MINISTÈRE
DE L'ÉDUCATION
NATIONALE
ET DE LA JEUNESSE
Liberté
Égalité
Fraternité

EAE SIN 2

SESSION 2024

AGREGATION
CONCOURS EXTERNE

Section : SCIENCES INDUSTRIELLES DE L'INGÉNIEUR

Option : SCIENCES INDUSTRIELLES DE L'INGÉNIEUR ET INGÉNIERIE INFORMATIQUE

MODÉLISATION D'UN SYSTÈME, D'UN PROCÉDÉ OU D'UNE ORGANISATION

Durée : 6 heures

Calculatrice autorisée selon les modalités de la circulaire du 17 juin 2021 publiée au BOEN du 29 juillet 2021.

L'usage de tout ouvrage de référence, de tout dictionnaire et de tout autre matériel électronique est rigoureusement interdit.

Il appartient au candidat de vérifier qu'il a reçu un sujet complet et correspondant à l'épreuve à laquelle il se présente.

Si vous repérez ce qui vous semble être une erreur d'énoncé, vous devez le signaler très lisiblement sur votre copie, en proposer la correction et poursuivre l'épreuve en conséquence. De même, si cela vous conduit à formuler une ou plusieurs hypothèses, vous devez la (ou les) mentionner explicitement.

NB : Conformément au principe d'anonymat, votre copie ne doit comporter aucun signe distinctif, tel que nom, signature, origine, etc. Si le travail qui vous est demandé consiste notamment en la rédaction d'un projet ou d'une note, vous devrez impérativement vous abstenir de la signer ou de l'identifier.
Le fait de rendre une copie blanche est éliminatoire

Tournez la page S.V.P.

INFORMATION AUX CANDIDATS

Vous trouverez ci-après les codes nécessaires vous permettant de compléter les rubriques figurant en en-tête de votre copie.

Ces codes doivent être reportés sur chacune des copies que vous remettrez.

Concours	Section/option	Epreuve	Matière
EAE	1417A	102	2680

Optimisation de la production d'un spécialiste des pâtisseries et produits traiteurs surgelés.

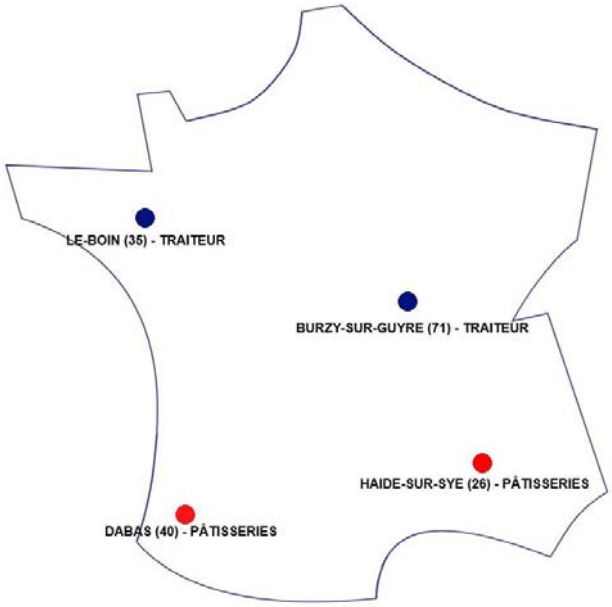
Présentation

Contexte

Une filiale d'un grand groupe de l'agroalimentaire, spécialisée dans les pâtisseries et produits traiteurs surgelés a récemment investi près de dix millions d'euros afin d'augmenter la production d'une de ses usines de fabrication.

L'usine de Dabas (figure 1) est ainsi passée de trois à cinq lignes de production dans l'objectif d'accroitre son flux de 8200 tonnes à 12000 tonnes de pâtisseries par an en cinq ans. Cela n'a pas été le cas.

Comme illustré à la figure 1, la filiale comporte quatre usines réparties dans toute la France et commercialise ses produits à l'international.



Elle propose une large gamme de produits et de format de vente qui évolue en fonction des recherches de son laboratoire de Dabas. Elle peut, également répondre aux besoins ponctuels de ses clients.

Figure 1 - Répartition des usines en France

Application TRS

L'agence locale d'un groupe européen de services digitaux a eu pour mission de mettre en place un tableau de bord temps réel (KPI) associé à une application d'évaluation du TRS (Taux de rendement synthétique).

Le Taux de Rendement Synthétique (TRS) est un indicateur composite mesurant l'occupation d'une ressource de production (machine, ligne, voire atelier de fabrication). C'est un ratio, calculé sous la forme d'un pourcentage de 0 à 100. Lorsque le TRS atteint 100% cela signifie qu'un équipement est entièrement opérationnel. Lorsqu'il atteint 0%, cela signifie qu'un équipement n'a produit aucune pièce bonne.

Une première version de l'application a été conçue, développée et déployée.

Le choix s'est porté sur une application web basée sur une architecture AWS (Amazon Web Services) afin de centraliser la représentation du TRS de l'ensemble des quatre usines sans investissement matériel supplémentaire pour l'entreprise.

L'architecture est représentée à la figure 2.

Les données des automates sont récoltées par une informatique de périphérie (via Greengrass) puis transmises au Cloud à l'aide du protocole MQTT. Elles y sont traitées comme provenant d'IOT.

Des informations sont également récoltées par le biais de tablettes et d'une interface Web. Ces données sont saisies par les conducteurs de ligne. Il s'agit, entre autres, des événements d'arrêts de ligne (pauses, pannes, réglages, etc.).

Enfin, un stockage Cloud de ces données est fait par Amazon S3 (Simple Storage Service) qui permet le stockage bon marché de gros objets et par DynamoDB qui favorise la sécurisation des accès. Athena rend possible l'exploitation des requêtes SQL.

Le document technique DT1 donne un lexique succinct.

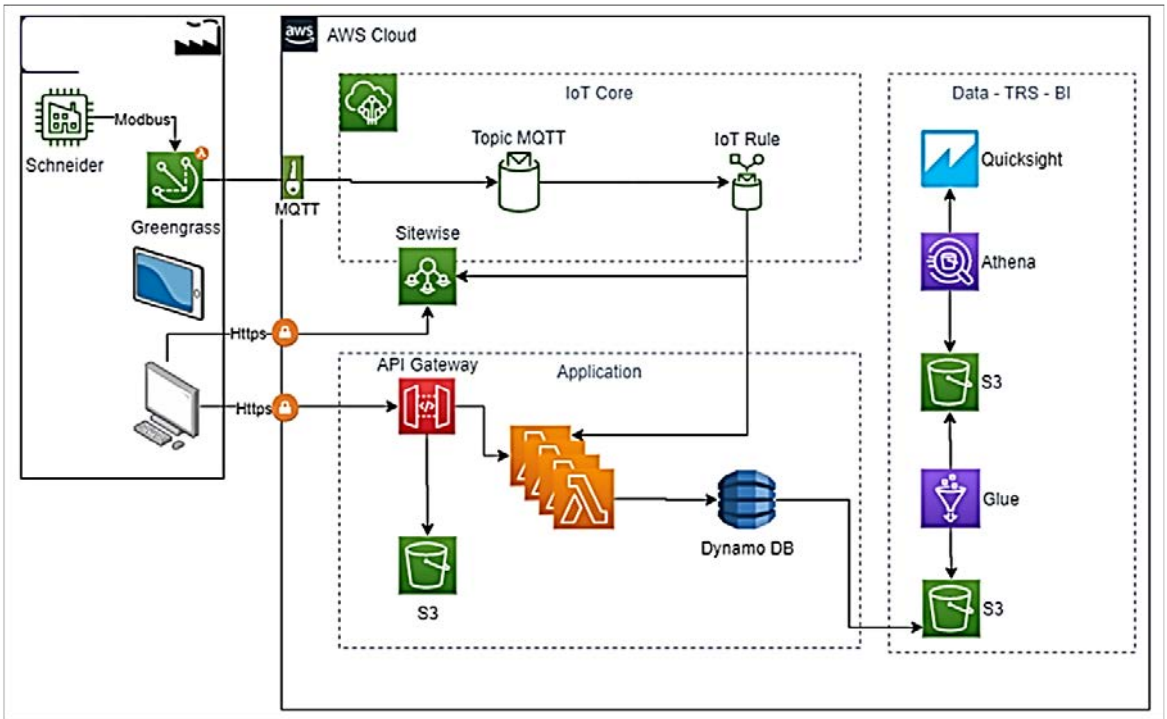


Figure 2 - Architecture de la récolte des données et leur traitement

Un exemple de capture d'écran de l'application est donné figure 3. Cette capture montre un ordre de fabrication en cours. Il s'agit de tartes à la myrtille. Elle fait apparaître le nombre de pièces prévues, le nombre de pièces produites et les arrêts planifiés.

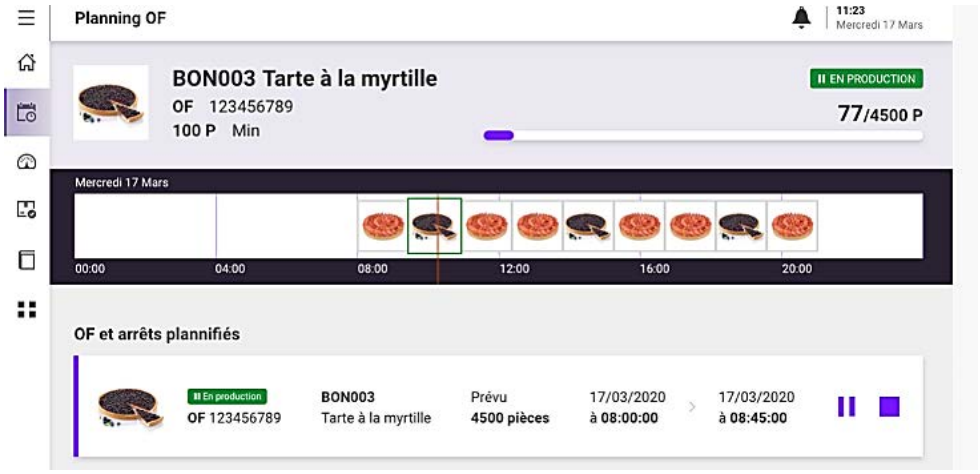


Figure 3 - Capture d'écran OFActifs (Ordres de Fabrications actifs)

Calcul du TRS

La norme NFE 60-182 définit des indicateurs pour les équipements de production. Le TRS (Taux de rendement synthétique) est l'indicateur de performance le plus utilisé.

La meilleure façon de calculer le TRS est basée sur les trois indicateurs pertinents qui sont la disponibilité, le taux de performance et le taux de qualité.

La disponibilité prend en compte tous les événements qui interrompent la production planifiée pendant une durée suffisante pour qu'il soit utile de déterminer la raison de l'arrêt (généralement plusieurs minutes).

La performance prend en compte tout ce qui fait que le processus de fabrication fonctionne à une vitesse inférieure à la vitesse possible lorsqu'il est en marche. Ceci inclut les cycles lents et les petits arrêts. Le taux de performance ne devrait jamais être supérieure à 100 % car cela indiquerait un temps de cycle mal défini.

La qualité correspond au rapport entre le nombre de bons produits et le nombre de produits total. Cela revient à prendre le rapport entre le temps pleinement productif et le temps d'exécution net.

Ces calculs sont illustrés dans le diagramme de la figure 4 dans lequel

- TRS : Taux de Rendement Synthétique
- TRG : Taux de Rendement Global
- TRE : Taux de Rendement Économique

Tournez la page S.V.P.

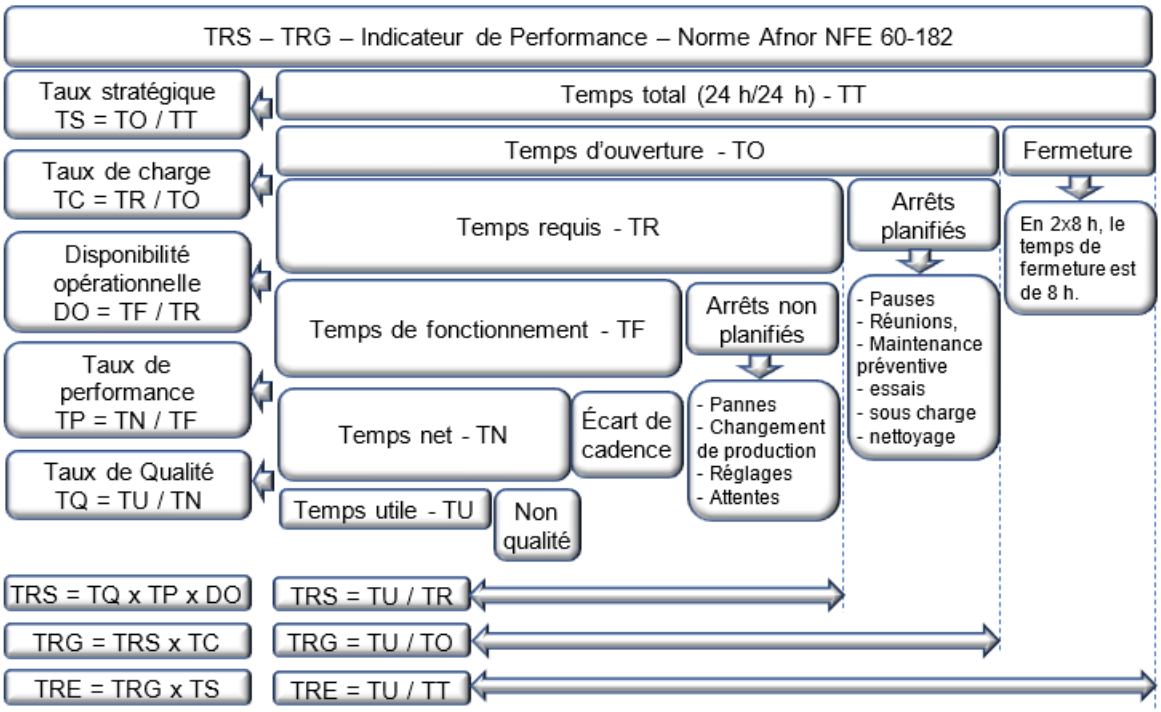


Figure 4 - Indicateurs de performance

Ce sujet va permettre de modéliser différentes sections de l'application afin de vérifier leurs performances en matière de précision des informations obtenues et de qualité de service. Des pistes d'amélioration pourront être envisagées.

Il comporte quatre parties de taille variable :

- Qualification du modèle choisi pour le calcul du TRS

  Les lignes de production de la pâtisserie étant modulaires (adaptables à la production), cette première partie consistera à tenter d'identifier par analyse statistique les différentes équations représentatives du TRS et de la propagation du défaut afin d'en extraire un modèle. Une analyse de la base de données de l'application suivra afin d'être en mesure de proposer des améliorations.

- Estimation de l'efficacité de la récolte des données en matière de transmission

  Cette seconde partie portera sur l'analyse des modules logiciels existant, leur modélisation UML et l'impact des réseaux sur l'efficacité de la récolte des données.

- Amélioration du modèle choisi pour l'identification des défauts en sortie de four

  Dans cette troisième partie, il va s'agir d'analyser et de modéliser le fonctionnement de l'application existante en lien avec la reconnaissance des pâtisseries en sortie de four, de modéliser le refroidissement du gâteau et de proposer une amélioration.

- Impact d’un transfert du Deep Learning vers le Cloud

  Dans cette dernière partie, il va s’agir de modéliser différentes situations de transfert entre les usines et le Cloud afin de vérifier la possibilité de traiter les images dans le Cloud.

Toutes les parties sont indépendantes.

Partie 1. Qualification du modèle choisi pour le calcul du TRS

L’objectif de cette partie est de modéliser la propagation des défauts dans une ligne de production suivant son architecture, d’identifier le modèle qui a été choisi pour la première version de l’application et ses inconvénients par rapport à la flexibilité attendue par l’entreprise. Il sera ensuite possible d’envisager une amélioration.

