Université de Reims Champagne-Ardenne

Rapport

Dans le cadre de la matière d'info0503

Jarod NOUZILLE – Lucas BAILLY
12/12/2021

I. Table des matières

II. Présentation globale :

Il nous a été demandé tout au long du semestre de développer un système SCV prenant place dans le cas d'un concessionnaire et de la construction de Voiture pour la vente auprès de client via un Site web. Pour mettre en place ce genre de système, nous avons suivi la modélisation suivante :

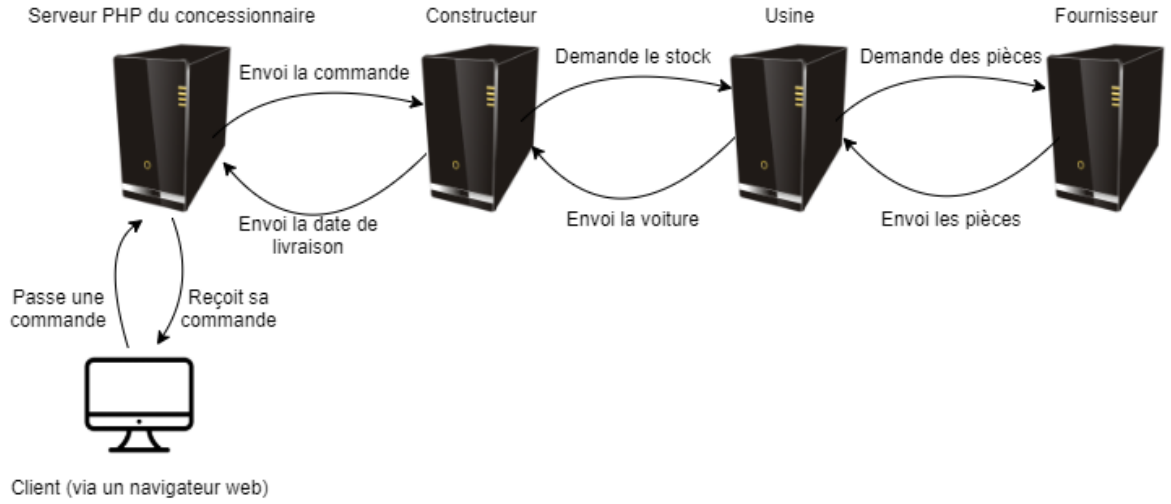

Figure 1: Modélisation des différents acteurs du projet

Afin que le système soit fonctionnel, il vaut mieux démarrer de la fin (donc à partir du Fournisseur si nécessaire ou de l'Usine) puis démarré le serveur d'Usine et le Constructeur. Enfin, avec l'aide d'un serveur PHP tel que Mamp ou Wamp, il sera possible de démarrer la page web du client. Par ailleurs, on utilise des serveurs http différents pour chaque constructeur. Ainsi, selon la marque choisie par le Client, il sera redirigé vers le bon serveur avec ses Handlers respectifs.

III. Les types de communication :

A. Concessionnaire/Constructeur :

Pour la communication entre ces deux acteurs, nous avons opté pour un échange enchaîné. Le concessionnaire enverra au Constructeur la Commande passée par le client sous forme d'un JSONObject. Le constructeur se chargera par la suite d'interroger l'Usine pour connaître son stock et ainsi fournir en retour, au Concessionnaire, une date de livraison approximative voire la voiture si toutefois l'Usine l'a déjà en stock.

Pour résumer, nous avons donc une première requête émise par le concessionnaire qui contient la commande passée par le Client (en format JSON). Cette commande est ensuite interprété par le Constructeur (nous verrons cela dans la partie suivante). Ensuite le Constructeur envoi en réponse au Concessionnaire la Voiture (au format JSON) ou un délai de livraison si la voiture doit être construite par l'Usine. Une fois la réponse envoyée par le Constructeur, la connexion entre les deux peut s'arrêter. Pour vous illustrer les deux cas de figures possibles (le cas où la voiture est déjà en stock dans un des parkings de l'Usine et sa négation) :

