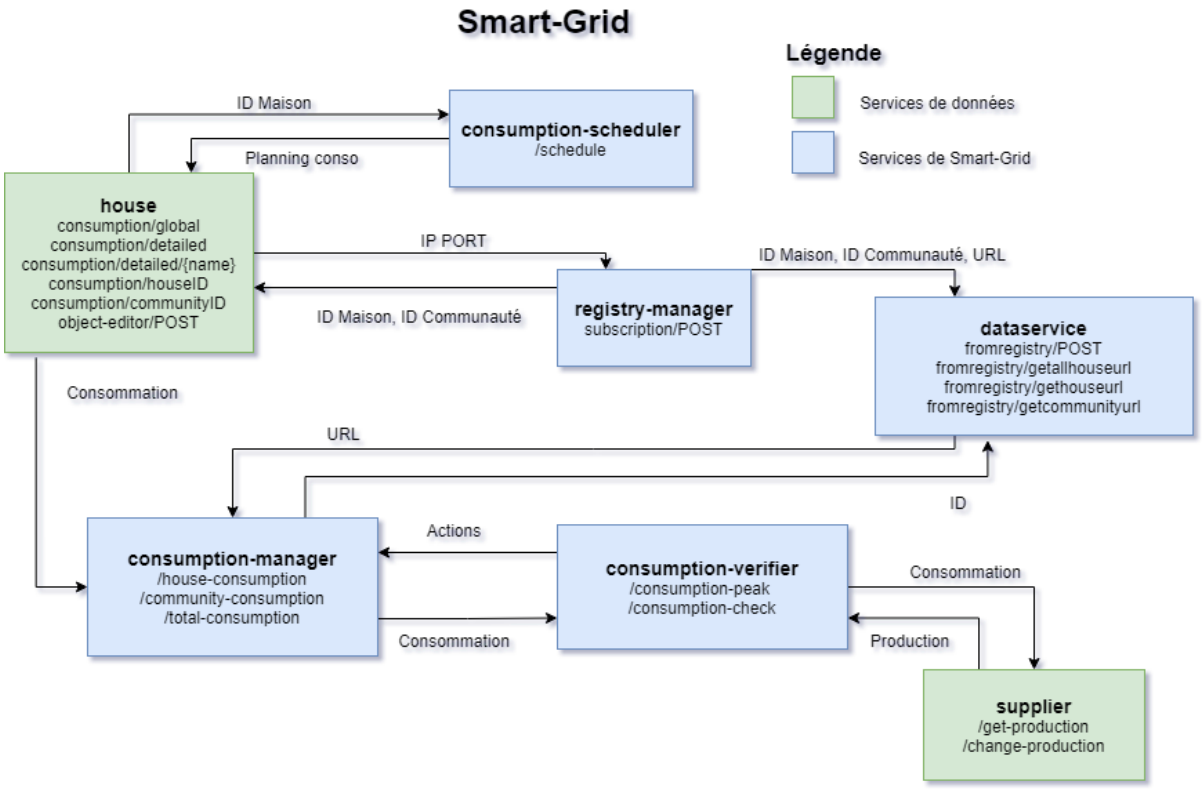
Rapport SOA : Rendu intermédiaire

Valentin Campello, Yann Martin d'Escrienne, Lucie Morant, Yohann Tognetti, Tigran Nerssisian

Etat actuel de notre architecture



Notre architecture se compose en 7 services :

- **house**, représentant la maison et ses objets (voiture, frigo…),
- **consumption-scheduler**, qui permet de gérer la charge d'électricité de la maison et d'ainsi faire des plannings (lorsque par exemple celle-ci branche sa voiture électrique),
- **registry-manager,** qui va enregistrer les nouvelles maisons en leur donnant un ID unique dans dataservice. Il va également leur attribuer un ID de communauté,
- **dataservice**, qui contient toutes les informations sur les maisons clientes de Smartrix Grid (ID des maisons mais aussi des communautés),

- **consumption-manager**, qui récupère la consommation des maisons et/ou communautés et permet ainsi de visionner la consommation des clients. Il sert également pour d'autre services,
- **consumption-verifier**, vérifiant si tout se passe bien dans la grid (la consommation est égale à la production ? Y a t'il un pic de consommation dans une communauté ?) et notifiant le supplier en cas de changement ou de problème,
- **supplier**, qui est le fournisseur d'électricité, donnant la production qu'il fournit et la change si on lui notifie.