L’usine de Dabas fonctionne en 3x8, 24 h/24 h. La ligne d’emballage de tartes en 27 cm a une cadence nominale d’une palette par quart d’heure. Une palette contient 96 cartons de 2 tartes chacun.

Un mercredi, la ligne est planifiée pour fonctionner de 5h00 à 17h30.

Le temps de prendre connaissance de la production à venir, de régler les tapis, les rayons X et les bonnes étiquettes pour le bon client, le conducteur lance la production à 5h30.

Au cours de la journée, différents arrêts sont enregistrés.

La ligne a une capacité de production théorique de 8064 tartes mais elle n’en produit que 7580 dont 480 partent au rebut.

Q1. Calculer TU et TR puis en déduire le TRS.

La performance d’un système de production dépend de plusieurs facteurs. Elle dépend du mode d’exploitation, du degré de qualification du personnel exploitant et de la complexité du système, dont la structure peut prendre des configurations allant du simple (série, parallèle, expansion ou assemblage) au complexe (combinaison de certaines des configurations de base).

L’efficience représente le bon rapport entre les résultats obtenus et les moyens mis en place. Elle correspond à la capacité d'atteindre des objectifs avec le moins de ressources possibles**.** Par conséquent, être efficient implique une optimisation qui est la finalité de l’application.

La figure 5 représente un exemple de ligne de production et un modèle sous la forme d’un diagramme de bloc interne. Cette configuration permet de toujours disposer de constituants (pâte brisée, farce, etc.) frais pour l’assemblage des pâtisseries.

Q2. Nommer la configuration de cette ligne de production (figure 5). Argumenter.

Tournez la page S.V.P.

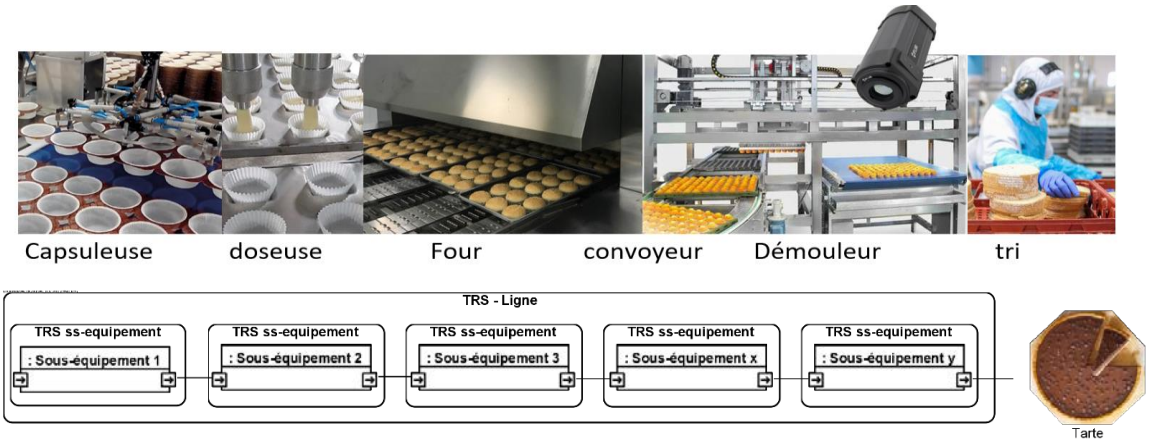


Figure 5 - Exemple de ligne de production de l'usine de Dabas.

Le taux de rendement synthétique (TRS) conventionnel tel que défini par Nakajima en 1988 permet de calculer la performance individuelle de chaque équipement d'un système. La notion de TRS global (TRS_G) a donc été introduite pour permettre l'évaluation des contributions individuelles des différents éléments constitutifs des systèmes de base (série, parallèle, expansion et assemblage). Le TRS_G peut être évalué sous deux aspects : la productivité et la sûreté de fonctionnement. L'un met l'accent sur la quantité de produit réalisé, l'autre sur les comportements fonctionnel et dysfonctionnel des équipements mis en œuvre. Le calcul du premier est basé sur l'efficience des différentes composantes que sont : la qualité, la disponibilité opérationnelle et la performance. Le second est basé sur les taux de ces mêmes composantes (Taux de Qualité, Taux de Performance et Disponibilité Opérationnelle).

Point de vue productivité

Dans la suite, les conventions suivantes seront prises :

TRS_i – taux de rendement synthétique du sous-système i

TRS_n – taux de rendement synthétique du sous-système n (le dernier, nommé y sur la ligne la figure 5)

$t_{P_i}^{(th)}$ – taux (fréquence) de production théorique du sous-système i

$t_{P_i}^{(r)}$ – taux (fréquence) de production réel du sous-système i

Q_{eff_j} – Qualité efficiente (pertes associées incluant la non qualité, les reprises, les défauts de fabrication, etc.) du sous-système j

D_{eff} – Disponibilité efficiente (pertes associées incluant les ralentissements, les écarts de cadence, les blocages, etc.)

P_{eff} – Performance efficiente (pertes associées incluant les arrêts induits, les arrêts propres, les pannes, etc.)

T_{CR} – temps de cycle de référence

Le TRS_G (Taux de Rendement Synthétique Global) pour les sous-systèmes (série, parallèle, expansion et assemblage) est basé sur une approche prenant en compte le temps que passe un équipement dans un état inactif (état d'attente).

Les définitions associées sont les suivantes :

$$t_P^{(r)} = \frac{Nombre\ de\ pièces\ réalisées}{Temps\ Net} = \frac{NPR}{TN}$$

$$t_P^{(th)} = \frac{Nombre\ de\ pièces\ théoriquement\ réalisable}{Temps\ Requis} = \frac{NPTR}{TR} = \frac{1}{T_{CR}} \quad (2)$$

Il est rappelé que la ligne a une capacité de production de 8064 tartes.

Q3. Calculer le taux (fréquence) de production théorique de la ligne d'emballage de Dabas.

Dans le modèle de la figure 5, lorsque le sous-système 1 a produit son premier lot de pâtes à tartes, le sous-système 2 se met en fonctionnement pour le traiter. Chaque sous-système se met progressivement en fonctionnement pour transformer le lot de tartes.

À l'inverse, lorsque le sous-système 1 a fini de produire ses pâtes à tartes, le dernier lot de pâtes se propage dans les différents sous-systèmes qui s'arrêtent progressivement jusqu'à la sortie de ligne.

Enfin, le cas où la pâte à tarte produite par le sous-système 1 est détériorée par le sous-système 3 nécessitera que le sous-système 1 produise une pâte à tarte supplémentaire pour atteindre la production demandée.

Dans la suite, seules les deux configurations simples série et parallèle seront modélisées.

La notation $t_{P_S}^{(th)}$ représente le taux de production théorique en sortie d'une ligne série complète pendant le temps requis TR.

« n » correspond au numéro du dernier sous-système (nommé y sur la figure 5).

Q4. Argumenter afin de montrer que $t_{P_S}^{(th)} = min\left\{t_{P_i}^{(th)}\right\}$. Avec i = 1,..,n. (3)

Il est possible d'écrire que, pour un sous-système comme pour la ligne complète :

$$TRS_i = \frac{Nombre\ de\ pièces\ bonnes\ (sous-système\ i)}{Nombre\ de\ pièces\ théoriquement\ réalisables\ (sous-système\ i)} = \frac{NPB_i}{NPTR_i}$$

Tournez la page S.V.P.

La figure 4 implique que, pour chaque sous-système de la ligne, le TRS est donné par l'expression suivante :

$$TRS = D_{eff} \times P_{eff} \times Q_{eff}$$

pour laquelle :

$$Q_{eff} = \frac{Nombre\ de\ pièces\ bonnes}{Nombre\ de\ pièces\ réalisées} = \frac{NPB}{NPR}$$

$$D_{eff} = \frac{TF}{TR}$$

$$P_{eff} = \frac{TN}{TF} \times \frac{t_p^{(r)}}{t_p^{(th)}}$$

Dans le cas du modèle série, il est cohérent d'affirmer que le temps requis par le sous-système 1 pour produire une bonne tarte en sortie de ligne est le même que celui de la ligne complète. Par ailleurs, afin de simplifier le modèle de propagation du défaut dans la ligne, l'hypothèse suivante sera prise :

$$\frac{t_{p_j}^r}{t_{p\ j}^{th}} \approx 1\ pour\ j = i+1\ \text{à}\ n$$

Q5. Montrer qu'il est possible d'écrire que le TRS du système global série peut se mettre sous la forme :

$$TRS_G = \frac{\min\left\{min_{i=1,2,..,n-1}\left\{\mathrm{TRS_i} \cdot t_{P_i}^{(th)} \cdot \prod_{j=i+1}^{n} \mathrm{Q_{eff_j}}\right\}, \mathrm{TRS_n} \cdot \mathrm{t_{P_n}^{(th)}}\right\}}{\min_{i=1,2,..,n}\left\{\mathrm{t_{P_i}^{(th)}}\right\}} \qquad (4)$$

Il convient, maintenant, d'envisager la configuration en parallèle des sous-systèmes, illustrée sur la figure 6.

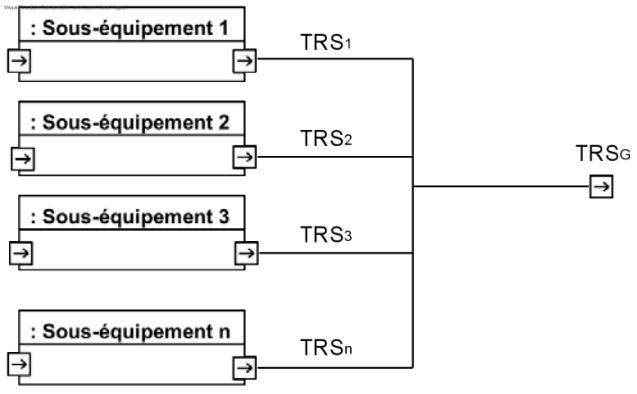


Figure 6 - Ligne de production comportant n éléments en parallèle

Q6. Décrire une situation, liée au marché, qui permettrait de passer l'usine de Dabas en configuration parallèle avec redondance d'un ou plusieurs sous-systèmes.

Q7. En supposant que tous les sous-systèmes fonctionnent, montrer que :

$$TRS_G = \frac{\sum_{i=1}^{n}\left(TRS_i \times t_{p_i}^{(th)}\right)}{\sum_{i=1}^{n} t_{p_i}^{(th)}} \qquad (5)$$

Il est, de la même façon, possible d'obtenir le même type de formule pour les configurations assemblage, expansion ou complexes. Le TRS_G est alors égal à un mélange des deux configurations précédentes.

Sûreté de fonctionnement

Du point de vue « sûreté de fonctionnement », les procédures de calcul permettant de déterminer le TRS global (TRS_G) d'un système redondant dépendent de la réparabilité du système. L'évaluation du TRS_G sera différente selon que les composants sont réparables ou non.

Le temps d'arrêts du sous-système i est noté t_{Ami} et est composé du temps d'arrêt de production, du temps perdu pour non qualité et du temps perdu pour écart de cadence.

Q8. Justifier, par une analyse qualitative de la chaîne série, l'expression suivante :

$$TRS_G = \frac{TR - max_{i=1..n}\{t_{Am_i}\}}{TR} \qquad (6)$$

Il convient, maintenant, de se pencher sur la configuration en parallèle des n sous-systèmes de la figure 6. Tous les sous-systèmes seront supposés en fonctionnement et pourvus de caractéristiques identiques ($t_{p_i}^{(th)}$).

Q9. Montrer, comme précédemment, que l'expression suivante est valide :

$$TRS_G = \frac{n \times TR - \sum_{i=1}^{n} t_{Am_i}}{n \times TR} \qquad (7)$$

Dans une configuration assemblage, la sortie des n sous-systèmes en parallèle est envoyée à un sous-système d'assemblage qui pourrait correspondre à la mise en palettes.

Q10. En nommant TRS_a le TRS du composant assembleur, exprimer le TRS global à partir de l'équation (7) en tenant compte de TRS_a.

Un extrait de fichier SQL d'utilisation de la table OF (Ordre de Fabrication) est fourni en document technique DT2.

Q11. Retrouver la ou les lignes de calcul du TRS et montrer qu'il respecte la définition de la figure 4.

Q12. Donner le modèle (format diagramme de bloc interne) de la ligne de production mis en œuvre dans la première version de l'application et conclure sur le ou les inconvénients de cette solution.

Tournez la page S.V.P.

Un extrait du schéma de la base de données est fourni dans le document réponse DR1.

Q13. Compléter le schéma de la base de données fourni dans le document réponse DR1 en ajoutant les associations et en complétant la table OF_events.

Dans la partie présentation de ce sujet, il a été expliqué la nécessité d'une grande flexibilité de la ligne de production afin de répondre rapidement aux demandes des clients et aux évolutions de la gamme de produits proposée.

Q14. Présenter l'intérêt de concevoir une application qui utilise les modélisations (4), (5), (6) et (7) par rapport à la finalité de l'application.

Choix d'un modèle comportemental pour améliorer le calcul du TRS

L'application est mise en place pendant une durée d'un mois. L'interface permet de visualiser, en temps-réel, l'état des différents sous-systèmes mais aussi d'identifier une certaine périodicité des dysfonctionnements.

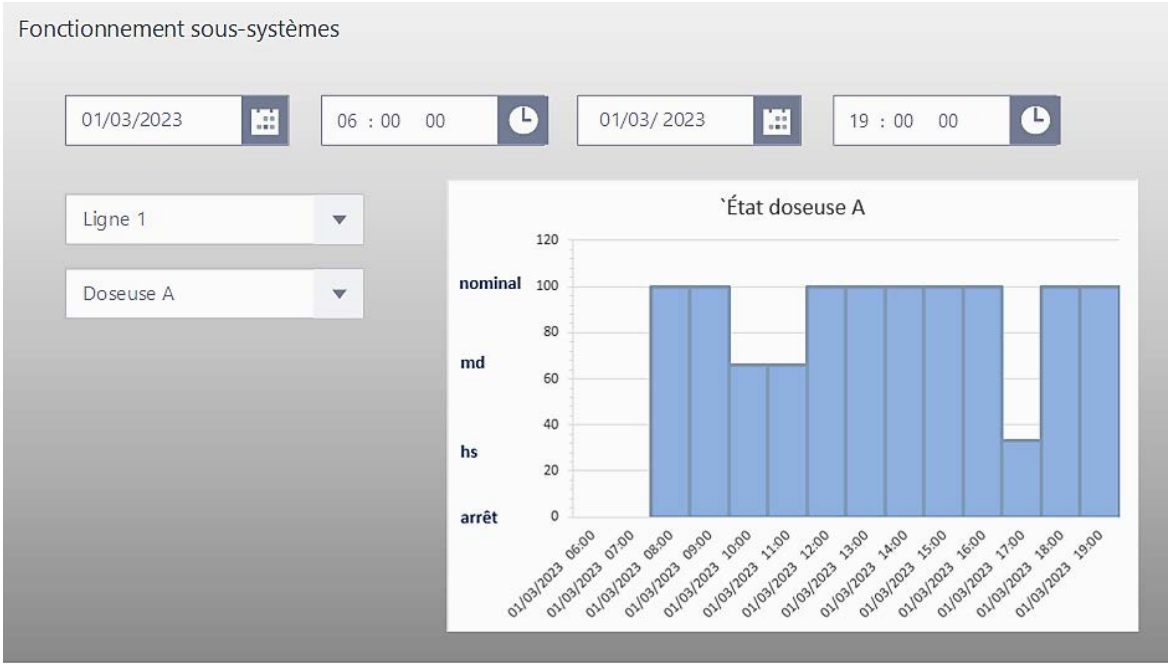


Figure 7 – Capture d'un onglet de l'interface web temps réel

Les calculs précédents étaient basés sur une configuration simple du système de production. Le système réel ayant des capacités de reconfiguration et des redondances, sa modélisation requiert l'utilisation de modèles comportementaux. Il va s'agir, dans la suite, de construire un diagramme d'états-transitions d'une partie du système.

