecturaux des différentes dynamiques

Les différentes dynamiques permettent de mieux appréhender les techniques de rafraîchissement du côté de l'interface.

8.3.1. Données froides

Dans le cadre d'une donnée froide, aucun processus de rafraîchissement de donnée n'est nécessaire, ainsi un simple appel à l'API fournira les données.

8.3.2. Données chaudes

A contrario, une donnée chaude va changer au cours du temps. Suivant les cas d'application il pourra être pertinent de ne jamais mettre à jour la donnée. Mais dans les autres cas, il nous faut avoir un mécanisme de mise à jour.

Dans le cas de données périodiques, un simple rafraîchissement régulier de l'information est suffisant dans la plupart des cas. Dans le cadre d'un flux d'information à faible niveau d'alerte, la récupération d'une fenêtre de temps fixe est suffisante avec une interaction possible pour que l'utilisateur puisse rafraichir les données.

Dans le cadre de données événementielles, il existe deux approches possible :

- Rafraîchissement fréquent : l'interface va récupérer les nouvelles données régulièrement (toutes les cinq secondes par exemple) ce qui donne l'illusion de l'instantanéité du système.
- Pousser les données vers l'interface : le serveur sous son initiative pousse les données demandée à l'interface. Ceci garanti une fraicheur des données et allège le travail du serveur qui peut propager la même requête sur plusieurs interfaces en même temps.

Dans le deuxième cas, de nombreuses solutions existent depuis plusieurs années. Les premières approches (COMET) abusaient du protocole HTTP sans clôturer la connexion pour pousser des messages au fur et à mesure. Depuis la popularisation de HTML5, nous pouvons utiliser le mécanisme des WebSockets qui permettent d'ouvrir une connexion bidirectionnelle entre l'interface et le serveur. Au dessus de ce mécanisme, des approches telles que STOMP permettent de créer des canaux de diffusions d'événements.

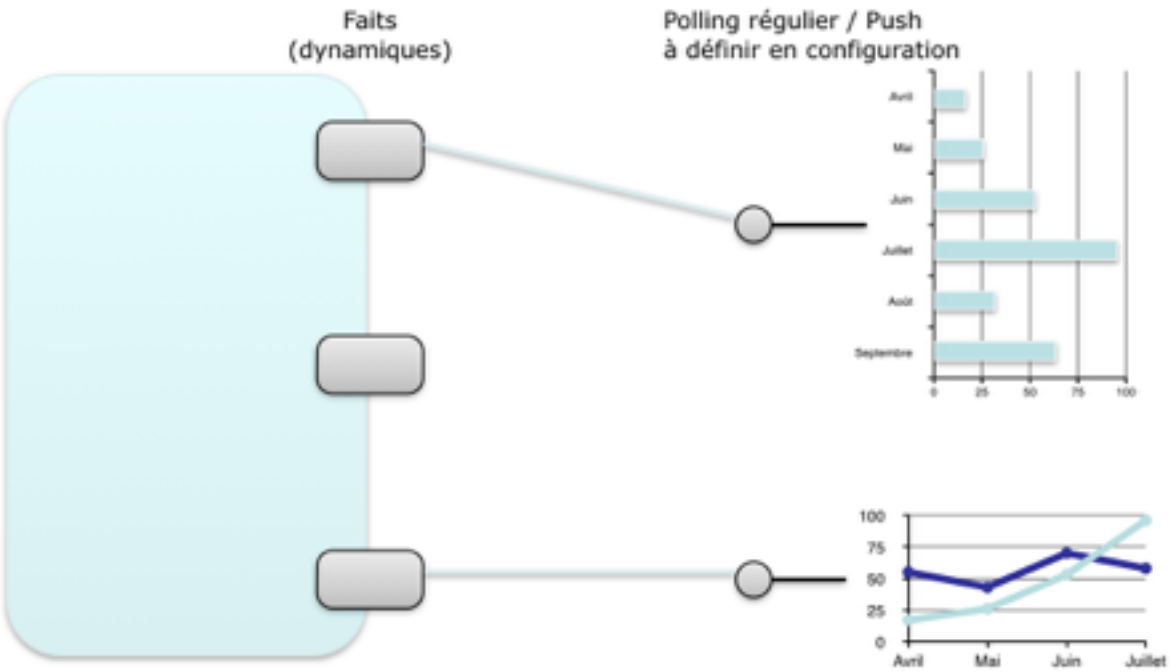
9. Proposition de contribution technique