Lors de nos premières réflexions sur notre architecture, nous n'avions que trois services (car nous avions voulu rester ultra-MVP pour la première semaine) : house, power-grid et supplier. Or, nous avons remarqué que notre service power-grid avait beaucoup de responsabilités, vu qu'il représentait tout ce qu'est Smartrix Grid, et était donc un composant "dieu". Pour palier à ce problème et aux nouvelles problématiques, nous nous sommes dit que nous devions découper notre power-grid en plusieurs services afin de déléguer toute la responsabilité que power-grid contenait.

Nous avons donc essayé d'identifier tout ce que Smartrix Grid effectuait : la récupération de consommation d'une maison et d'une communauté, des vérifications au niveau de la consommation et de la production, de l'identification des maisons, de leur inscription… et transformer chacune de ces fonctionnalités en service pour finalement nous retrouver avec le diagramme ci-dessus.

Cette architecture nous semble donc cohérente car aucun des services n'a trop d'appels vers lui et tout est équilibré pour que chacun ait son propre métier. Elle est également plus ouverte à des changements ou ajouts de fonctionnalités.

Scénarios créés

Lors de ce rendu, nous avons créé plusieurs scénarios, présentés ci-dessous, en plus des user stories du sujet. Nous avons choisi ceux-ci car ils nous semblent pertinents et en adéquation avec ce que l'on attend de notre projet; ce sont des actions "basiques" et essentielles au bon fonctionnement de Smartrix Grid. Nous pensons également que ce sont de bons scénarios car ils passent dans tous nos services et démontrent ainsi une bonne communication entre nos services. Seuls les scénarios principaux sont montrés dans notre MVP. Les scénarios secondaires démontrent des cas plus précis d'utilisation qui sont couverts de manière plus globale dans les principaux.

Scénarios principaux

A. Inscription et consommation

1. J'inscris ma maison à SmartGrid
2. Ma maison reçoit un ID et est enregistrée dans une communautée avec un autre ID
3. Les informations de ma maison sont sauvegardées dans le système de gestion de donnée de SmartGrid
4. Je peux consulter la consommation de ma maison

Ce scénario permet de vérifier l'inscription d'une maison auprès de registry-manager. Elle permet également de vérifier qu'un ID unique lui est bien fourni et qu'elle est ajoutée dans une

communauté. Les données sont bien ajoutées dans dataservice car on y accède. On peut enfin montrer que l'on peut accéder à la consommation de la maison de façon interne.

B. Charge de la voiture, scheduler, monitoring par SmartGrid

1. Je connecte une voiture dans ma maison
2. On demande un planning de charge à SmartGrid
3. Une plage horaire est renvoyée et affectée à la voiture
4. Depuis SmartGrid, avec L'ID de la maison, je regarde la consommation à l'horaire de charge planifié pour la voiture. La maison renvoie sa consommation et on peut voir que la voiture est en train de charger.

Ce scénario permet de montrer que l'on peut ajouter des objets paramétrables à une maison et de leur affecter un plage horaire "calculée" par consumption-scheduler. On montre également qu'il est possible d'accéder au planning d'un objet. Enfin, depuis consumption-manager, on démontre que l'on peut récupérer la consommation de la maison en donnant son ID et que l'objet paramétrable est bien actif dans la période donnée.

C. Changement de consommation et adaptation

1. On monitore la consommation et récupère la consommation totale de la grille depuis SmartGrid
2. On la compare à la production du fournisseur , celle-ci est égale à la production
3. Un objet est branché/débranché du réseau
4. On vérifie de nouveau la consommation globale de la communauté et on remarque que la production ne correspond plus à la consommation
5. On demande au fournisseur d'adapter sa production à la nouvelle consommation
6. On constate que la production s'est bien adaptée

Ce scénario montre un cas où la production s'adapte à la consommation. On constate que l'on peut récupérer la consommation de toute la grille depuis consumption-manager et vérifier que la production est égale à la consommation à l'aide de consumption-verifier. Ce scénario démontre également que consumption-verifier peut demander à supplier de changer sa production si celle-ci est incorrecte.