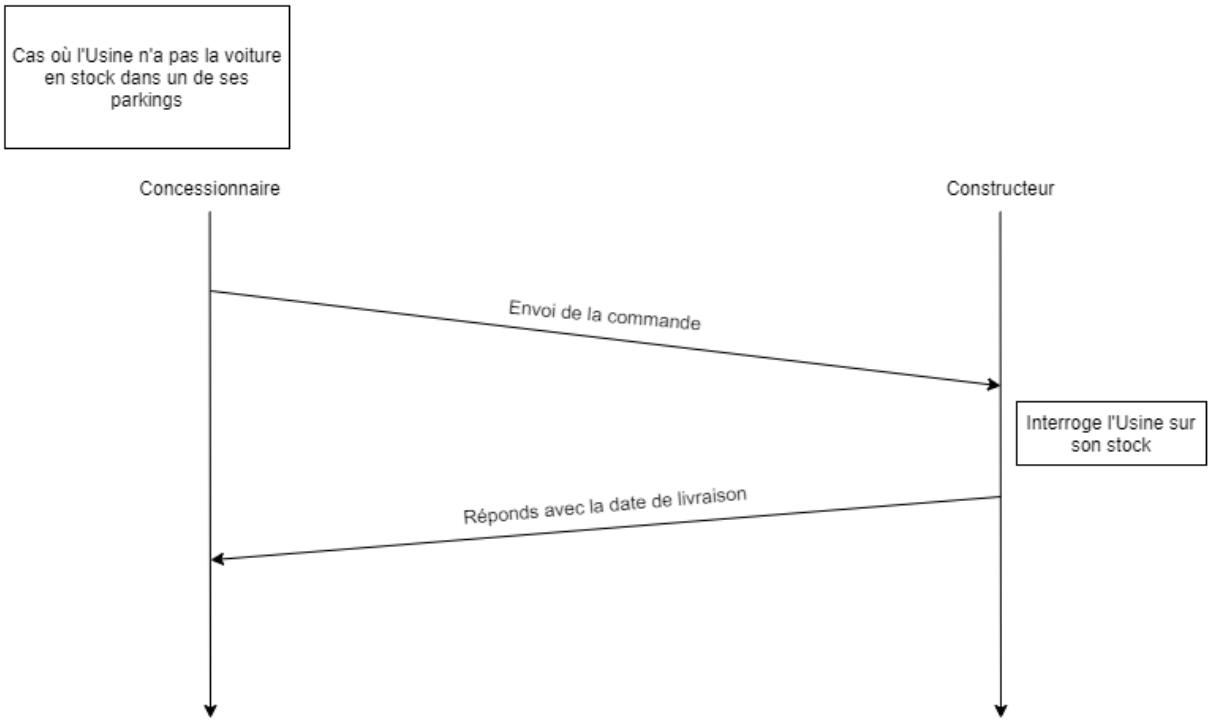

Figure 2: Chronogramme dans le cas où l'Usine ne possède pas la voiture en stock

Dans le cas présenté ci-dessus, l'Usine ne possède pas la voiture dans un de ses parkings et ne l'a donc pas en stock. Ainsi le Constructeur envoi au Concessionnaire la date de livraison selon la date de construction que lui a fourni l'Usine. Ainsi, les requêtes seraient structurées de la manière suivante :

Envoi de la commande {Source, Destination, typeMessage, step, timestamp, Commande (au format JSON)}
Réponse du Constructeur {Source, Destination, typeMessage, step, timestamp, dateLivraison (en string)}