La figure 7 illustre le comportement du sous-système « doseuse A » au cours d'une journée pour un ordre de fabrication lancé à 6h00 et arrêté à 19h00. Deux événements sont survenus conduisant à un fonctionnement en mode dégradé de 10h00 à 12h00 et à un état hors service de 17h00 à 18h00.

Il est possible de considérer trois états de fonctionnement caractéristiques de l'efficience d'un système :

- l'état de fonctionnement nominal (m) pour lequel le TRS est supérieur ou égal à 85 %
- l'état de fonctionnement dégradé (md) pour lequel le TRS est inférieur à 85 % mais supérieur à 25 %
- l'état hors service (hs) pour lequel le TRS est inférieur ou égal à 25 %.

Sur la figure 4, deux temps sont caractéristiques du TRS. Il s'agit de :

- TA qui représente les temps d'arrêt auxquels s'ajoutent les écarts de cadence et les retards dus à la non qualité,

- TR qui représente le temps requis.

Q15. Donner le diagramme d'état-transition d'un sous-système réparable. Les transitions ne devront faire apparaître que les temps TA et TR.

Q16. Donner le nombre d'états nécessaires pour représenter le comportement de deux sous-systèmes série puis des 5 sous-systèmes de la figure 2.

Q17. Identifier, à partir du schéma fourni dans le document réponse DR1, la table et les colonnes qui fournissent les données nécessaires au codage de ce comportement et montrer que cette base de données est suffisante pour situer correctement les failles de chaque ligne de production.

En conclusion, dans le but d'améliorer la productivité globale, il est utile de récolter les données et les informations en lien avec tous les sous-systèmes de la ligne de production comme illustré sur la figure 8. Cela permettra de cibler au mieux la cause d'un mauvais TRS et de reconfigurer, en temps réel, le système de production.

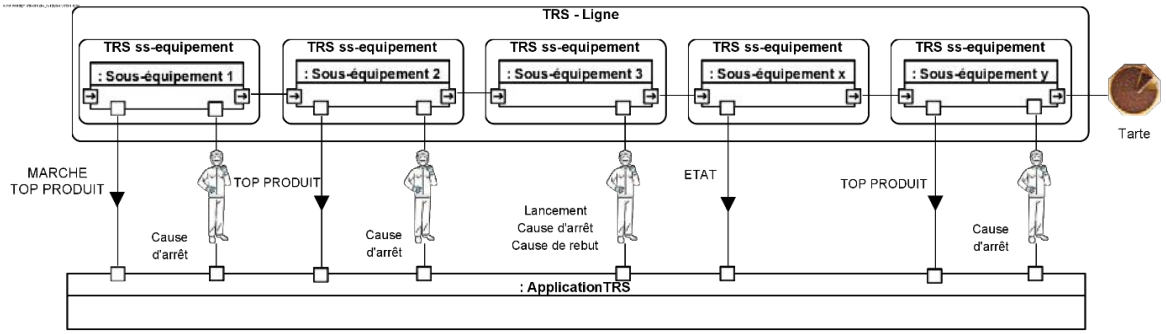


Figure 8 - Récolte des données sur une ligne de production

Partie 2. Estimation de l'efficacité de la récolte des données

L'objectif de cette partie est de modéliser le réseau local et sa partie périphérique afin de vérifier que l'application peut être déployée dans les différents réseaux possibles dans les différentes usines de l'entreprise. En effet, il s'agit de rénovation et non de construction.

L'architecture choisie afin de récolter les données (voir figure 8) de l'ensemble des usines est fournie à la figure 2.

Cette partie porte sur l'aspect local de l'application TRS tel qu'illustré à la figure 9.

Tournez la page S.V.P.

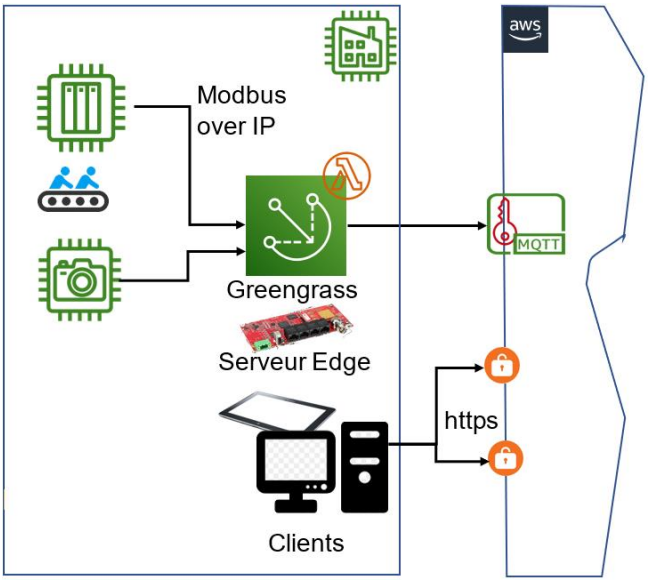


Figure 9 - Partie locale de l'installation

Le document technique DT1 décrit les principaux éléments de la figure 2 et de la figure 9.

Q18. Après analyse, donner le rôle de Greengrass au sein de l'application TRS.

Analyse du programme commun aux quatre sites

Dans un premier temps, il est nécessaire d'analyser comment l'application de récolte des données a été construite.

Le document technique DT3 donne un extrait du fichier de configuration « json » et du programme qui permet de transposer les données de l'automate en MQTT pour envoi vers le Cloud. Il est exécuté sur le serveur Edge de la figure 9 ou informatique en périphérie.

La fonction poll_device(mb_client, device, mqtt_client) permet, pour le client Modbus TCP « mb_client » et le périphérique « device », de :

- lire l'ensemble des registres et bits de ce périphérique avec les méthodes read_coils(adresse, count, unit) et read_holding_registers(addr_from, number, unit) de mb_client,

- transformer les données récoltées en message au format JSON à l'aide de la méthode msg_mqtt(name, value),

- publier vers le serveur MQTT avec la méthode publish(topic, payload) de mqtt_client.

Q19. Analyser l'extrait de programme du document technique DT3 et proposer un comportement du point de vue du logiciel en complétant le diagramme de séquence UML du document réponse DR2.

Q20. Dans le document réponse DR3, proposer un comportement du point de vue du matériel en complétant le diagramme de séquence système.

Analyse de l'impact des réseaux

Étant donné qu'il s'agit d'ajouter une application centralisée au sein d'une usine existante, le déploiement peut se faire dans un réseau préexistant. Ce dernier pourra avoir été l'objet d'une amélioration ou pas. Il impactera donc les performances de l'application.

La figure 10 présente les protocoles utilisés. Le protocole Modbus TCP est utilisé dans le réseau local, à l'intérieur de l'usine, par le serveur Edge ou l'informatique en périphérie, pour collecter des données provenant des automates. Le protocole MQTT est utilisé pour publier les données avant qu'elles ne soient traitées par les services AWS.

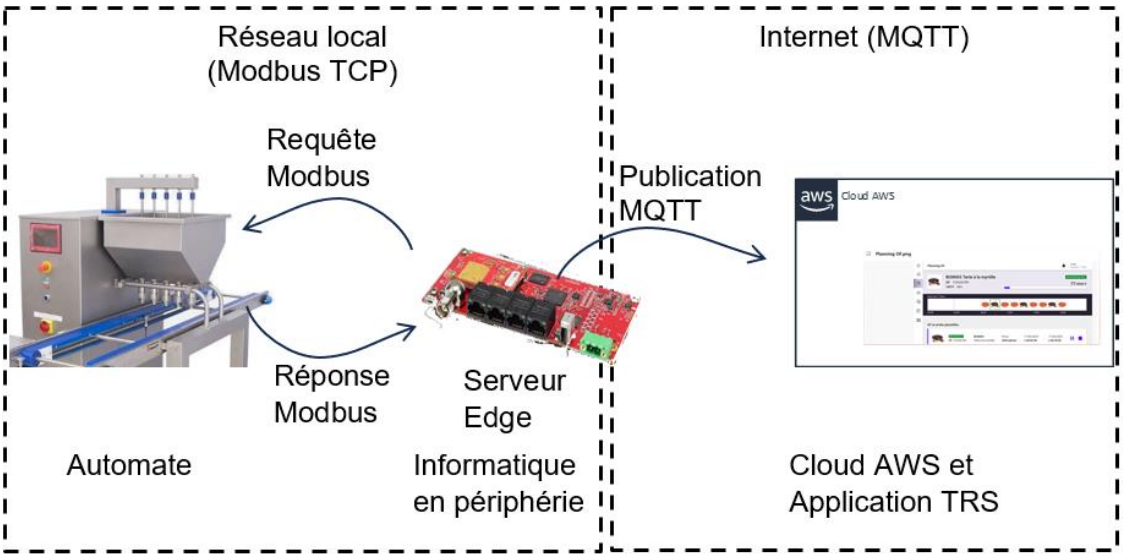


Figure 10 - Protocoles utilisés

Le document technique DT4 donne, sur deux pages, quelques extraits des spécifications Modbus TCP. Elle présente en particulier le format de la trame Ethernet qui encapsule un message Modbus TCP, le format d'une requête ou d'une réponse Modbus (Modbus ADU – Application Data Unit) et le détail de l'entête MBAP.

L'extrait de programme analysé précédemment exploite deux codes fonction : la lecture des « Coils » (niveau bit) et la lecture des « Holding Registers ». La spécification des requêtes et des réponses liées à ces deux fonctions sont données dans le document technique DT4.

Q21. Après analyse des documents techniques DT4 et DT3, calculer la taille de la requête puis celle de la réponse Modbus TCP dans la transaction qui lit les « holding registers ». En déduire la taille de la trame Ethernet qui encapsule la requête puis celle de la réponse. Justifier les 4 résultats.

Q22. Après analyse des documents techniques DT4 et DT3, calculer la taille de la requête puis celle de la réponse Modbus TCP dans la transaction qui lit le « coils ». En déduire la taille de la trame Ethernet qui encapsule la requête puis celle de la réponse. Justifier les 4 résultats.

La figure 11 propose qu'une connexion TCP soit ouverte pour chaque paire requête/réponse.

Tournez la page S.V.P.

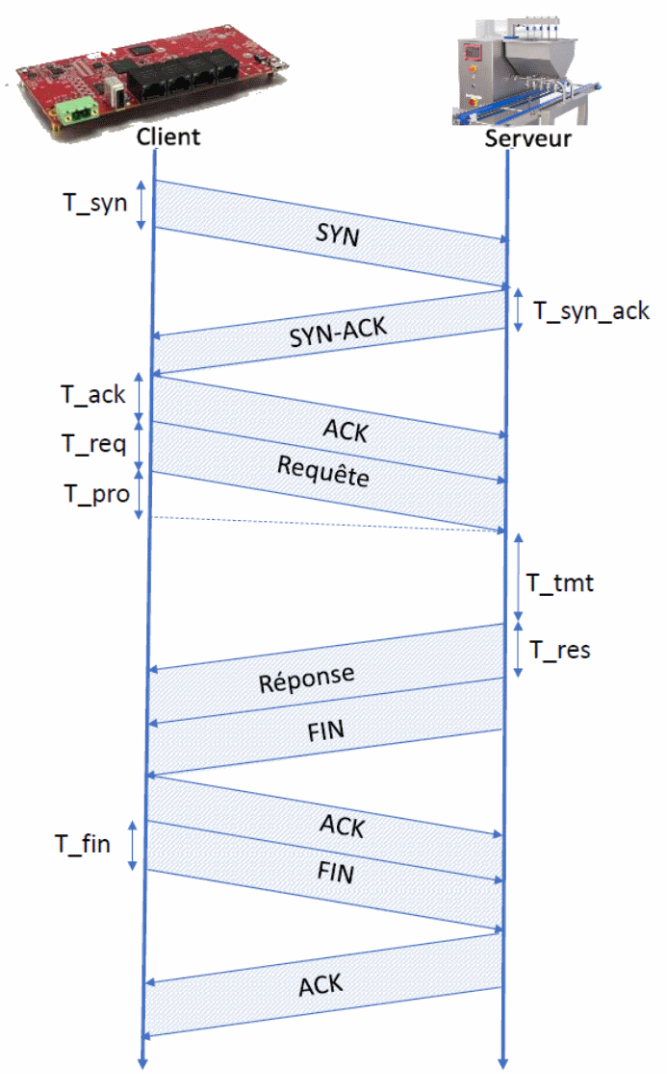


Figure 11 – Détail des échanges TCP par transaction Modbus

Les temps sont notés de cette façon :

- T_req : temps de transmission d'une trame requête Modbus,
- T_pro : temps de propagation entre le client et le serveur,
- T_tmt : temps de traitement d'une requête Modbus par le serveur,
- T_res : temps de transmission d'une trame réponse,
- T_syn : temps de transmission d'une trame de synchronisation TCP SYN,
- T_syn_ack : temps de transmission d'une trame TCP SYN-ACK,
- T_ack : temps de transmission d'une trame TCP ACK,
- T_fin : temps de transmission d'une trame TCP FIN.

Q23. Exprimer le temps nécessaire pour réaliser une collection de données en fonction de T_req, T_pro, T_tmt, T_res, T_syn, T_syn_ack, T_ack, T_fin.

Cas Ethernet full-duplex à 10 Mbit·s^{-1} :

Dans un premier temps, les hypothèses suivantes seront prises :

- l'automate de la figure 10 et le serveur Edge sont interconnectés par une liaison directe Ethernet full-duplex à 10 Mbit·s^{-1},

- la taille des trames (y compris l'entête et le checksum Ethernet) est la suivante : TCP_SYN = 78 octets, TCP_SYN_ACK = TCP_ACK = TCP_FIN = 70 octets,

- le temps moyen de traitement (t_tmt) est de 65 µs,

- la longueur moyenne de la liaison Ethernet est de 100 m et la vitesse de propagation du signal électrique dans le câble est de 200 000 km·s^{-1}.

Il est rappelé qu'un extrait du programme de polling et du fichier de configuration « json » est fourni dans le document technique DT3. Ce dernier détaille les informations à récupérer dans l'automate.

Q24. Calculer la durée d'une connexion Modbus à 1 transaction de type « holding_registers » dans le cas des informations du document technique DT3. Justifier ce résultat.

Il est, maintenant, proposé de considérer qu'une connexion TCP est ouverte pour 2 transactions requête/réponse, comme illustré à la figure 12 de la page suivante.

Q25. Exprimer et calculer la durée de la connexion TCP pour réaliser une transaction « holding registers » et une transaction « read coils ».

Tournez la page S.V.P.

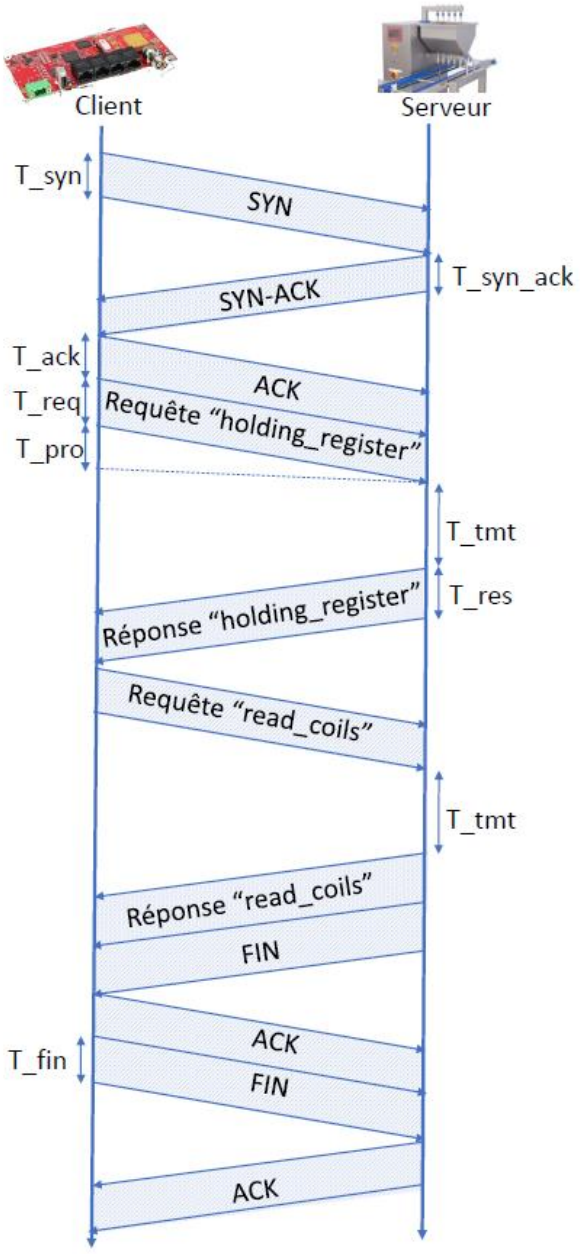


Figure 12 - Connexion TCP pour une transaction "holding register" et une transaction "read coils"

Réseau Ethernet commuté travaillant au débit de 10 Mbit·s^{-1}

Il est, maintenant, proposé de vérifier les performances dans le cas où l'application est intégrée dans un réseau Ethernet commuté travaillant au débit (D) de 10 Mbit·s^{-1}. Les données du paragraphe précédent (taille de trames, distance, temps de traitement) sont toujours valables.