D. Pic de consommation dans une communauté

1. On monitore la consommation globale d'une communauté en donnant son ID depuis SmartGrid
2. On regarde s'il n'y a pas de pic de consommation dans la communauté, Il n'y en a pas
3. Un objet gourmand en énergie est branché sur le réseau
4. On regarde s'il n'y a toujours pas de pic de consommation dans la communauté. On constate qu'il y a maintenant un pic de consommation.

Ce scénario montre que depuis la consumption-verifier, il est possible de constater un pic de consommation dans une communauté. On voit aussi que consumption-manager permet de

récupérer la consommation de toute une communauté. A travers le besoin d'augmenter la production, on remarque qu'il est possible d'ajouter des objets sur le service house même si ce n'est pas le but de ce scénario.

Scénarios secondaires

E. Chargement d'une voiture pendant la nuit

1. On souhaite faire charger une voiture durant la nuit
2. On ajoute une planning de consommation pour la voiture dans la période voulue
3. On regarde la consommation de la maison dans la nuit et on constate que la voiture est bien entrain de charger

F. Monitoring de la consommation totale et d'une communauté

1. On monitore la consommation globale d'une communauté en donnant son ID depuis SmartGrid
2. On récupère la consommation de la communauté afin de s'en servir pour des besoins extérieurs ou effectuer ensuite une action sur ce secteur.
3. On souhaite vérifier le poid de cette communauté sur la consommation totale
4. On récupère la consommation totale de grille afin d'effectuer ensuite un calcul statistiques

Notre compréhension du sujet et explication de nos choix de conception

Durant la première lecture du sujet, l'objectif concret de Smart Grid nous a quelque peu échappé. Cependant les user-stories nous ont permis de mieux cerner les premiers besoins du client. D'un côté, il faut offrir un service de visualisation de la consommation et des différents objets de la maison d'un client. Il faut également pouvoir en ajouter et pouvoir paramétrer leur temps de consommation. D'un autre côté, il faut donner la possibilité de manager la consommation des clients par maison ou bien par communauté et de fournir des plannings de consommation pour les objets paramétrables consommant sur certaines plages horaires. Enfin, il faut pouvoir communiquer aux producteurs d'électricité la consommation sur toute la grille afin qu'ils puissent s'adapter et prévenir d'éventuels pics de consommation.

L'ajout d'user stories au cours des semaines nous demande de revoir notre architecture et d'adapter nos communications entre services pour pouvoir être dans les temps tout en intégrant les nouvelles fonctionnalités. Par exemple, pour cette semaine, nous avons dû découper notre "power-grid", qui était le service se situant entre la maison et le fournisseur d'électricité. Nous nous étions rendu compte qu'il avait trop de responsabilité et que cela était mauvais d'un point de vue architectural et fonctionnel vu que cela pouvait nous faire un single point of failure. La semaine d'avant, c'était le service horloge que nous n'avions pas choisi d'implémenter (mais plutôt de passer la date dans les requêtes que nous faisions) par souci de temps. En effet, en l'absence de base de données dans l'architecture actuelle, il n'est pas possible de stocker périodiquement la

consommation des maisons. Nous avons donc choisi de passer en paramètre une date qui permet à la maison de dire quels objets sont en cours de consommation à ce moment.

Nous avons fait le choix de ne pas créer de service pour une communauté de maison, qui aurait pu nous simplifier la gestion et la récupération des données mais cela n'illustre pas un cas où de nombreuses maisons sont présentes. En effet, il serait alors très lourd d'ajouter un nouveau service à chaque fois qu'une maison isolée géographiquement des autres vient de rejoindre SmartGrid. Nous avons donc décidé de passer simplement par un ID de communauté retenu par la maison et notre service de données.