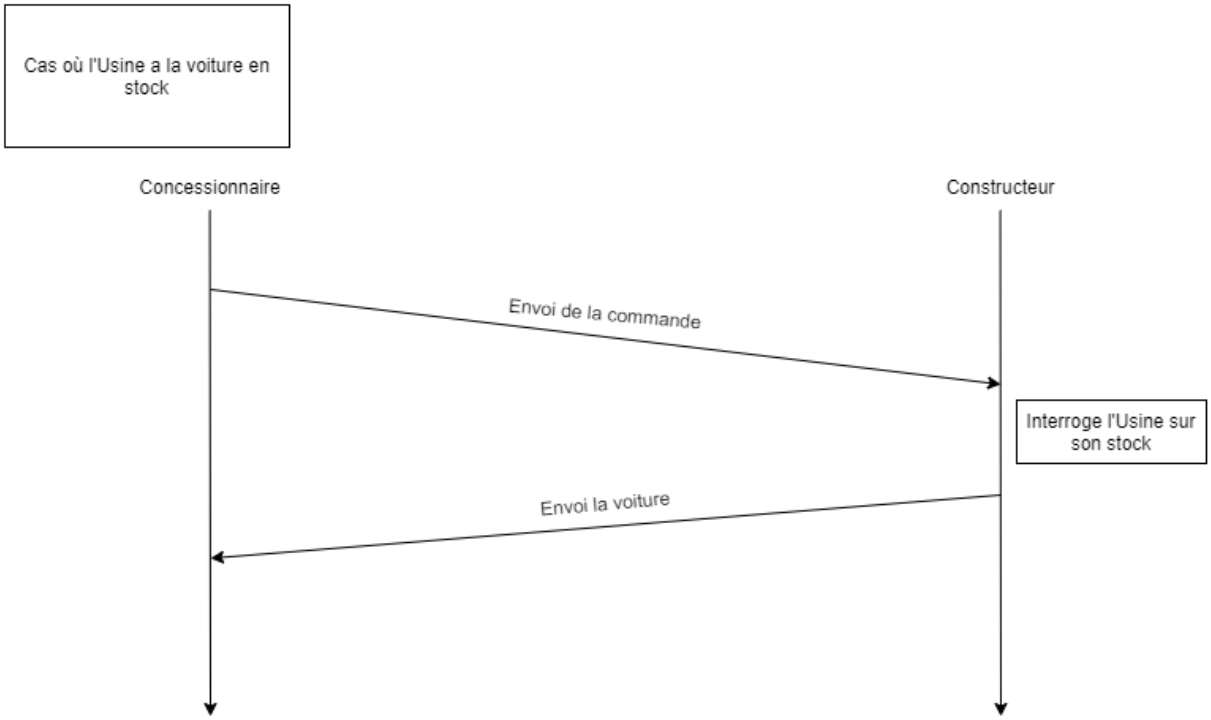

Figure 3: Chronogramme dans le cas où l'Usine possède la voiture en stock

Typiquement, le chronogramme ci-dessus est très similaire au chronogramme précédent. Cependant, ce qui change principalement est la donnée envoyée par le Constructeur au Concessionnaire. En effet, cette fois-ci le constructeur enverra la voiture au Concessionnaire directement car l'Usine lui aura envoyé précédemment vu que ladite voiture est déjà présente dans un de ses parkings. Ainsi, nos requêtes auraient pour structure :

Envoi de la commande {Source, Destination, typeMessage, step, timestamp, Commande (au format JSON)}
Réponse du Constructeur {Source, Destination, typeMessage, step, timestamp, Voiture (au format JSON ou sérialisé en Java selon le scénario)}

B. Constructeur/Usine :

Tout comme la partie précédente, ici nous avons un échange enchainé. Chaque constructeur possède son propre serveur TCP. Le constructeur questionnera l'Usine sur son stock et l'Usine lui répondra « oui » si elle possède déjà la voiture dans un de ses parkings ou « non » dans le cas contraire. Si la première réponse de l'Usine est un « oui » alors le constructeur pourra demander la livraison de la voiture et l'Usine se chargera de lui fournir cette-dernière. Dans le cas contraire, le constructeur demandera une date de livraison approximative à son Usine et celle-ci lui

communiquera la date de construction. Nous distinguons, là encore, deux cas de figure différents que nous allons vous illustrez avec deux chronogrammes différents.