La communication entre les automates et le serveur Edge passe par un commutateur Ethernet. Ceci est illustré sur la figure 13.

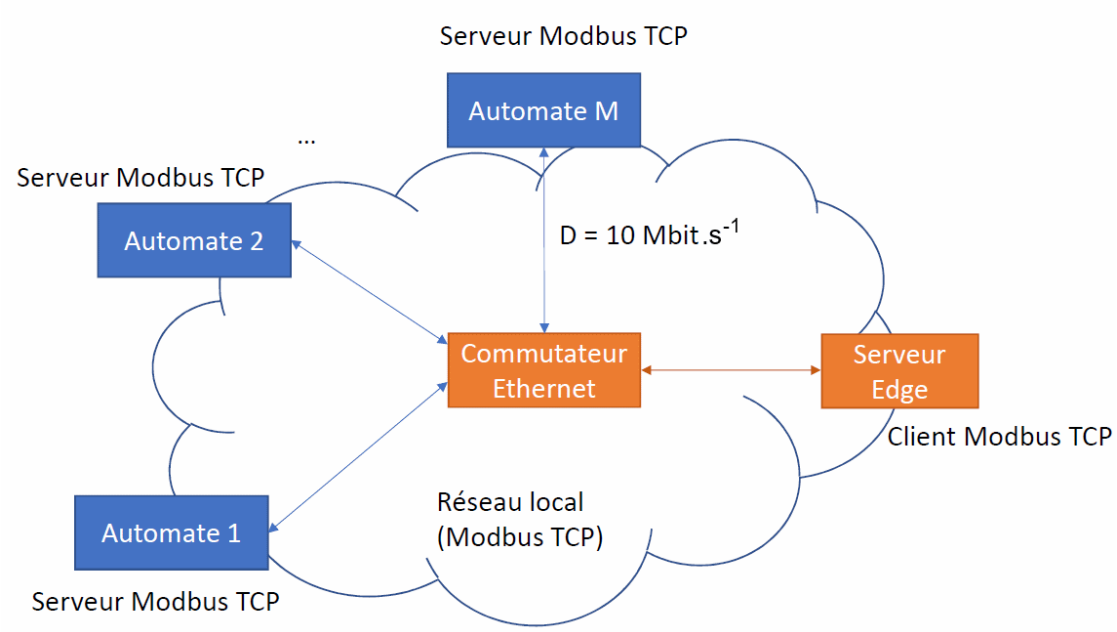


Figure 13 - Réseau Modbus TCP sur Ethernet commuté

Le commutateur de la figure 13 fonctionne en mode Store-and-Forward. Il reçoit complètement une trame venant d'une interface d'entrée avant de la mettre dans la file d'attente puis de l'envoyer sur l'interface de sortie. La figure 14 représente la file d'attente associée à l'interface sortie vers le serveur Edge.

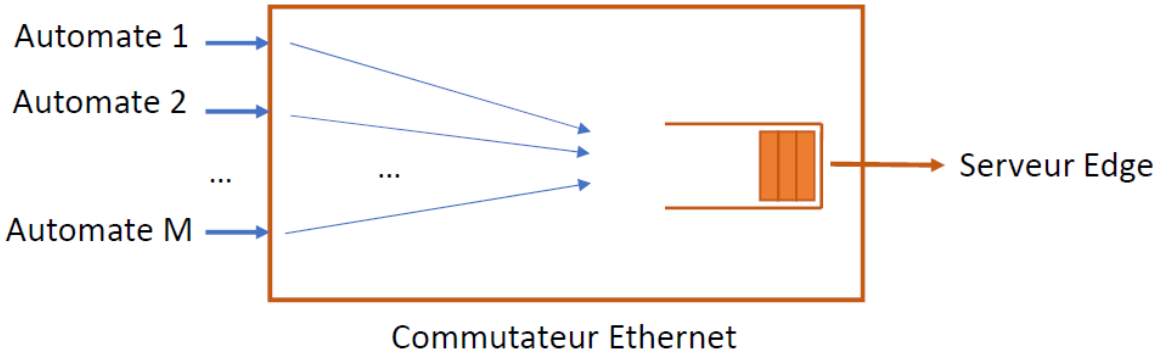


Figure 14 - Modèle de file d'attente dans le commutateur Ethernet

Pour des raisons de simplification, la distance entre automate et commutateur et entre commutateur et informatique de périphérie (serveur Edge) aura une valeur de 100 m. Le contexte (ligne de production dans une usine agroalimentaire) permet de faire cette hypothèse.

Q26. Exprimer et calculer la durée minimale de la connexion TCP qui réalise 1 transaction « holding_register » et 1 transaction « read_coils » entre le serveur Edge et un automate. Détailler le raisonnement.

Tournez la page S.V.P.

Certains registres tels que « niveau_réservoir1 », spécifié dans le document technique DT3, nécessitent une consultation régulière afin de disposer de toutes les informations utiles au diagnostic. Les concepteurs identifient que le cycle de polling des M automates ne doit pas dépasser 3 minutes. Dans ces conditions, le nombre maximal d'automates peut être irréaliste. Le but est de simplement analyser les performances de l'application.

Q27. Calculer le nombre maximal d'automates.

Impact des caméras

Comme illustré à la figure 5 dans le cas du démouleur, il est envisagé d'ajouter des caméras à différents niveaux de la ligne afin de surveiller l'état des pâtisseries au cours de la fabrication.

À ce stade, il sera considéré que le transfert d'une image de taille supérieure à 2500 octets de chaque caméra vers le serveur Edge peut survenir à tout moment. Le protocole exact sera étudié en partie 3.

Le modèle de la figure 15 considère K caméras dans une usine.

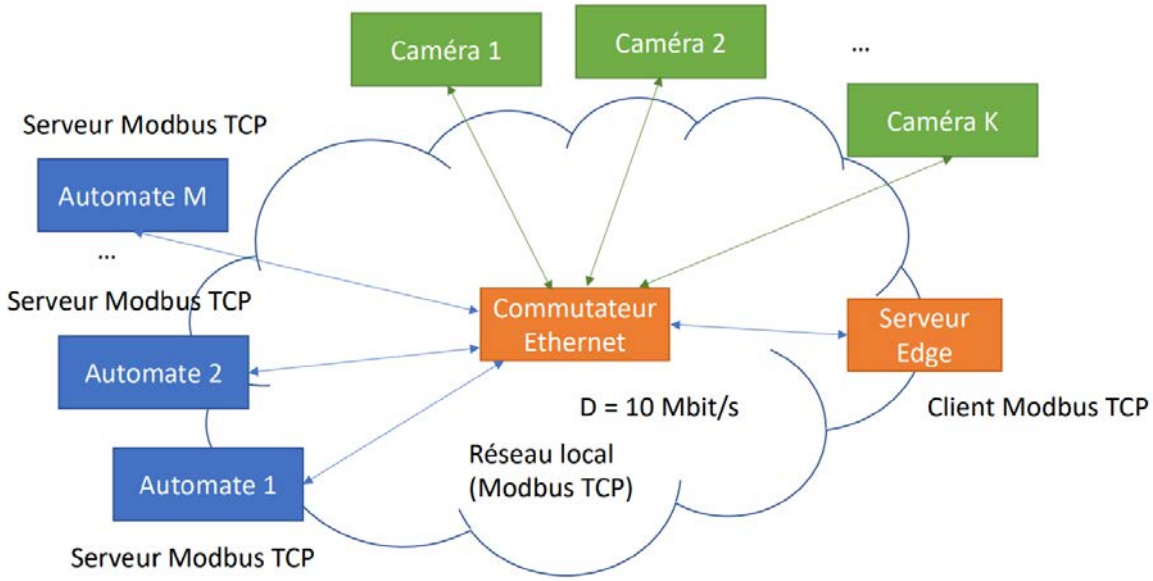


Figure 15 - Réseau de K caméras et M automates

Le rapport entre le nombre d'automates et le nombre de caméras est de 5 automates pour 2 caméras. Le temps maximal du cycle de polling des M automates ne doit pas dépasser 3 minutes. Il sera postulé que lorsqu'une trame Modbus arrive au commutateur, il n'y a pas de trame restante dans la file d'attente mais les trames d'image venant des caméras pourraient arriver au commutateur en même temps que la trame Modbus.

Q28. Exprimer, en fonction de M et K de la figure 15, la durée maximale d'une connexion TCP Modbus qui réalise les deux transactions précédentes. En déduire le nombre d'automates et de caméras supportés par le système.

Réseau Wifi

Pour connecter un grand nombre d'automates et de caméras, la solution câblée devient contestable par le maître d'ouvrage. Il convient donc de vérifier les performances de l'application dans le cas d'un réseau Wifi.

Dans la suite, il sera considéré un point d'accès Wi-Fi 802.11g qui fonctionne à 54 Mbit·s^{-1}, comme illustré dans la figure 16.

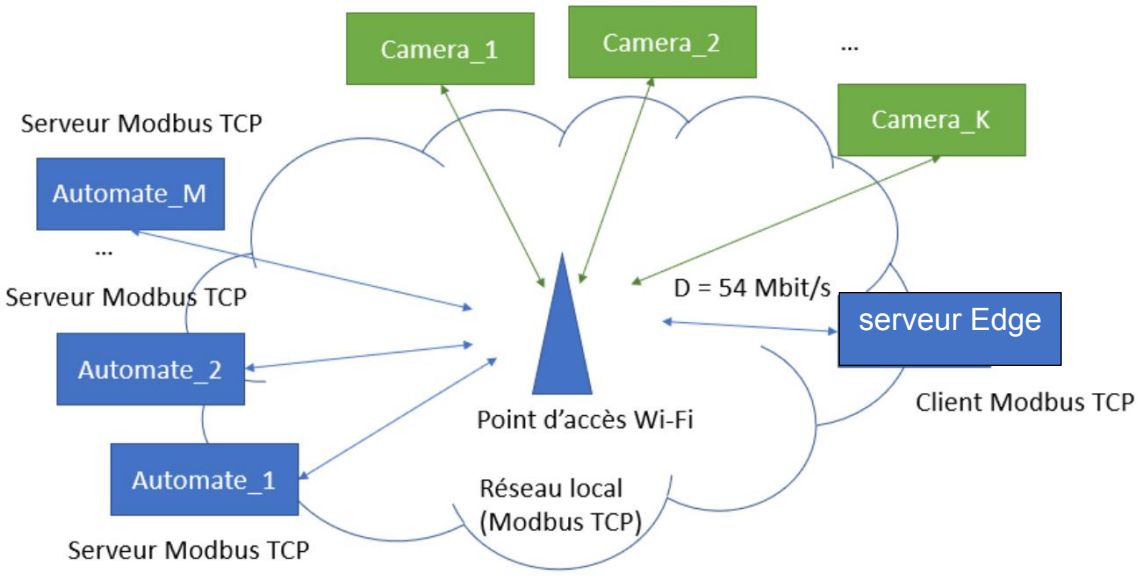


Figure 16 - réseau local Wi-Fi pour interconnecter M automates et K caméras

Un des problèmes du réseau Wi-Fi est la collision. Quand une station (un automate, une caméra ou le serveur Edge) doit émettre une trame de données, elle écoute le canal. Si le canal est libre pour un temps d'attente inter-trame DIFS (Distributed coordination function Interframe Space) de 50 µs, la station émet la trame. À la réception de la trame de données, le point d'accès Wi-Fi attend un temps inter-trame court SIFS (Short Interframe Space) de 10 µs et envoie une trame d'acquittement (ACK). La figure 17 détaille l'envoi d'une trame de données du serveur Edge à un automate en passant par le point d'accès Wi-Fi.

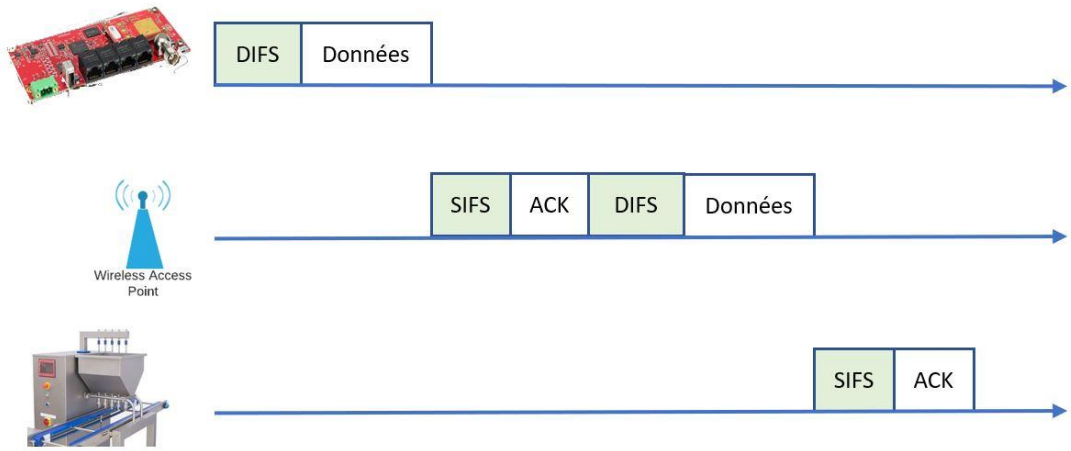


Figure 17 – Envoi d'une trame de données serveur Edge vers automate

Tournez la page S.V.P.

Le format de la trame Wi-Fi est présenté dans la figure 18. Le chiffre spécifié au-dessus de chaque champ correspond au nombre d'octets du champ. Une trame ACK est une trame de contrôle qui n'a pas de données (Frame body = 0 octet)". Une trame de données Wi-Fi peut transporter au maximum 2312 octets de données.

Figure 18 - Format de la trame Wi-Fi

Q29. En négligeant le temps de propagation, calculer le temps minimal pour réaliser les deux transactions Modbus précédentes entre le serveur Edge et un automate. Justifier le résultat.

Dans ces conditions, le nombre maximal d'automates peut paraître irréaliste. L'objectif est, toujours, de vérifier les performances.

Q30. Calculer le nombre maximal d'automates sans présence de caméra.

Il convient, maintenant, de se pencher sur ce qu'il se passe lorsque le canal est occupé ou devenu occupé durant le temps DIFS.

La station émettrice doit ajouter un temps d'attente aléatoire (T_backoff) mesuré en nombre de tranches (R) de temps. La norme 802.11g définit une durée de tranche de temps de 20 µs. La valeur de R est un nombre entier entre 0 et la taille maximale de la fenêtre de contention (CW - Contention Window) moins un (CW-1) comme présenté dans le document technique DT8. Après le temps DIFS, la station doit attendre un temps T_backoff avant d'émettre la trame. En cas de collision, c'est-à-dire si la station émettrice ne reçoit pas d'acquittement, la trame sera retransmise avec la taille de la fenêtre de contention doublée (document technique DT8). La figure 19 illustre l'envoi d'une trame de données du serveur Edge au point d'accès Wi-Fi dans le cas où le canal est occupé lors de la première écoute et avec une retransmission dû à une collision.