Ainsi, de notre point de vue, le sujet cherche à nous pousser à faire les choix qui nous semblent les meilleurs pour arriver à un produit minimum viable dans les temps que nous avons.

Contraintes

- Besoin de précision sur la différence de production/consommation et de réactivité pour prévenir le producteur en cas de différentiel.
- Les maisons doivent toujours pouvoir accéder à l'électricité demandée, sauf lors de cas spéciaux où les objets branchés peuvent attendre et demander un planning de consommation (voitures électriques par exemple).
- Les clients doivent être capables d'accéder facilement à leur consommation sans forcément passer par des services de SmartGrid.
- Les maisons doivent être regroupées en communautés mais, lors de l'enregistrement d'une maison, celle-ci ne doit pas être bloquée par des problèmes internes à smartgrid. Elle doit pouvoir être ajoutée rapidement dans une communauté.
- Les communautés doivent être monitorées de façon à ce qu'il n'y ait pas de pic de consommation.

Choix de simplifications

- Dans notre projet, les producteurs d'électricité sont pour le moment une entité monolithique, qui ne possède pas de limite de production, car nous pensons que nous concentrer sur les producteurs n'avait que peu d'intérêt pour les users stories données.
- La date des demandes est passée dans les requêtes, car implémenter une horloge émulée et commune nous permettant de tester différentes périodes en peu de temps était trop compliqué dans la limite de temps que nous avions.
- La consommation de nos maisons est demandée par la smart grid (et non pas envoyée par les maisons à la smartgrid de leur propre chef) pour permettre à celle-ci d'avoir un contrôle sur le flot des données entrantes.
- Les communautés sont créées de façon arbitraire par SmartGrid, et non en fonction de la localisation des maisons. Cela pourra sûrement évoluer dans le futur.
- Pour l'instant, notre scheduler donne des plannings par lui-même de façon arbitraire par manque de temps mais, par la suite, il sera relié au consumption-verifier, qui monitore les communautés, et donnera des plannings en fonction de la consommation de la communauté à laquelle est rattachée la maison.
- Les données des maisons (id/communauté/adresse de communication) sont stockées dans un service nommé dataservice afin de pouvoir accéder à ce que l'on souhaite facilement. Ce service pourra être répliqué et fractionné par région.

- S'il y a un pic de consommation, il sera seulement détecté. Nous ne faisons rien au niveau du supplier. C'est à la charge de la personne gérant l'électricité de décider ce qu'elle souhaite effectuer.
- Les maisons commencent déjà avec une consommation de base. Cela correspond à des appareils électroniques consommant constamment.

Couverture des users stories par notre architecture

Notre architecture permet au système de couvrir toutes les stories selon notre compréhension du sujet. Tout d'abord, nous avons créé notre système à l'aide de celles-ci, il répond ainsi au mieux aux besoins évoqués par les clients.

Pour ce qui est du besoin de Pierre et Marie, il leur est possible de voir la consommation de leur maison (/*consumption/global*) mais également de chacun des objets de celle-ci (/*consumption/detail: {name}*) en donnant la date où ils veulent voir la consommation du foyer.

Pour Nikola, il lui est possible de demander à la production de s'adapter à la consommation totale de la grille à l'aide de consumption-verifier (/*consumption-check*) et de regarder s'il n'y a pas de pic de consommation dans certaines communautés (/*consumption-peak*).

Charles, quant à lui, peut constater la consommation de tous ses clients avec consumption-manager (/*total-consumption*), de la consommation de certaines communautés (/*community-consumption*) ou bien même d'une maison en particulier (/*house-consumption*). Il lui est également possible d'envoyer des plannings de consommation pour les objets paramétrables de ses clients à l'aide de consumption-scheduler quand ceux-ci lui font une demande en envoyant l'ID de leur maison (/*schedule*). Un service d'inscription et un autre de stockage des données des clients permet de faciliter tous les échanges de nos services avec les maisons ou producteurs et de répondre au mieux à des demandes de management des clients.

Enfin, Elon (Musk) peut charger ses véhicules la nuit en affectant directement un planning de consommation à ses objets paramétrables. Il peut le faire sans passer directement par consumption-scheduler.