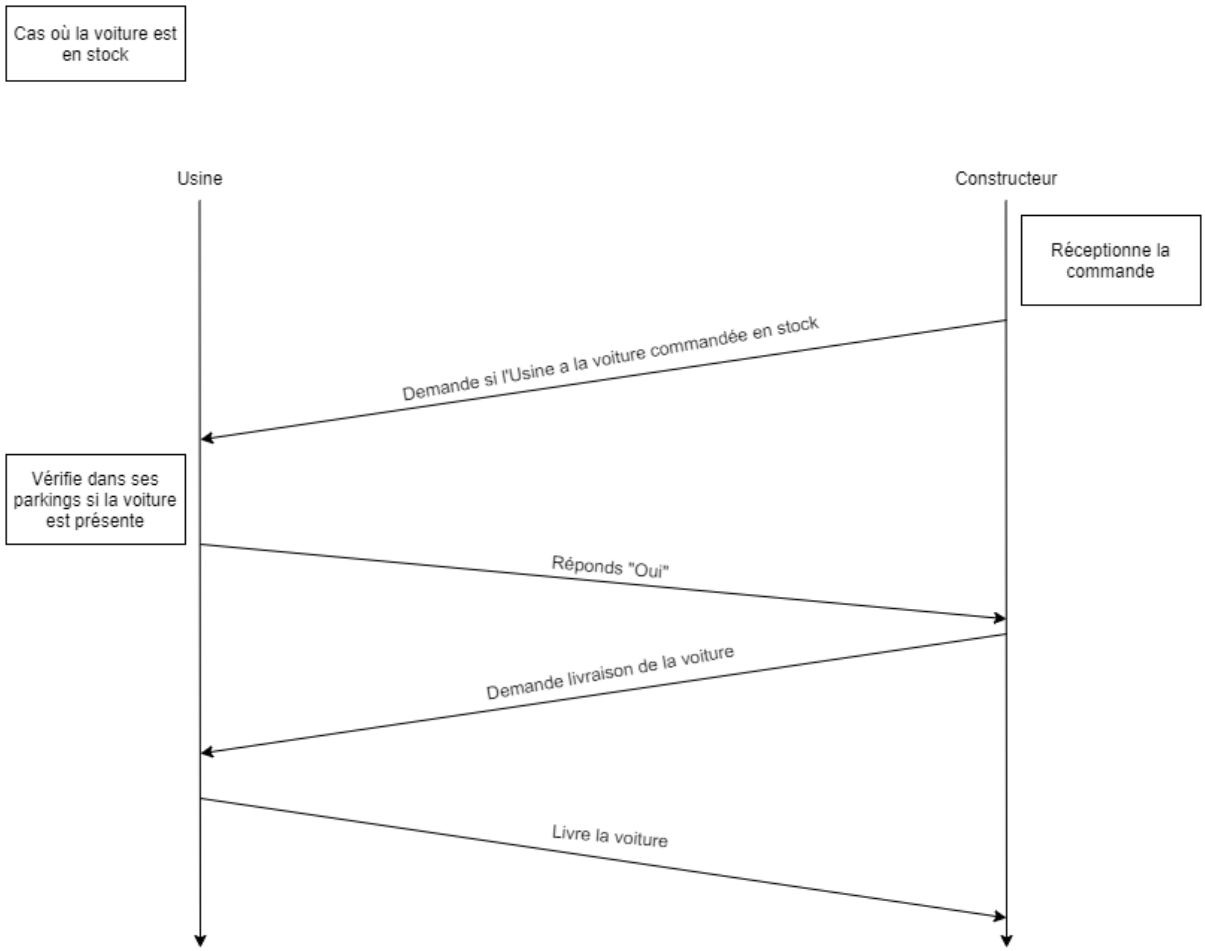

Figure 4: Chronogramme dans le cas où l'Usine a la voiture en stock

Ici nous partons du principe que l'Usine possède bien le véhicule en stock dans un de ses parkings. A la demande du Constructeur, l'Usine répondra alors « Oui ». Le Constructeur demandera ensuite la livraison de la voiture et l'Usine se chargera de lui envoyer. Les requêtes envoyées par les deux acteurs prendront alors la forme suivante :

Demande si l'usine a la voiture en stock {Source, Destination, typeMessage, step, timestamp, Voiture (au format JSON)}
Réponse de l'Usine {Source, Destination, typeMessage, step, timestamp, boolean(true)}
Réponse livraison de la voiture {Source, Destination, typeMessage, step, timestamp, Voiture (au format JSON)}
Réponse de l'Usine {Source, Destination, typeMessage, step, timestamp, Voiture (en JSON ou sérialisée selon le scénario)}