Figure 19 – Illustration d'un envoi dans le cas où le canal est occupé

La raison principale qui conduit à avoir des collisions dans le réseau est la charge de celui-ci. C'est particulièrement le cas quand les caméras sont ajoutées dans le système. Dans l'hypothèse où une caméra représente un débit d'environ 408 kbit·s^{-1}, un réseau Wi-Fi à 54 Mbit·s^{-1}ne peut pas supporter plus de 132 caméras. Par conséquent, le nombre de caméras présents dans le réseau a un impact direct sur le nombre de collisions consécutives et donc le nombre de retransmissions requises. Il sera considéré que si le nombre de caméras est entre 1 et 6, il faut une retransmission

pour que la trame soit effectivement transmise. Si le nombre de caméras est entre 7 et 18, il faut 2 retransmissions. Si le nombre de caméras est entre 19 et 36, il faut 3 retransmissions. Si le nombre de caméras est entre 37 et 70, il faut 4 retransmissions. Si le nombre de caméras est entre 71 et 100, il faut 5 retransmissions. Enfin, si le nombre de caméras est entre 101 et 132, il faut 6 retransmissions pour qu'une trame soit transmise avec succès.

Il est rappelé que, dans le modèle choisi, chaque groupe de 5 automates est équipé de 2 caméras.

Q31. En déduire le nombre maximal de groupes des 5 automates et 2 caméras pour que le temps de collecte des données dans un cycle de collecte ne dépasse pas 3 minutes.

Partie 3. Amélioration du modèle choisi pour l'identification des défauts en sortie de four

L'objectif de cette partie est de vérifier les performances de la solution choisie pour localiser les zones de cuisson défectueuses ou conduisant à une mauvaise qualité de la pâtisserie. Cette solution doit être valide quelle que soit la pâtisserie et son format (tartelette, tarte 27 cm, bread, mini-fondants, canelés, etc.). Après une modélisation de la cuisson du gâteau, une amélioration sera envisagée.

Plusieurs gâteaux sont cuits simultanément dans un four. La température n'y est pas homogène.

La figure 20 montre un exemple de plateau pour la confection de mini-gâteaux. Sa forme, sa matière et la taille des gâteaux vont influer sur le refroidissement du gâteau et la qualité de la détection des défauts par caméra thermique.

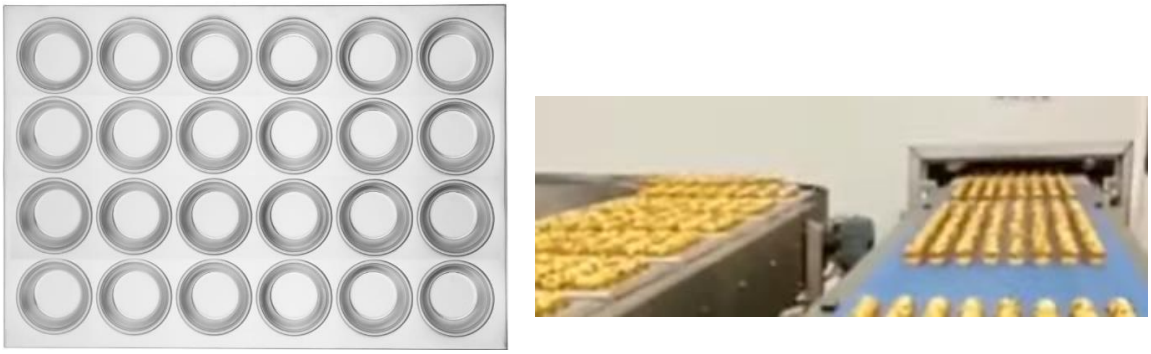


Figure 20 - Exemple de plateau utilisé

À la sortie du four, les gâteaux sont déposés avec leur plateau en acier inoxydable sur des tapis. Ils sont ensuite convoyés jusqu'à une machine nommée « démouleur » sur la figure 5. Ils sont alors démoulés et déposés sur un nouveau tapis afin d'être transportés vers une nouvelle machine (nappage par exemple) ou bien la mise en carton.

Capturer une image thermique des gâteaux cuits permet de diagnostiquer un problème d'homogénéité de la chaleur dans le four mais aussi d'identifier des pâtisseries imparfaites (forme non commercialisable). Dans le contexte de l'application TRS, il ne s'agit pas de remplacer l'opérateur mais de confirmer ses observations.

La photo peut être prise en sortie du four (180°C) alors que les gâteaux sont encore dans leur plateau ou bien 5 mn plus tard, quand ils sont démoulés. Cette dernière solution a été choisie par les concepteurs de l'application comme illustré à la figure 5.

Analyse de l'existant

Toutes les 3 s (rythme de démoulage des gâteaux), l'informatique en périphérie (serveur Edge) transmet l'ordre « store -j temp.jpg » à la caméra thermique qui capture une image et l'enregistre sous le nom « temp.jpg ». Un nouvel ordre (« RETR temp.jpg ») est transmis afin de récupérer l'image et de la stocker avec le nom de fichier « file-YYYYMMDD-hhmmss.jpg ». Enfin, l'ordre de suppression du fichier temporaire est envoyé (« del temp.jpg »). Le serveur Edge traite alors l'image afin d'en extraire une caractérisation sous la forme d'une position (x,y) et d'un rayon pour chaque gâteau du plateau. Ces données sont, enfin, transmises à l'application cloud. Ceci est illustré par la figure 21.

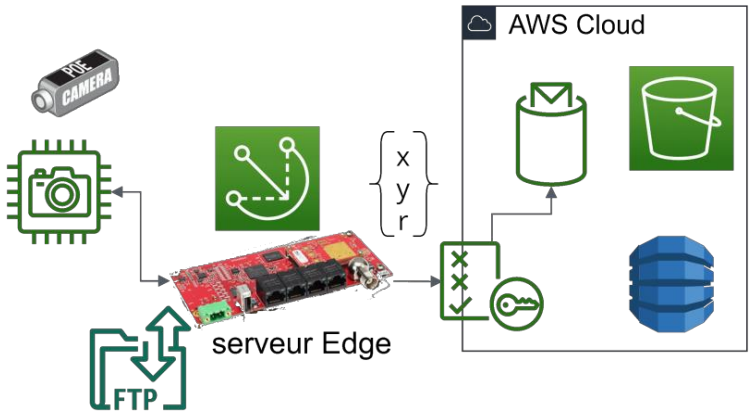


Figure 21 - Principe de la valorisation de la donnée

Un extrait du programme effectuant cette transformation est donné à la figure 22 de la page suivante.

La classe FLIR est déclarée comme indiqué dans l'extrait ci-dessous :

```
class FLIR(object):
    def __init__(self, host):
        self.tn = None
        self.host = host
        self.port = 23
        self.timeout = 2
```

La méthode shootJPG(path) permet de prendre une photo distante depuis le serveur Edge et sera étudiée ultérieurement.

Une description sommaire de quelques méthodes de la bibliothèque « openCV » est fournie dans le document technique DT5.

Q32. Donner le diagramme d'activités décrivant le comportement de l'extrait de programme de la figure 22.

```
cam = flir('192.168.1.10')

while True:
    mqtt_client.initPayload(cam)
    cam.connect()
    cam.slowFocus()
    filename = cam.shootJPG('')

    src = cv.imread(filename)
    src_gray = cv.cvtColor(src, cv.COLOR_BGR2GRAY)
    img_blur = cv.GaussianBlur(src_gray, (3,3),3,10,cv.BORDER_CONSTANT )
    lesCircles = cv.HoughCircles(img_blur, cv.HOUGH_GRADIENT,1,90,param1=56,
                                 param2=5, minRadius=20,maxRadius=60)

    ch=1
    if lesCircles is not None:
        lesCircles = np.round(lesCircles[0, :]).astype("int")
        for (x, y, r) in lesCircles:
            payload = mqtt_client.addPayload('x:{0},y:{1},r:{2}'.format(x,y,r))
            ch=ch+1

        mqtt_client.publish(
            topic='dt/device_data/{0}'.format(device['name']),
            payload=json.dumps(payload))

    time.sleep(2)
```

Figure 22 - Extrait du programme de traitement de l'image

Comme vu précédemment, la méthode shootJPG(path) permet de prendre une photo depuis le serveur Edge. Elle envoie l'ordre « store -j temp.jpg » puis la requête « RETR temp.jpg » avant d'enregistrer la photo et de supprimer le fichier « temp.jpg » dans la caméra avec la commande « del ». La commande « store » (*store*) est une commande spécifique de la caméra utilisable après une connexion telnet. Elle demande à la caméra d'accepter les données et de les enregistrer dans le fichier portant le nom passé en paramètre. Si le fichier n'existe pas, elle le crée, sinon elle l'écrase. La commande « retr » (*RETRIEVE*) demande à la caméra une copie du fichier dont le chemin d'accès est passé en paramètre.

Quelques méthodes de la bibliothèque « ftplib » sont fournies dans le document technique DT6. La partie connexion « telnet » ne sera pas traitée.

Q33. Donner le code python qui va permettre l'envoi et le traitement de la requête « RETR temp.jpg ».

Pour la question suivante, Il conviendra de rester au niveau de la couche Application des réseaux. La partie connexion telnet ne sera toujours pas traitée.

Q34. Donner le diagramme de séquence SysML des échanges entre le serveur Edge et la caméra lors de l'appel à shootJPG.

Tournez la page S.V.P.

Dès le premier jour de test in-situ, des problèmes sont progressivement apparus au cours de la journée. Lors de l'analyse des données affichées sur l'interface :

- plus de gâteaux sont détectés avec un défaut par l'application que par l'opérateur humain,

- des gâteaux sont positionnés à des endroits incohérents par rapport à l'observation.

La figure 23 illustre ce phénomène par deux représentations à deux moments de la journée. Cette figure fournit la photo après transformation en niveaux de gris 8 bits à laquelle est superposée une représentation des coordonnées de chaque cercle.

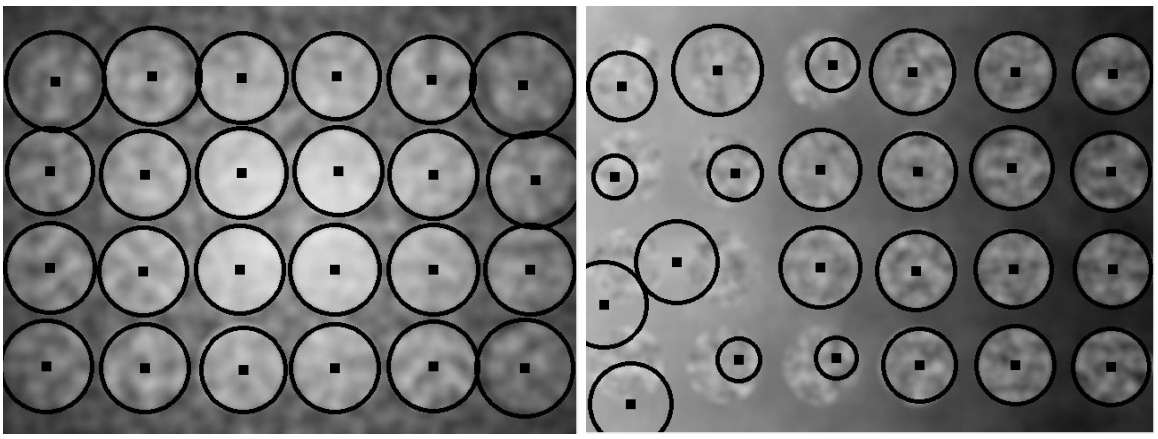


Figure 23 - Exemple de résultat après traitement d'une image

Q35. À partir du diagramme d'activités, de l'extrait de code et du diagramme de séquence précédents, identifier trois causes possibles des dysfonctionnements listés précédemment.

D'après la description de la sous-partie « Analyse de l'existant », toutes les 3 s, le serveur Edge demande une photo à la caméra, la récupère puis la traite. Il y a potentiellement 5 lignes dans l'usine et 1 four par ligne de production.

D'après la situation décrite à la partie 1, la ligne est planifiée pour fonctionner de 5h00 à 17h30.

Un extrait de la documentation du serveur Edge et de la caméra est donné en document technique DT7.

Q36. En mettant de côté toute autre considération, calculer l'occupation maximale (en Mio) en mémoire de stockage de ces photos prises toutes les 3 s.

Q37. Comparer cette valeur à la documentation du serveur Edge et argumenter, sur ce premier choix, en matière de qualité du modèle choisi, de pertinence et de compatibilité avec le matériel.

Plusieurs plateaux sont généralement cuits simultanément dans un four. Les gâteaux de la figure 20 sont généralement cuits en 20 mn à 180°C. Ce n'est pas le cas de tous. La durée dépend du type de gâteau et de sa taille. La cadence de démoulage du lot de plateaux de gâteaux cuits se fait ensuite à la cadence constante d'un plateau toutes les 3 s.

La figure 24 propose une évolution de l'application sous la forme d'un diagramme d'activités SysML partiel et simplifié. La partition de gauche représente les activités de la caméra. La partition de droite représente celle du serveur Edge.

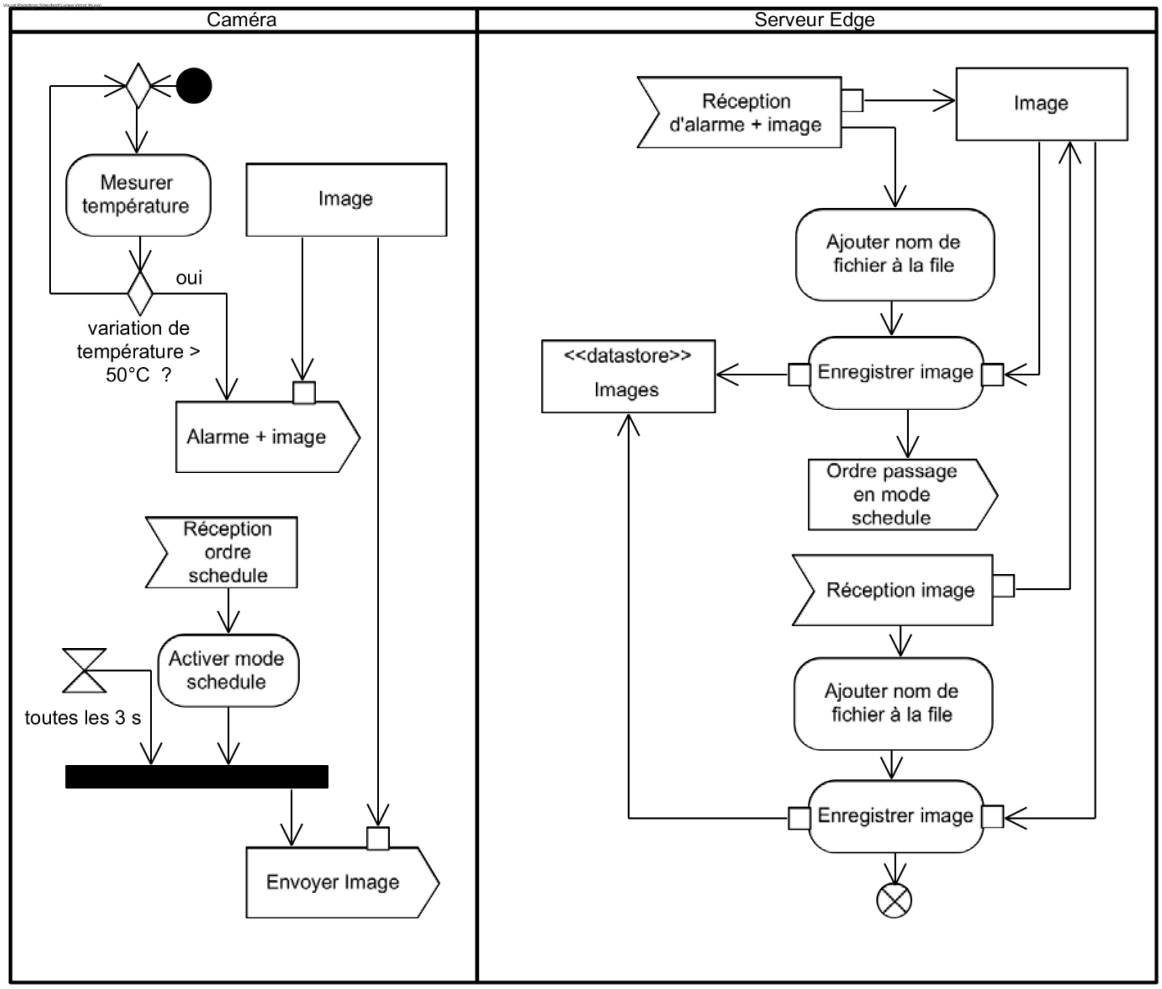


Figure 24 - Diagramme d'activité SysML partiel et simplifié de la solution envisagée

Q38. Interpréter le diagramme de la figure 24 et décrire comment la mise en œuvre de ce modèle peut résoudre les problèmes constatés.