Nous avons présenté précédemment les aspects théoriques et pratiques à la gestion de données. Quant à la technique, voici ce que nous proposons pour ce projet.

9.1. Boite à outil de visualisation

Nous proposons de développer une boite à outil de visualisation. Le projet Waves a pour but de fournir des accès serveurs à un ensemble de données structurés sous forme de flux. C'est ainsi que l'interface avec le client intervient. Il faut en effet créer un composant qui permet de créer un ensemble de faits capable de gérer les différentes dynamiques.

Une fois ce composant instancié, l'interface peut s'y connecter pour récupérer la donnée et effectuer des transformations vers le modèle interne de visualisation pour finalement afficher le résultat.



Ainsi, nous résolvons plusieurs problème récurrents dans le cadre de la compositions d'applications :

- L'hétérogénéité des modèles
  - Car les modèles sont correctement découplés
- Le transfert de données et le rafraichissement
  - Car cela est gérée de manière transparente entre le composant serveur et le composant client qui sont paramétrables
- La complexité des requêtes
  - Car c'est le projet qui fournit cette capacité

En termes technologique cet ensemble de mécanisme est faisable grâce aux avancés récentes sur les applications web. En effet, des bibliothèques telles que AngularJS permettent de correctement créer des composants ainsi que de composer le cycle de vie de la donnée depuis le serveur vers l'affichage en prenant en compte les différentes interactions possibles (filtres par exemple).

La spécification de l'API et des configurations possibles restent à produire encore.

9.2. Composant serveur

Comme évoqué précédemment, le composant fournissant la table des faits doit être largement personnalisable pour permettre tout types de requêtes, y compris le fait de pouvoir naviguer dans l'historique.

Ses paramètres doivent comporter les fonctionnalités suivantes

- Le mode d'interrogation (persistant ou instantanée)
- Si persistant :

- Le mode de rafraichissement (manuel, périodique, ou par débit)
- Une action simple qui permet d'obtenir l'accès à des données historique

Pour le reste (taille de la fenêtre par exemple) cela est entièrement déterminé par l'expression de la requête dans le langage sur de système de gestion de donnée sous-jacent.

9.3. Tableau de bord composable

Enfin, dans ce projet, nous proposons le fait de pouvoir composer son propre tableau de bord le plus efficacement possible. Nous n'aurons bien entendu pas le temps de préparer un produit aussi puissant que peut l'être d'autres alternatives sur le marché (par exemple Elastic Kibana[4]). Toutefois, nous souhaitons pouvoir créer un tableau de bord par le simple ajout des composants sur une grille prédéfinie en faisant le lien avec les tables de faits correctement synchronisés depuis le serveur.

10. Conclusion

En conclusion, nous avons vu dans ce document que la gestion du temps et de la vélocité avait un impact à l'intérieur de la visualisation de données. Toutefois, grâce à la formalisation des besoins, cet impact reste contrôlable à la fois en termes théorique et pratique. En effet, l'identification des différentes dynamiques de données permet de correctement identifier le problème qui est en partie géré par le système de gestion de flux de données et en partie géré par l'interface graphique.

Nous avons exposé notre proposition de contribution technologique. Plusieurs aspects ont déjà été expérimenté au sein de Data-Publica mais restent actuellement des essais indépendants.

[4] https://www.elastic.co/products/kibana