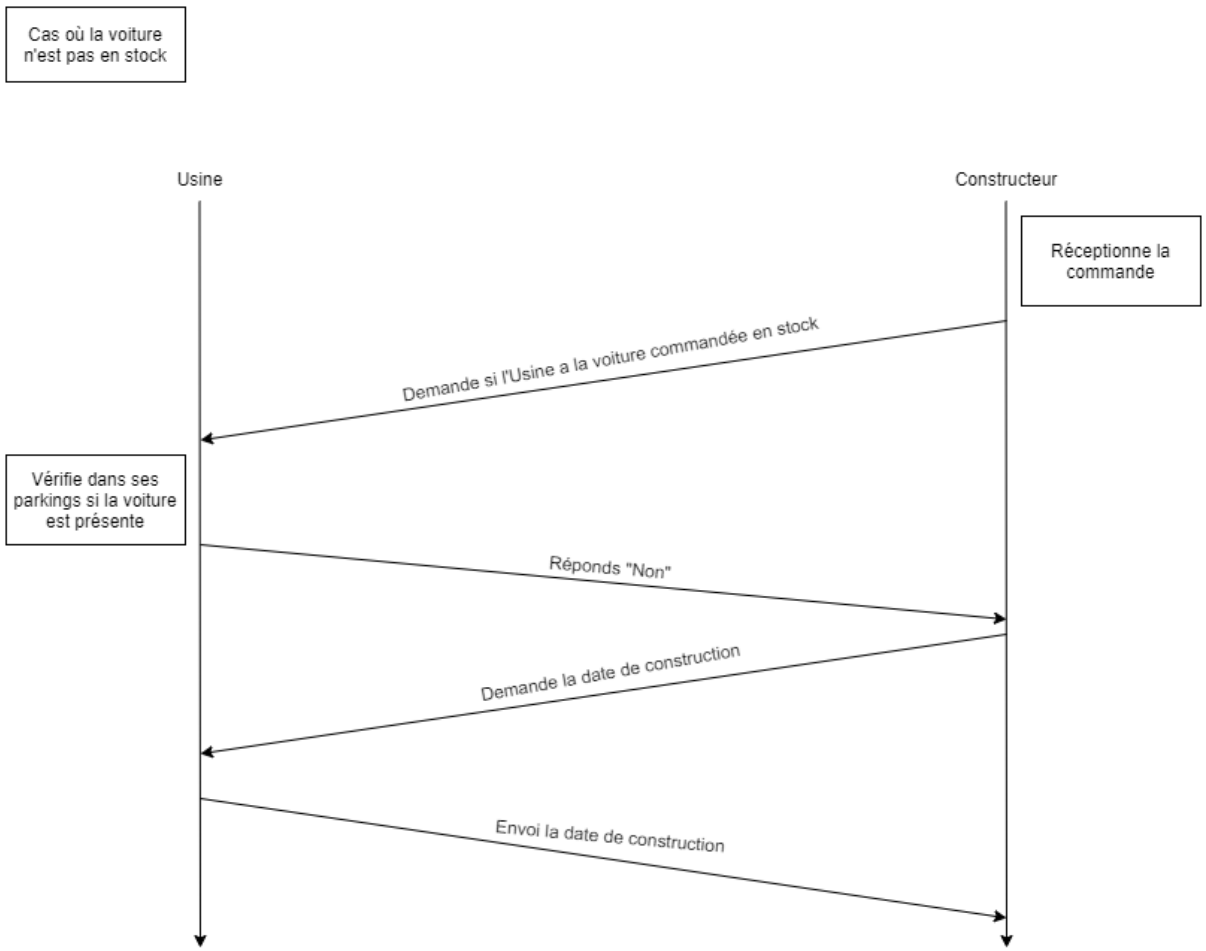

Figure 5: Chronogramme dans le cas où la voiture n'est pas présente dans le stock de l'Usine

Dans ce chronogramme, il n'y a pas beaucoup de différences avec le précédent. La seule différence réside dans le fait que ce qui est renvoyé par l'Usine est différent et de surcroît la seconde requête du Constructeur est aussi différente. Ici notre Usine n'a pas le véhicule en stock dans l'un de ses parkings, elle renvoi donc « non » à la première requête (sous forme d'un booléen valant « false »). A la réception de la réponse émise par Usine, le constructeur demandera une date de fabrication et l'Usine lui répondra en conséquence. La date de construction doit être différente de la date de livraison. En effet, la date de livraison est estimée à J+7 par rapport à la date de construction de la voiture (le temps pour celle-ci d'être livrée par le Livreur). La structure de nos requêtes prendrait alors la forme suivante :

Demande si l'usine a la voiture en stock {Source, Destination, typeMessage, step, timestamp, Voiture (au format JSON)}

Réponse de l'Usine {Source, Destination, typeMessage, step, timestamp, boolean(false)}

Réponse livraison de la voiture {Source, Destination, typeMessage, step, timestamp}

Réponse de l'Usine {Source, Destination, typeMessage, step, timestamp, date de construction}

Vous remarquerez l'absence de contenu dans la seconde requête envoyée par le Constructeur. Cela s'explique par le fait que le Constructeur n'envoi aucune donnée à l'Usine : il s'agit ici d'une demande effectuée par le Constructeur vers l'Usine.

C. Usine/Fournisseur :

Le format de requête choisi pour ce couple est une requête simple. L'Usine demandera à son Fournisseur de lui fournir une pièce en particulier et ce-dernier se chargera de lui envoyer la pièce correspondante. On a choisi de faire une communication UDP pour que l'Usine puisse se démarrer sans forcément que le serveur Fournisseur le soit. En effet, la communication entre l'Usine et le Fournisseur n'est pas obligatoire si l'Usine a déjà construit la voiture ou même si l'Usine a déjà les pièces dont elle a besoin pour la fabrication de la voiture. Cela justifie donc notre choix. Pour illustrer la communication entre les deux acteurs, je vais vous présenter un chronogramme :