Un extrait de la documentation de la caméra est donné en document technique DT7.

Q39. Vérifier que la caméra permet bien la mise en œuvre de ce modèle. Justifier.

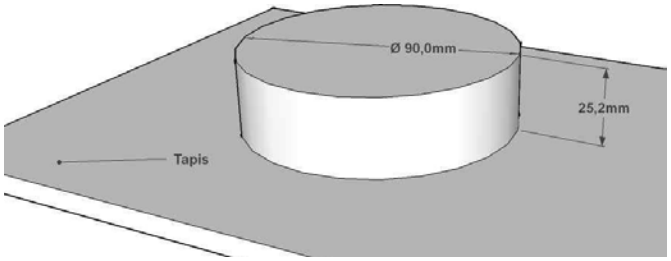
En dépit de cette évolution, des problèmes d'identification des défauts ont persisté. Après réflexion, il est apparu que les concepteurs de l'application n'ont pas tenu compte de contraintes « métier ». Une vérification des phénomènes physiques est nécessaire.

Tournez la page S.V.P.

Modélisation de la répartition de la chaleur

L'objectif de cette sous-partie est de vérifier l'évolution de la température du gâteau et celle du tapis au cours de l'exécution de l'ordre de fabrication afin d'évaluer l'adéquation des performances de l'application avec le réel.

Le modèle choisi pour le gâteau est un cylindre de rayon 90 mm et de hauteur 25,2 mm. Sa masse sera de 95 g.



Les hypothèses destinées à simplifier la suite du sujet seront :

- le système {gâteau} est incompressible,
- il n'y a pas de transfert par travail lors du refroidissement,
- il n'y a pas d'énergie mécanique lors du refroidissement.

Alors, le premier principe de la thermodynamique donne :

(8) $\Delta Et = \Delta U = Q$

Avec : Et – énergie totale

ΔU – variation d'énergie interne du système incompressible {gâteau}

Q – transfert de chaleur avec l'extérieur

L'équation (8) donne : $m \times C_m \times (\theta_f - \theta_i) = Q$ (9)

Où : m est la masse du gâteau, C_m est sa capacité massique, θ_f est la température finale et θ_i est sa température initiale

Q40. Sachant que la loi phénoménologique de Newton stipule que le taux de perte de chaleur d'un corps est proportionnel à la différence de température entre le corps et le milieu environnant, montrer que l'équation (9) peut s'écrire :

$$m \times C_m \times d\theta = h \times S \times (\theta_f - \theta(t)) \times dt$$

avec : h = coefficient d'échange convectif et S surface de contact avec l'air.

Q41. En déduire l'équation différentielle qui permettra d'obtenir l'évolution de la température du gâteau en fonction du temps.

La température dans les ateliers est généralement comprise entre 7 °C et 9 °C. Toutefois, autour des fours, la température est approximativement de 20 °C.

Q42. En déduire l'équation de la température du gâteau en fonction du temps.

Après démoulage, le gâteau est posé sur un tapis à température ambiante. La capacité massique du gâteau étudié est :

C_m = 1408 J·kg^{-1}·K^{-1}

Le coefficient d'échange convectif de l'air légèrement ventilé est :

h = 20,0 W·m^{-2}·K^{-1}

Q43. Calculer la température atteinte au bout de 30 s pour les 3 situations suivantes :

- seule la surface supérieure du gâteau est soumise à l'air ambiant,
- la moitié du gâteau (toujours posé sur la tapis) est soumis à l'air ambiant,
- le gâteau entier (toujours posé sur la tapis) est soumis à l'air ambiant.

Cette analyse montre, dans le pire des cas, une différence de 5 °C entre les gâteaux les plus refroidis et ceux les moins refroidis. La question se pose sur la possibilité ou non de différencier le gâteau et le tapis sur une image thermique.

Les caractéristiques du tapis sont :

- surface élémentaire de tapis considérée : L = l = 110 mm
- épaisseur du tapis : e = 0,6 mm
- masse volumique du tapis : m_{v_t}= 1200 kg·m^{-3}
- capacité thermique massique du tapis : C_{m_t} = 1674J·kg^{-1}·m^{-1}

Q44. Exploiter l'équation obtenue précédemment afin d'en déduire la température minimale et la température maximale de l'élément de tapis.

Q45. Conclure sur l'évolution possible de la température du gâteau et de celle du tapis au cours de l'exécution de l'ordre de fabrication. Préciser l'impact sur le programme de la figure 22.

Q46. Indiquer les données à transmettre au serveur Edge afin d'utiliser ce modèle dans le programme.

Sur les lignes de production, le format des pâtisseries (27 cm ou mini, tarte aux fruits ou gâteaux marbrés, etc.) varie en fonction de la demande des clients et de l'imagination des chercheurs du laboratoire de recherche de Dabas. La modélisation précédente montre qu'il est difficile de construire un modèle unique permettant la détection précise des défauts. Par ailleurs, plusieurs hypothèses ont été prises pour obtenir les résultats précédents. Une période d'apprentissage avec le réel pourra être nécessaire. Les concepteurs envisagent donc d'utiliser l'intelligence artificielle.

Un extrait de la documentation du serveur Edge et de la caméra est donné en document technique DT7.

Q47. Argumenter sur la possibilité de faire du Deep Learning avec le serveur Edge disponible.

Tournez la page S.V.P.

Partie 4 – Impact d'un transfert du Deep Learning vers le Cloud

L'objectif de cette partie est de modéliser différentes situations de transfert entre les usines et le Cloud afin de vérifier la possibilité de traiter les images dans le Cloud.

Il est envisagé de transmettre les images au Cloud. En effet, ces fournisseurs disposent d'outils de Deep Learning performants.

Les données sont envoyées des 4 usines installées à Le-boin, Burzy-sur-guyre, Haide-sur-sye et Dabas. Le centre de données AWS se trouve en Ile-de-France.

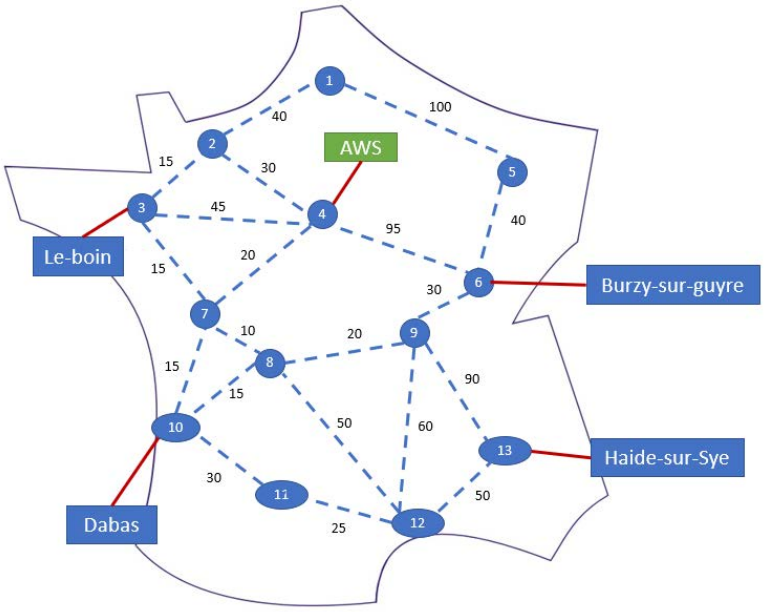


Figure 25 - Topologie du réseau de transport TRS

Afin d'évaluer les besoins, il conviendra de considérer les hypothèses suivantes :

- 1 ligne de production comporte 5 automates,

- 1 ligne de production contient 1 four,

- 1 usine compte 5 lignes de production,

- 1 image d'au maximum 153600 octets est transmise toutes les 3 s,

- 1 message MQTT « Publish » fait 257 octets au maximum, y compris l'entête et le checksum Ethernet,

- 1 message MQTT est construit pour chacune des 4 informations,

- 1 message MQTT est construit pour chaque « coil »,

- le réseau local est basé sur Ethernet commuté décrit précédemment.

Q48. Exprimer et calculer le débit à réserver sur le chemin entre chaque usine et le centre de données AWS dans le cas où les images sont traitées en local (*Serveur Edge*). Justifier le résultat.

Q49. Calculer le débit à réserver dans le cas où les images sont traitées par les services AWS. Justifier le résultat et conclure.

Le document technique DT9 décrit l'algorithme de Dijkstra.

Q50. À l'aide de l'algorithme de Dijkstra, trouver, document réponse DR4, et proposer le chemin le plus court du centre de données AWS vers chacun des sites.

Les coûts des chemins correspondent au délai mesuré en millisecondes. Les routeurs peuvent rejeter les paquets en cas de congestion.

Q51. Si la probabilité de perte de paquets dans chaque routeur est égale à p, exprimer la probabilité qu'un paquet partant du site de Dabas arrive au serveur AWS avec succès.

Files d'attente

La figure 26 illustre les files d'attente dans le routeur 4.

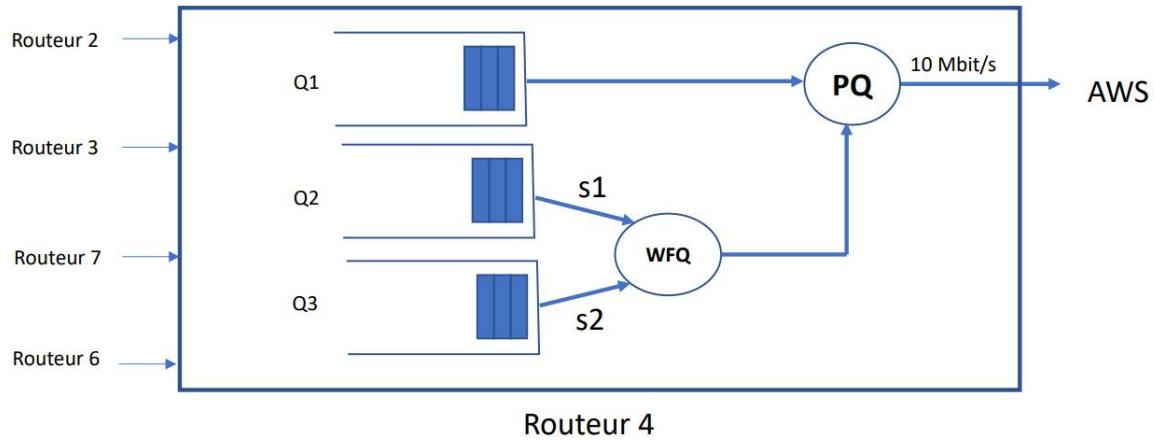


Figure 26 - Files d'attente dans le routeur 4

La file d'attente Q1 reçoit les données multimédia comme les images (qui arrivent à Q1 au débit D1).

Les données IoT (à un débit D3) sont envoyées à la file d'attente Q3.

La file d'attente Q2 est utilisée pour servir d'autres types de trafic.

Les files d'attente Q2 et Q3 sont servies avec l'algorithme Weighted Fair Queuing (WFQ) avec les taux de service s1 et s2 respectivement (s1 + s2 = 1).

La file d'attente Q1 est servie par l'algorithme Priority Queuing dans lequel les paquets de la file Q1 ont une priorité absolue.

Le taux de perte de paquet dans chacune des files d'attente est égal à p = 0,1.

Q52. Exprimer en fonction de D1 [Mbit·s^{-1}], D3 [Mbit·s^{-1}] et p puis calculer la valeur minimale du taux de service s2 pour assurer que le taux de perte d'un paquet MQTT est égal ou inférieur à p.

Tournez la page S.V.P.

Dans TCP, pour chaque segment de données envoyé, l'émetteur démarre un temporisateur pour attendre l'acquittement. Après un temps Retransmisison TimeOut (RTO), si l'acquittement n'est pas reçu, l'émetteur retransmet le segment de données. Le temps RTO est souvent calculé comme 2 fois le temps d'aller-retour (Round Trip Time – RTT).

Le délai moyen pour publier des données est calculé entre le moment ou le serveur Edge envoie un segment TCP qui encapsule un message MQTT et le moment ou il reçoit l'acquittement. Il est supposé qu'un segment TCP est perdu au maximum une seule fois.

Q53. Exprimer et calculer le délai moyen pour publier des données transportées par un message MQTT issu du serveur Edge de l'usine de Dabas.

Document technique DT1 : organisation ou lexique

S3 (Simple Storage Service) est un service en ligne de stockage de fichiers statiques. Amazon S3 propose de stocker des données dans des *buckets (*littéralement, des... seaux). Ce sont des sortes de gros conteneurs qui peuvent stocker autant de fichiers que l'on veut éventuellement répartis dans des dossiers.

Amazon S3 (Amazon Simple Storage Service) fournit des outils de journalisation et de surveillance qui peuvent être utilisés pour surveiller et contrôler la façon dont les ressources sont utilisées.

DynamoDB est un service de base de données NoSQL entièrement géré.

Athena est un service de requêtes interactif qui facilite l'analyse de données directe dans Amazon Simple Storage Service (Amazon S3) via la syntaxe SQL standard. Athena SQL fonctionne sans serveur, ce qui signifie qu'il n'y a aucune infrastructure à mettre en place ou à gérer et que seules les requêtes exécutées sont facturées.

Glue est un service d'intégration de données évolutif et sans serveur qui facilite la découverte, la préparation et la combinaison des données pour l'analyse, le machine learning et le développement d'applications.

QuickSight permet aux membres des organisations de comprendre leurs données en posant des questions en langage naturel, en les explorant via des tableaux de bord interactif, ou en recherchant automatiquement des modèles et des anomalies alimentés par le machine learning.

API Gateway est un service qui permet aux développeurs de créer, publier, gérer, surveiller et sécuriser facilement des API à n'importe quelle échelle. Les API servent de « porte d'entrée » pour que les applications puissent accéder aux données, à la logique métier ou aux fonctionnalités de services backend. À l'aide d'API Gateway, il est possible de créer des API RESTful et des API WebSocket qui permettent de concevoir des applications de communication bidirectionnelle en temps réel.

IoT SiteWise est un service géré qui facilite la collecte, l'organisation et l'analyse de données d'équipements industriels à grande échelle.

MQTT (Message Queuing Telemetry Transport) est un protocole de messagerie léger adapté aux situations où les clients doivent utiliser peu de code et sont connectés à des réseaux peu fiables ou limités en bande passante. Il est principalement utilisé dans la communication entre machines (M2M) ou sur les types de connexions propres à l'Internet des Objets.

IoT Greengrass est un environnement d'exécution en périphérie qui étend les fonctionnalités du cloud aux appareils locaux. Il permet aux appareils de collecter et d'analyser les données plus près de la source des informations, de réagir de manière autonome aux événements locaux et de communiquer en toute sécurité sur les réseaux locaux.

Lambda *(λ)*  *est* un service informatique qui permet d'exécuter un code sans demander la mise en service ou la gestion des serveurs.