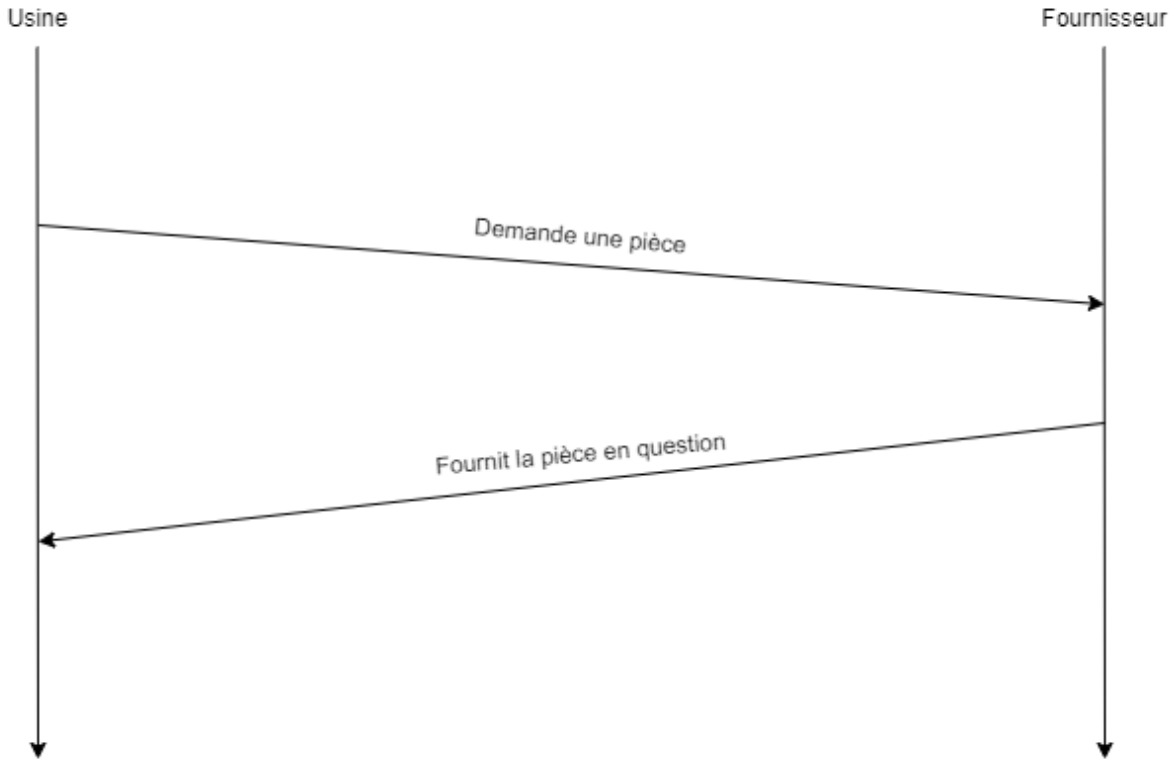

Figure 6: Chronogramme illustrant la communication entre Usine et Fournisseur

Comme expliqué auparavant, l'Usine demande une pièce à un de ses Fournisseurs. Le Fournisseur se charge alors de lui répondre en envoyant la pièce demandée par l'Usine dans sa requête. Le format des requêtes serait donc :

Demande une pièce {Source, Destination, typeMessage, step, timestamp, PièceDemandée (au format String)}
Réponse du Constructeur {Source, Destination, typeMessage, step, timestamp, Pièce (en format JSON)}

La pièce demandée est envoyée en String au serveur du Fournisseur qui se chargera de lui fournir. Pour identifier la pièce voulue, le Fournisseur se charge d'interpréter la

requête pour envoyer la pièce souhaitée par l'Usine. Le Fournisseur envoi, après interprétation de la requête, la pièce correspondante au format JSON et c'est donc à l'Usine de traduire la pièce envoyée en Java.