Tournez la page S.V.P.

IoT rules transfèrent les données issues des d'équipements industriels à d'autres services AWS. Elles écoutent les messages MQTT spécifiques, formatent les données dans les charges utiles des messages et envoient le résultat à d'autres services.

AWS Identity and Access Management (**IAM**)  permet de contrôler l'accès aux services et aux ressources, de gérer de manière centralisée les autorisations précises et d'analyser l'accès pour affiner les autorisations.

Document technique DT2 : extrait de la requête Athena de construction des données d'affichage d'un ordre de fabrication

```
select  /* infos Ordres de Fabrication */
    "OF".id,
    "OF".fabricationOrder,
    "OF".factory,
    "OF".responsible,
    "OF".line,
    "OF".fabricationNumber,
    "OF".fabricationName,
    "OF".clientId,
    "OF".clientLabel,
    "OF".startDateYear,
    "OF".startDateHour,
    "OF".startDate_timestamp,
    "OF".endDateYear,
    "OF".endDateHour,
    "OF".endDate_timestamp,
    "OF".state,
    "OF".processed,
    "OF".img,
    "OF".numberOfItems,
    "OF".productFamily,
    "OF".format,
    "OF".breadFamily,
    "OF".deliveryType,
    "OF".packagingType,
    "OF".budgetPeopleNb,
    "OF".budgetRate,
    "OF".nominalRate,
    "OF".outUV,
    "OF".finalUV,
    "OF".weightUV,
    "OF".unitPRV,
    OF_events_first.start_timestamp as Actual_startDate_timestamp,
    OF_events_last.start_timestamp as Actual_EndDate_timestamp,
    cast( (OF_events_last.start_unixtime - OF_events_first.start_unixtime) as double) /
    1000 / 3600         as temps_ouverture,
    temps_ouverture - cast( (OF_events_eat.start_unixtime - OF_events_eat.start_unixtime) as double) /
    1000 / 3600 as temps_requis,
    (OF_events_last.conform / "OF".nominalRate ) /  temps_requis    as TRS,
    thisOF_history.delta as nb_plateau_deb_periode
from
    "OF"
        left join
            (
                select *
                from OF_events
                where num_event_tab=1
            ) as OF_events_first
                on "OF".fabricationOrder=OF_events_first.fabricationOrder
        left join
            (
                select *
                from OF_events
                where   reason_name='Arrêt' and completed=true
            ) as OF_events_last
                on "OF".fabricationOrder=OF_events_last.fabricationOrder
        left join
            (
                select *
                from OF_events
                where   reason_name='Repas'
            ) as OF_events_eat
                on "OF".fabricationOrder=OF_events_last.fabricationOrder
```


Tournez la page S.V.P.

List of reserved keywords in SQL SELECT statements

Athena uses the following list of reserved keywords in SQL `SELECT` statements and in queries on views.

If you use these keywords as identifiers, you must enclose them in double quotes (") in your query statements.

```
ALTER, AND, AS, BETWEEN, BY, CASE, CAST,
CONSTRAINT, CREATE, CROSS, CUBE, CURRENT_DATE, CURRENT_PATH,
CURRENT_TIME, CURRENT_TIMESTAMP, CURRENT_USER, DEALLOCATE,
DELETE, DESCRIBE, DISTINCT, DROP, ELSE, END, ESCAPE, EXCEPT,
EXECUTE, EXISTS, EXTRACT, FALSE, FIRST, FOR, FROM, FULL, GROUP,
GROUPING, HAVING, IN, INNER, INSERT, INTERSECT, INTO,
IS, JOIN, LAST, LEFT, LIKE, LOCALTIME, LOCALTIMESTAMP, NATURAL,
NORMALIZE, NOT, NULL, OF, ON, OR, ORDER, OUTER, PREPARE,
RECURSIVE, RIGHT, ROLLUP, SELECT, SKIP, TABLE, THEN, TRUE,
UNESCAPE, UNION, UNNEST, USING, VALUES, WHEN, WHERE, WITH
```

- **Min(), max()** – The min() and max() aggregation functions now allow unknown input types at query analysis time so that you can use the functions with NULL literals.

Document technique DT3 : extraits du code de la passerelle Modbus/MQTT et documentation python time sleep (2 pages)

#Extrait du fichier « ModbusToIOT.py »

```python
import greengrasssdk  # noqa

from pymodbus.client.sync import ModbusTcpClient as ModbusClient  # noqa
from pymodbus.payload import BinaryPayloadDecoder  # noqa
from pymodbus.constants import Endian  # noqa

with open('bonnepat.json') as json_file:
  config = json.load(json_file)

HOSTS = os.environ.get('HOSTS', '{0}:{1}'.format(config['mqtt_host'],config['modbus_port']))
POLL_INTERVAL = os.environ.get('POLL_INTERVAL', config['polling_freq'])
UNIT = os.environ.get('UNIT', 1)
mqtt_client = greengrasssdk.client('iot-data')
mb_clients = get_modbus_clients(HOSTS)

while True:
    for  mbc in mb_clients:
        for device in config['servers']:
            poll_device(mbc, device, mqtt_client)

    time.sleep(POLL_INTERVAL)
```

#Extrait du fichier « bonnepat.json »

```json
{
    "mqtt_host"             :   "127.0.0.1",
    "polling_freq"          :   5,
    "modbus_port"           :   5020,
    "servers"    :
    [
        {
            "name":"Doseuse_A",
            "hr_addr_from" : 150,
            "hr_addr_to" : 157,
            "holding_registers":
            [
                { "name":"compteur_plateaux", "address":150 },
                { "name":"niveau_réservoir1", "address":152 },
                { "name":"niveau_réservoir2", "address":154 },
                { "name":"courant", "address":156 }
            ],
            "coils":
            [
                { "name":"marche_production", "address":150 },
                { "name":"marche_reglage", "address":151 },
                { "name":"marche_vidange", "address":152 },
                { "name":"marche_lavage", "address":153},
                { "name":"arret_ligne", "address":154 },
                { "name":"defautAUZ1", "address":155 },
                { "name":"defautAUZ2", "address":156 },
                { "name":"defaut_bras_1", "address":157 },
                { "name":"defaut_bras_2_3", "address":158 },
                { "name":"defaut_boucle_entremet", "address":159 },
                { "name":"defaut_movifitZ1", "address":160 },
                { "name":"defaut_movifitZ2", "address":161 }
            ]
        },
```


Tournez la page S.V.P.

#Extrait modbus map dosing unit S105

dosing unit S105

Register Address

Via Unit 1	R/W	Data Type	Content
150 - 151	RW	DWord	plaques counter
152 - 153	R	real	reservoir level 1
154 - 155	R	real	reservoir level 2
156 - 157	R	real	current

#Extrait Python time sleep()

Python time *sleep* function is used to add delay in the execution of a program. We can use python sleep function to halt the execution of the program for given time in seconds. Notice that python time sleep function actually stops the execution of current thread only, not the whole program.

Python time *sleep*() function syntax

Python *sleep*() is a method of python time module. So, first we have to import the time module then we can use this method. Way of using python sleep() function is:

Example :

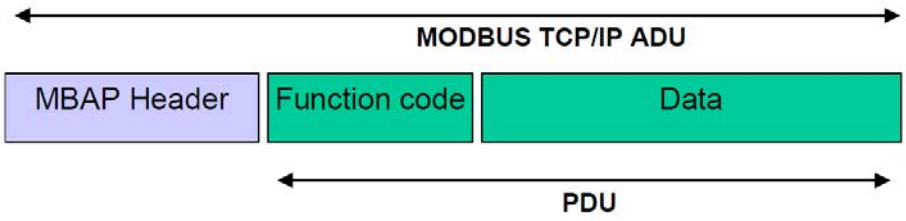
```
import time

t= 2 # 2 seconds

time.sleep(t)
```

Document technique DT4 : extrait de documentation Modbus TCP (2 pages)

Format de la trame Ethernet transportant un message Modbus TCP

Ethernet header (14B)	IP header (20B)	TCP header (20B)	**Modbus ADU** (max. 260B)	Checksum (4B)

Format d'une requête ou d'une réponse Modbus (Modbus ADU – Application Data Unit)



MBAP MODBUS Application Protocol

The MBAP Header contains the following fields:

Fields	Length	Description -	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)
Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

The header is 7 bytes long:

•**Transaction Identifier** - It is used for transaction pairing, the MODBUS server copies in the response the transaction identifier of the request.

•**Protocol Identifier** – It is used for intra-system multiplexing. The MODBUS protocolis identified by the value 0.

•**Length** - The length field is a byte count of the following fields, including the UnitIdentifier and data fields.

•**Unit Identifier** – This field is used for intra-system routing purpose. It is typicallyused to communicate to a MODBUS+ or a MODBUS serial line slave through agateway between an Ethernet TCP-IP network and a MODBUS serial line. This field

Tournez la page S.V.P.

isset by the MODBUS Client in the request and must be returned with the same value inthe response by the server.

Function codes description – Read Coils

6.1 01 (0x01) Read Coils

This function code is used to read from 1 to 2000 contiguous status of coils in a remote device. The Request PDU specifies the starting address, i.e. the address of the first coil specified, and the number of coils. In the PDU Coils are addressed starting at zero. Therefore coils numbered 1-16 are addressed as 0-15.

The coils in the response message are packed as one coil per bit of the data field. Status is indicated as 1= ON and 0= OFF. The LSB of the first data byte contains the output addressed in the query. The other coils follow toward the high order end of this byte, and from low order to high order in subsequent bytes.

If the returned output quantity is not a multiple of eight, the remaining bits in the final data byte will be padded with zeros (toward the high order end of the byte). The Byte Count field specifies the quantity of complete bytes of data.

Request

Function code	1 Byte	**0x01**
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of coils	2 Bytes	1 to 2000 (0x7D0)

Response

Function code	1 Byte	**0x01**
Byte count	1 Byte	**N***
Coil Status	**n** Byte	n = N or N+1

***N** = Quantity of Outputs / 8, if the remainder is different of 0 ⇒ N = N+1

6.3 03 (0x03) Read Holding Registers

This function code is used to read the contents of a contiguous block of holding registers in a remote device. The Request PDU specifies the starting register address and the number of registers. In the PDU Registers are addressed starting at zero. Therefore registers numbered 1-16 are addressed as 0-15.

The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

Request

Function code	1 Byte	**0x03**
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Response

Function code	1 Byte	**0x03**
Byte count	1 Byte	2 x **N***
Register value	**N*** x 2 Bytes	

***N** = Quantity of Registers

cv.GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[, borderType]]]) -> dst

Blurs an image using a Gaussian filter.

The function convolves the source image with the specified Gaussian kernel. In-place filtering is supported.

cv.HoughCircles(image, method, dp, minDist[, circles[, param1[, param2[, minRadius[, maxRadius]]]]] -> circles

Finds circles in a grayscale image using the Hough and Canny transform.

The function finds circles in a grayscale image using a modification of the Hough transform.

Parameters

image	8-bit, single-channel, grayscale input image.
circles	Output vector of found circles. Each vector is encoded as 3 or 4 element floating-point vector (*x*,*y*,*radius*) or (*x*,*y*,*radius*,*votes*) .
method	Detection method, see HoughModes. The available methods are HOUGH_GRADIENT and HOUGH_GRADIENT_ALT.
dp	Inverse ratio of the accumulator resolution to the image resolution. For example, if dp=1 , the accumulator has the same resolution as the input image. If dp=2 , the accumulator has half as big width and height.
minDist	Minimum distance between the centers of the detected circles. If the parameter is too small, multiple neighbor circles may be falsely detected in addition to a true one. If it is too large, some circles may be missed.
param1	First method-specific parameter. In case of HOUGH_GRADIENT and HOUGH_GRADIENT_ALT, it is the higher threshold of the two passed to the Canny edge detector (the lower one is twice smaller).
param2	Second method-specific parameter. In case of HOUGH_GRADIENT, it is the accumulator threshold for the circle centers at the detection stage. The smaller it is, the more false circles may be detected. Circles, corresponding to the larger accumulator values, will be returned first.
minRadius	Minimum circle radius.
maxRadius	Maximum circle radius. If <= 0, uses the maximum image dimension. If < 0, HOUGH_GRADIENT returns centers without finding the radius.

Tournez la page S.V.P.

Document technique DT6 : extrait de la documentation ftplib et de la fonction native open de python avec annotations

class ftplib.FTP(host='', user='', passwd='')

➔ When host is given, the method call connect(host) is made.

FTP.connect(*host=''*, *port=0*, *timeout=None*, *source_address=None*)

Connect to the given host and port. The default port number is `21`, as specified by the FTP protocol specification. It is rarely needed to specify a different port number. This function should be called only once for each instance; it should not be called at all if a host was given when the instance was created. All other methods can only be used after a connection has been made. The optional *timeout* parameter specifies a timeout in seconds for the connection attempt. If no *timeout* is passed, the global default timeout setting will be used. *source_address* is a 2-tuple `(host, port)` for the socket to bind to as its source address before connecting.

FTP.login(user='anonymous', passwd='', acct='')

Log in as the given *user*. The *passwd* and *acct* parameters are optional and default to the empty string. If no *user* is specified, it defaults to `'anonymous'`. If *user* is `'anonymous'`, the default *passwd* is `'anonymous@'`. This function should be called only once for each instance, after a connection has been established; it should not be called at all if a host and user were given when the instance was created. Most FTP commands are only allowed after the client has logged in.

FTP.cwd(*pathname : str*)

Set the current directory on the server

FTP.retrbinary(cmd : str, callback, blocksize=8192, rest=None)

Retrieve a file in binary transfer mode. *cmd* should be an appropriate `RETR` command: `'RETR filename'`. The *callback* function is called for each block of data received, with a single bytes argument giving the data block. The optional *blocksize* argument specifies the maximum chunk size to read on the low-level socket object created to do the actual transfer. If optional *rest* is given, a `REST` command is sent to the server, passing *rest* as an argument.

FTP.quit()

---

open(file : str, mode='r', buffering=- 1, encoding=None, errors=None, newline=None, closefd=True, opener=None)

Ouvre file et donne un objet fichier correspondant. *file* est un objet simili-chemin donnant le chemin (absolu ou relatif au répertoire courant) du fichier à ouvrir. *mode* est une chaîne optionnelle permettant de spécifier dans quel mode le fichier est ouvert. Par défaut, *mode* vaut `'r'` qui signifie « ouvrir en lecture pour du texte ».

Document technique DT7 : Extrait documentation caméra et Serveur Edge

ZumIQ Edge Computer AIQ-PE2

Key Features

IQ Application Environment Onboard: Linux-based platform to deploy and run industrial applications (e.g., those created with Python, Node-RED, C++, Go); flexibility to develop on a maker platform and deploy on ZumIQ

Low Power Consumption: 2.2 W (max @ 12 VDC)

Computing Resources: 1 GHz ARM Cortex-A8 Processor, 1 GB RAM, 1 GB Storage

Solar-Compatible Operating Voltage: +6 to +30 VDC; can be powered by batteries, fuel cell, solar, wind turbines, or DC

Wide Operating Temperature Range: -40°C to +75°C

Flexible Interface Options: 2 Ethernet (10 / 100 / 1000 Mbps) and 2 Serial (RS232 / RS422 / RS485)

Caméra FLIR

Caractéristiques :

– **plage de températures** de l'objet -20 à +120 °C (-4 à 248 °F) ou 0 à +350 °C (32 à 662 °F) selon réglages

– **précision** ±2 °C ou ±2 % de la valeur lue

– **résolution IR** 320 × 240 pixels – 16 bits

– **sensibilité thermique/NETD** < 0,05 °C à +30 °C/50 mK

– **gamme spectrale** : 7.5µm à 13µm

– fréquence image 8-9 Hz

– connexion 100 Mb Ethernet compatible Ethernet/IP et modbus TCP

– On schedule: file sending (FTP) or e-mail (SMTP) of analysis results or images.

– On alarms: file sending (FTP) or e-mail (SMTP) of analysis results or images.