IV. Les scénarios d'exécution :

A. Scénario A :

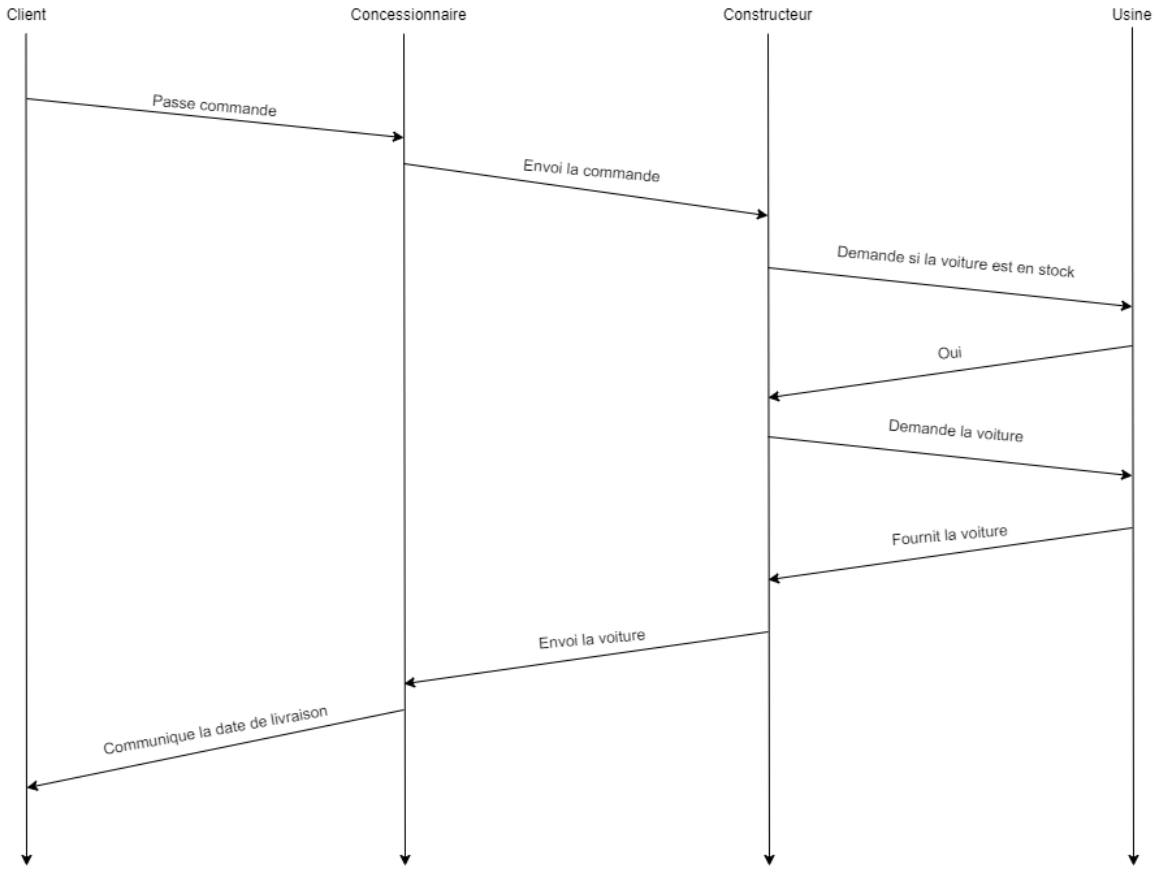

Figure 7: Chronogramme du scénario A

Vous constaterez ici que l'Usine renvoi directement la voiture au Constructeur ce qui n'est pas tout à fait le cas dans la réalité. En effet en réalité, l'Usine fournit un objet Livreur créé à partir d'un patron de conception Factory. Le Livreur se charge de livrer le véhicule au Client. Donc ce que fournit le Concessionnaire au Client est bien la date de livraison mais aussi le code de déchiffrement pour permettre au Client de récupérer sa voiture lors de la livraison. Cependant, comme dans ce scénario il n'y a pas de livraison, nous n'avons donc pas spécifier de Livreur dans notre chronogramme.

Afin de vérifier le bon fonctionnement du scénario A, veuillez démarrer le serveur http lié au constructeur Ford, en exécutant le fichier « ConstructeurFordHttp.java ». Ensuite, veuillez lancer dans l'ordre, le serveur TCP du constructeur Ford puis le client TCP d'une usine qui lui appartient. Pour cela, exécutez d'abord « ConstructeurFordTCP.java » puis « UsineSeveurFordTCP.java ». Ensuite, rendez

vous sur le serveur PHP du concessionnaire par le biais d'un serveur local, en utilisant l'adresse suivante : « http://localhost/SCV_NOUZILLE_BAILLY/SCV/src/ ». Enfin, commandez un véhicule de marque « Ford », de modèle « Focus », de motorisation « V6 », de couleur « VERT » et avec l'option « DIRECTION_ASSISSTEE ».

B. Scénario B :

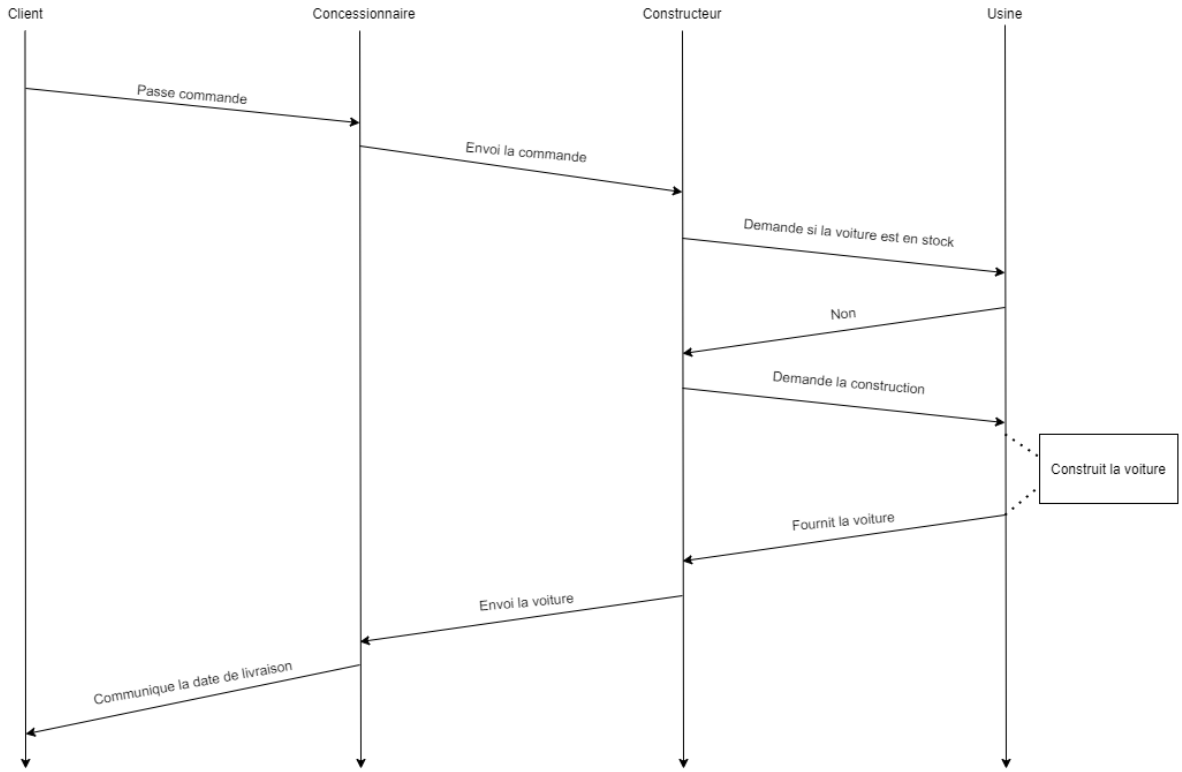

Figure 8: Chronogramme illustrant le scénario B

Le chronogramme ci-dessus est très similaire à celui du scénario A. En effet, la principale différence est la première réponse renvoyée par l'Usine au Constructeur et le fait que l'Usine doit construire la voiture. Comme l'Usine renvoi au constructeur la réponse « Non » à sa première requête, alors le Constructeur demande à l'Usine la construction de la voiture et attends que l'Usine lui envoi la voiture. Dans ce scénario, on part du principe que la voiture est construite instantanément ce qui n'est pas le cas dans la réalité. En réalité, l'Usine devrait fournir une date de fabrication au Constructeur pour que celui-ci la communique au Concessionnaire puis au Client ensuite. Ici aussi, la livraison n'est pas effectuée donc on communique uniquement une date de livraison au Client.

C. Scénario C :

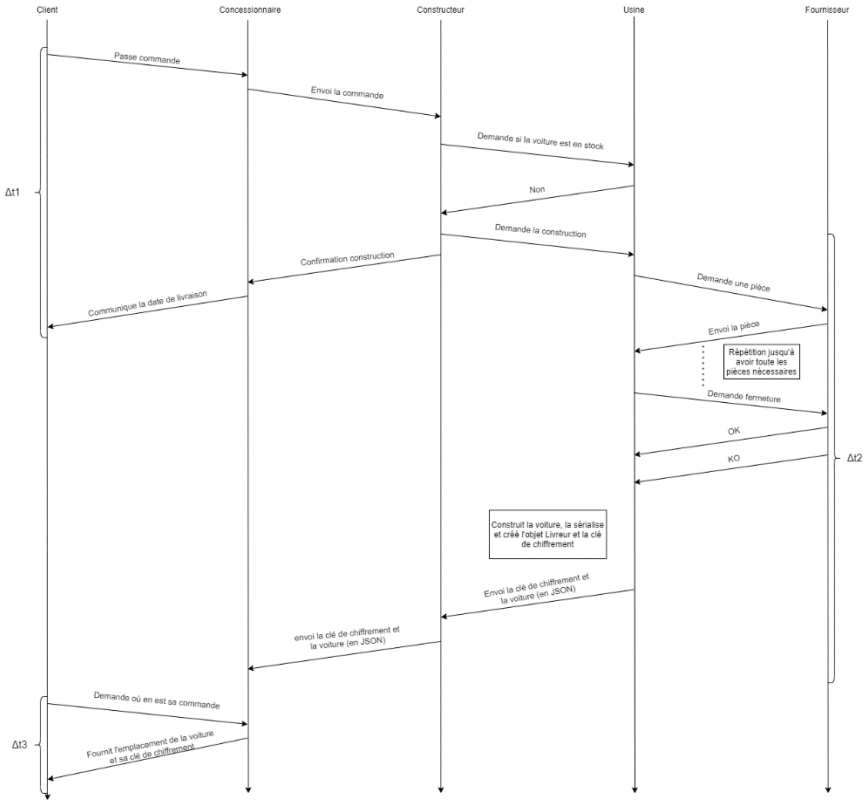

Figure 9: Chronogramme illustrant le scénario C

Dans le cas du scénario C, un objet Livreur est créé par l'Usine une fois la voiture construite. L'usine envoi alors la clé de chiffrement et la voiture (en format JSON). Le constructeur envoi la même chose au concessionnaire. Lorsque le Client demande où en est sa commande, le concessionnaire lui répondra en lui fournissant son emplacement et sa clé de chiffrement pour que Client puisse récupérer sa commande convenablement. La voiture ayant été chiffrée, c'est le seul moyen pour le Client de récupérer sa voiture. Vous constaterez la présence de 3 « Δt », nous avons mis cela en place dans ce chronogramme car les 3 communications prennent un temps différent. Par exemple, « Δt2 » mettra plus de temps que « Δt1 » car les requêtes émises par L'Usine vers le Fournisseur concerne la construction de la voiture tandis que « Δt1 » concerne uniquement le passage de commande par le Client.

V. Conclusion :

Pour conclure nous souhaitons ajouter que ce fut un projet très intéressant. Ce projet nous a permis de voir le fonctionnement de ce genre de système dans un cas réel. Vous constaterez que nous n'avons pas effectué de chiffrement des messages envoyés par manque de temps et que tous les scénarios ne sont pas fonctionnels. De plus, nous avons travaillés à partir de l'IDE IntelliJ, et nous n'avons pas réussis à recréer une architecture fonctionnelle hors de ce dernier. Nous avons fait de notre mieux pour vous fournir un rapport détaillé et un code relativement fonctionnel. Merci pour votre lecture et bon courage avec la correction de tout ceci.