– distance minimale de focus : 0.4 m

– FOV 25°x18.8° mise au point motorisée et mise au point automatique.

– IRMonitor Software

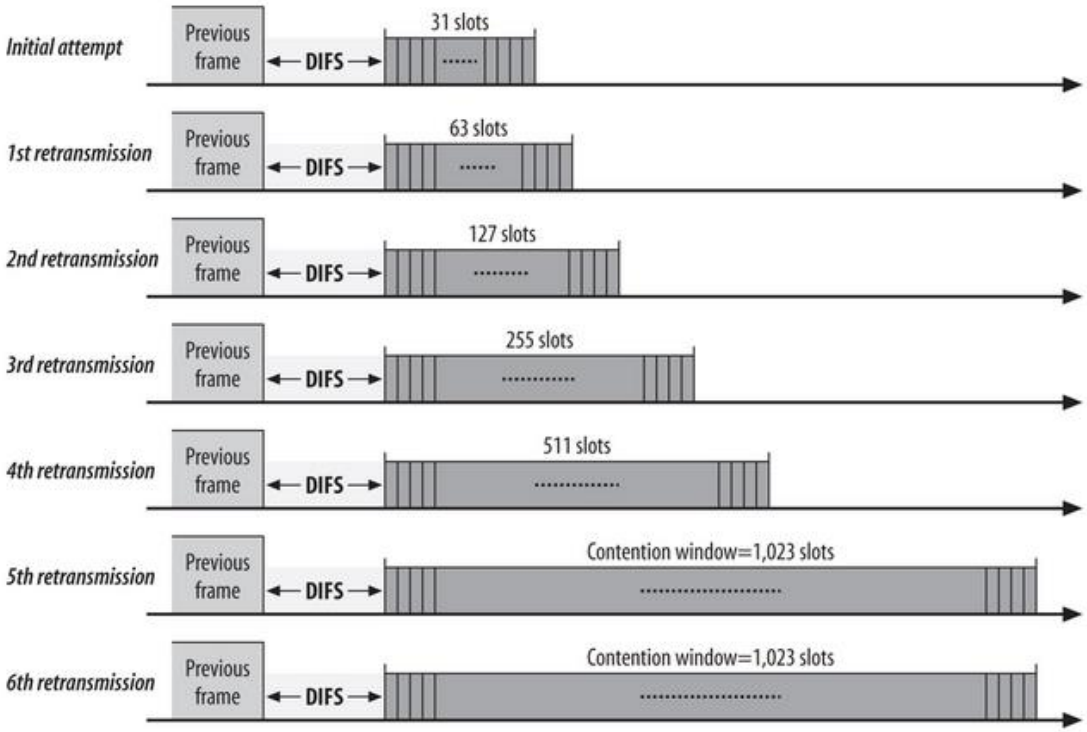
– constante de temps du détecteur : 12 ms

– External power operation : 12/24 V, 24 W max

– difference temperature alarms with a dynamically updated reference temperature.

Tournez la page S.V.P.

Document technique DT8 : Temps de Back-off Wi-Fi



Document technique DT9 : Algorithme de Dijkstra

L'algorithme de Dijkstra est un algorithme pour calculer les chemins les plus courts à partir d'un nœud vers les autres nœuds dans un réseau. Le nombre d'étape est égale au nombre de nœuds dans le réseau. Il maintient une liste L des nœuds déjà examinés, c'est-à-dire les nœuds pour lesquels le chemin le plus court est déjà trouvé. À l'étape initiale, la liste L contient seulement le nœud source. Le coût vers les nœuds à un saut est rempli dans le tableau tandis que le coût vers les autres nœuds reste infini comme illustré dans la Figure suivante.

L	1	2	3	4	5	6	7	8	9	10	11	12	13
4	-	30	45		-	95	20	-	-	-	-	-	-

Le nœud ayant le coût le plus court (nœud 7) est choisi comme le nœud déjà examine et ajouté dans la liste L. Dans chacun des étapes suivantes, l'algorithme essaie d'aller un saut plus loin à partir du nœud qui vient d'être ajouté à la liste L. Les coûts des chemins vers les nœuds hors de la liste L sont mis à jour si en passant par ce nœud l'algorithme trouve un meilleur chemin. Après la mise à jour, le nœud ayant le coût le plus faible à partir du nœud source est ajouté à la liste L. Ainsi de suite, à chaque étape, un nouveau nœud est considéré comme le nœud déjà examiné et ajouté à la liste L, jusqu'à ce que tous les nœuds du réseau se trouvent dans la liste L.

Modèle CMEN v3

Nom de famille :
(Suivi, s'il y a lieu, du nom d'usage)

Prénom(s) :

Numéro Candidat :

Né(e) le : / /

Cadre réservé aux candidats de concours de recrutement et examens professionnels

Concours : ..... **Option / Section :** ..... **N° d'inscription :**

Cocher une seule case parmi les six types de concours suivants :

☐ externe ☐ 3e externe ☐ externe spécial ☐ interne ou 1er interne ☐ 2nd interne ☐ 2nd interne spécial

Cocher public OU privé UNIQUEMENT pour les concours enseignants : ☐ public ☐ privé

Examen professionnel pour l'avancement au grade de : .....

Cadre réservé aux candidats d'examens et du concours général

Examen : ..... **Série / Spécialité :** .....

Epreuve - Matière : ..... **Session :** .....

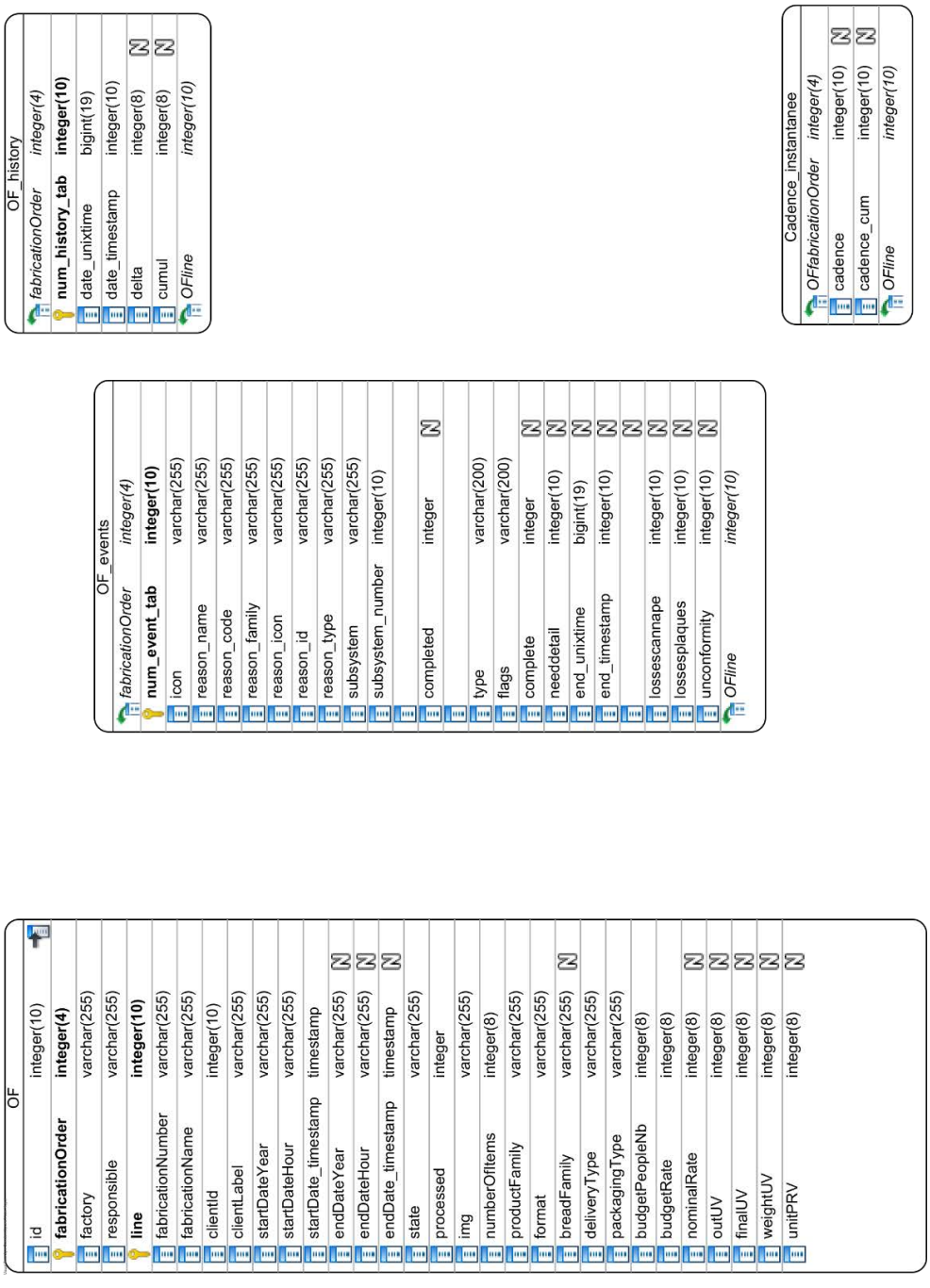
EAE SIN 2

DR1 - DR2 - DR3 - DR4

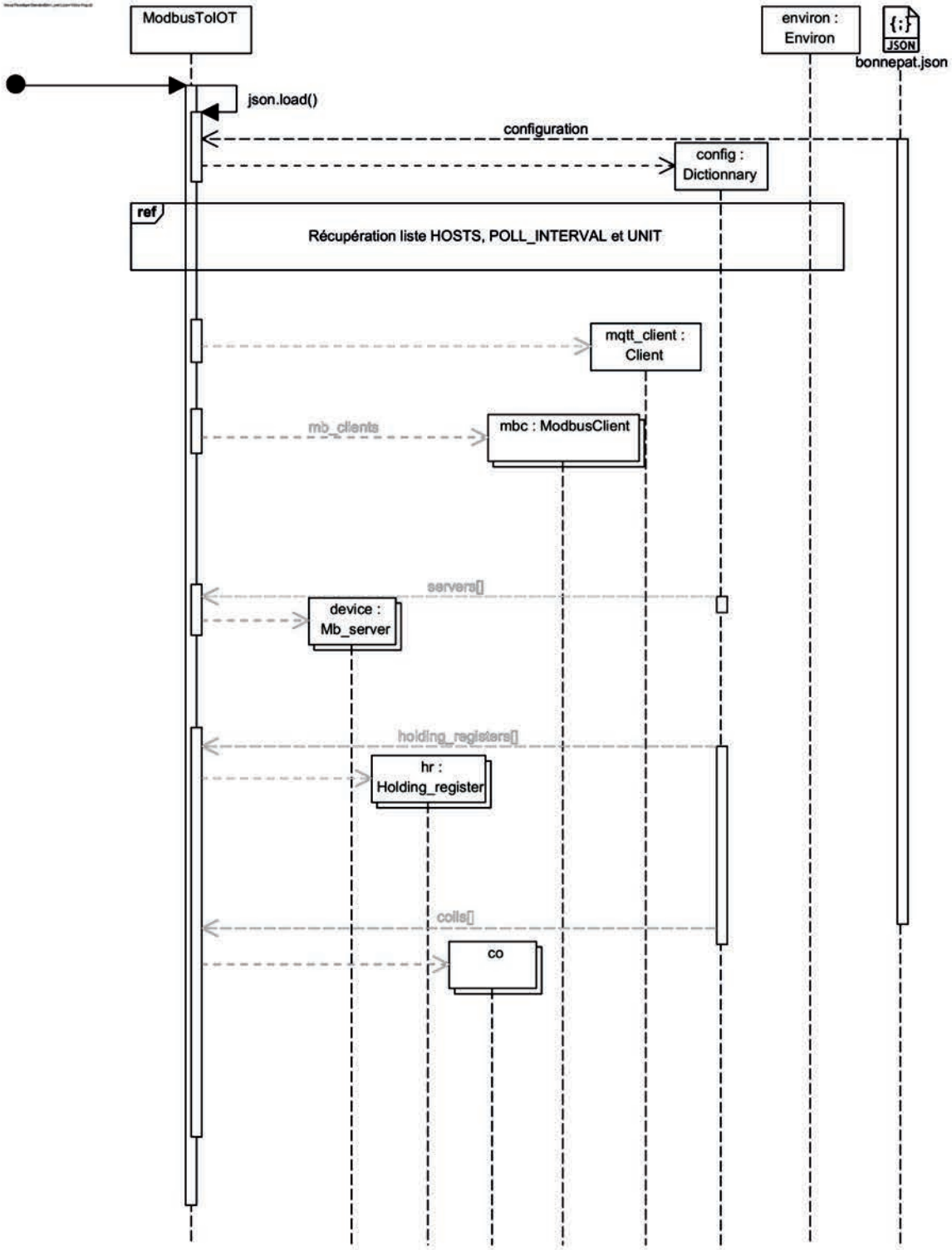
Tous les documents réponses sont à rendre, même non complétés.

NE RIEN ECRIRE DANS CE CADRE

Document réponse DR1

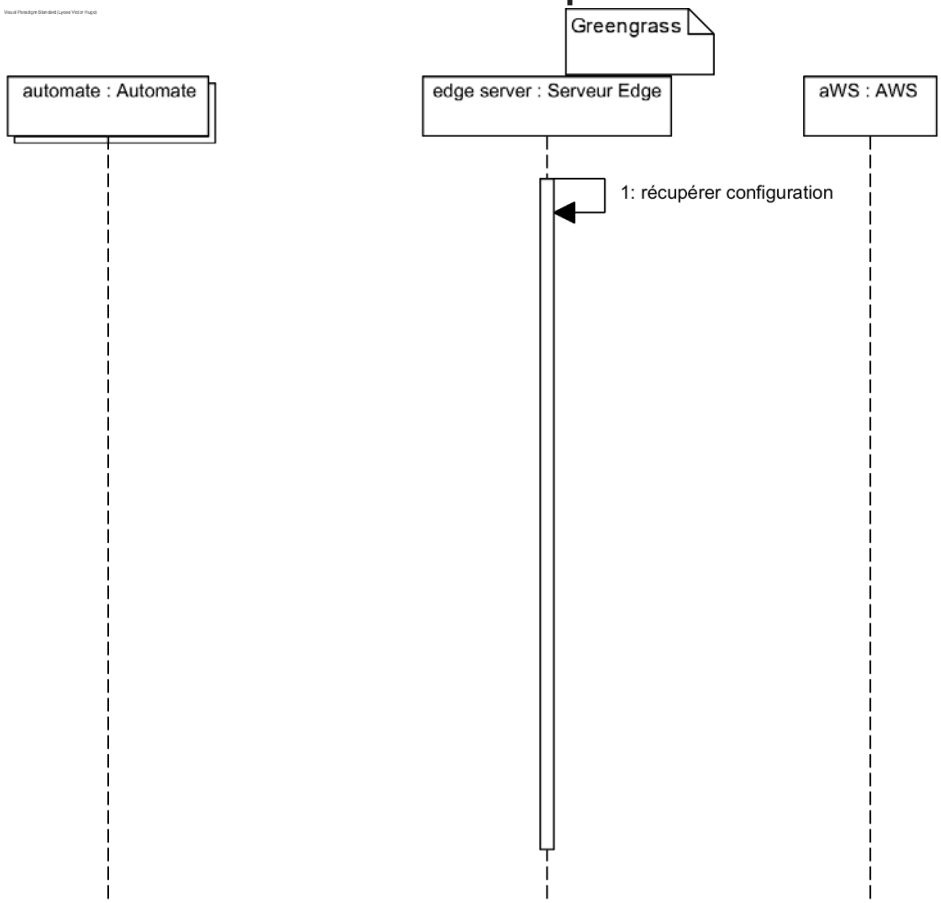


Document réponse DR2



Tournez la page S.V.P.

Document réponse DR3



Document réponse DR4

L	1	2	3	4	5	6	7	8	9	10	11	12	13
4	-	30	45		-	95	20	-	-	-	-	-	-