Rapport de projet temps réel 4IS

Nom du binôme :

Enseignant de TP :

— Ce qu’il faut faire —

**Consignes**

Le rapport est à rendre en pdf et en word..

Vous devez :

1) l’envoyer par mail à votre encadrant de TP au plus tard une semaine après votre dernière séance de TP, et

2) le déposer sous moodle. Le nom du rapport portera les deux noms du binome.

Vous devez aussi rendre votre code (uniquement les fichiers que vous avez écrits ou modifiés) sous la forme d’une archive (zip ou tar). Le nom du zip portera les deux noms du binôme.

**Critères d’évaluation** :

— Qualité rédactionnelle,

— Exhaustivité et justesse des règles de codage,

— Qualité de la conception (clarté, respect de la syntaxe, exhaustivité, justesse),

— Qualité des explications et justifications.

**Compétences évaluées :**

— rédaction et communication sur un dossier de conception

— concevoir une application concurrente temps réel : analyse fonctionnelle et proposition d’une architecture en tâches

— passer d’un modèle de conception à une implémentation

— écriture de code et utilisation des services d’un système d’exploitation temps réel

— vérification et validation

NB : pour le rapport final, vous pourrez supprimer tout ce qui est écrit en gris dans le document

* Mission et analyse opérationnelle

Système d’intérêt : quel est le système d’intérêt ?

*(positionner le système d’intérêt par rapport au système gyropode)*

Mission (service ou fonction que fournit le système) : quel est le problème à résoudre ?

*(A quoi sert le système ?)*

Contexte organique (point de vue structurel) :

Positionner, à l’aide d’un diagramme, le système d’intérêt dans son contexte d’utilisation (celui du TP), préciser ses frontières et ses interactions avec les constituants externes, ainsi que ses interfaces (liens physiques, sous forme de liens dirigés et nommés).

Contexte fonctionnel :

Même chose, mais en examinant les fonctions des constituants du contexte d'utilisation.

Flux d'entrées et de sorties entre les fonctions des constituants du contexte et la fonction ou mission du système étudié.

Scénarios opérationnels point de vue comportemental) :

Considérer les différentes phases d’utilisation du système (démarrage, phase opérationnelle, arrêt).

Pour la phase opérationnelle, décrire le comportement du système, intégrant le comportement nominal (où tout se déroule bien) et les scénarios non nominaux (comme l’arrêt d’urgence).

*Description textuelle et/ou à l’aide d’un modèle des scénarios (diagramme d’activité par exemple dans lequel on modélise les différentes activités à mener en parallèle, etc…)*

2 Expression des exigences

Exprimer sous forme de table les **exigences du système**.

S’aider pour cela des documents sous Moodle pour éliciter et spécifier les exigences, les caractériser (fonctionnelle ou non-fonctionnelle), les raffiner si nécessaire, les vérifier, les valider. Numéroter les exigences de façon unique.

Proposition d’une typologie indicative d’exigences (une typologie plus détaillée est disponible sous Moodle) :

**Exigences fonctionnelles (**issues de la mission du système)

**Exigences de performance (**quantifient les exigences fonctionnelles majeures)

**Exigences d'interfaces (fonctionnelles**: échanges de matière, énergie, information**, ou physiques**: connexions**)**

**Exigences opérationnelles**

* Modes opérationnels et scénarios opérationnels
* Exigences d'Ergonomie (et protocole d'échanges avec les opérateurs & utilisateurs)
* Exigences de Sûreté de Fonctionnement (sécurité, fiabilité, maintenabilité)

**Contraintes** (dimensions physiques, technologies, maintenance, règlements, standards)

* Contraintes de conception et de réalisation
* Contraintes physiques (dimensions)
* Contraintes de mise en service, de montage
* Contraintes de maintenance
* Contraintes de retrait de service

**Exigences de validation**

Table des exigences :

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro exigence | concerne | Description de l’exigence | Type |
|  |  |  |  |
| E1 |  |  |  |
| E2 |  |  |  |
| E2.1 |  |  |  |
| E2.2 |  |  |  |
| E3 |  |  |  |
|  |  |  |  |

Dictionnaire : (*si besoin*)

|  |  |
| --- | --- |
| Terme | Définition |
|  |  |
|  |  |
|  |  |
|  |  |

Exemple :

Dans le sujet, il est écrit : « L’arrêt d’urgence est déclenché par le superviseur pour des raisons de sécurité (détection de l’absence du pilote), si le niveau de batterie est trop faible ou si l’angle ẞ reste sort de l’intervalle (-20, +20) degrés. »

On en déduit 3 exigences, que l’on classe dans la catégorie des exigences fonctionnelles :

* E1, concerne l’arrêt d’urgence, « L’arrêt d’urgence est déclenché par le superviseur en cas de détection de l’absence du pilote »
* E2, concerne l’arrêt d’urgence, « L’arrêt d’urgence est déclenché par le superviseur si le niveau de batterie est trop faible »
* E3, concerne l’arrêt d’urgence, « L’arrêt d’urgence est déclenché par le superviseur si l’angle ẞ reste sort de l’intervalle (-20, +20) degrés »

Ces exigences ne sont toutefois pas assez précises et restent parfois ambiguës :

Comment détecte-t-on l’absence du pilote et qui le détecte ? Quel signal véhicule cette information ? Comment parvient-il au superviseur ? Dans quel délai l’arrêt est-il déclenché par le superviseur, est-ce à partir de la réception de ce signal ? Quel est le signal émis par le superviseur en cas d’arrêt d’urgence ? A qui est-il envoyé ? En quoi consiste concrètement l’arrêt d’urgence ? Que signifie « le niveau de batterie est trop faible » ? etc.

🡺 il faut donc procéder à une explicitation des exigences, par des exigences dérivées, comme par exemple :

E2.1 : le superviseur surveille périodiquement (*indiquer la fréquence*) le niveau de la batterie.

E2.1 : quand le niveau de la batterie est inférieur ou égal à 15% de sa charge maximale alors le superviseur émet un signal d’arrêt d’urgence

E2.3 : l’émission d’un signal d’arrêt d’urgence consiste à envoyer ‘très régulièrement’ (*à préciser*) des ‘trames d’arrêt’ (*à préciser*) au STM32 (*préciser via quelle interface*)

Etc.

Ce travail, une fois terminé, permet de spécifier très clairement l’ensemble des exigences du superviseur.

3 Architecture fonctionnelle

L’objectif est à cette étape de décrire comment le système d'intérêt fonctionnera pour répondre aux attentes, d’un point de vue fonctionnel, dynamique, comportemental et temporel.

À l’aide du tableau suivant, **recenser et expliciter les fonctions** que le superviseur devra assurer, en précisant les entrées et sorties de chaque fonction (une fonction transforme un flux de données d’entrée en flux de sortie, en consommant du temps et des ressources).

*(NB : Le nombre de lignes du tableau n’a aucune valeur de référence du nombre de fonctions à recenser*)

Recensement des fonctions :

|  |  |  |  |
| --- | --- | --- | --- |
| Nom de la fonction | Description du comportement | Entrées | Sorties |
|  |  |  |  |
| Recevoir\_STM32 | Reçoit les messages envoyés par le STM32 |  |  |
| Envoyer\_STM32 |  |  |  |
| … |  |  |  |
|  |  |  |  |

Définition de l'agencement des fonctions et de leurs interfaces :

Une fois chaque fonction décrite individuellement, **proposer un diagramme d’architecture fonctionnelle** montrant l’ensemble des échanges entre ces fonctions (caractérisation des interfaces, des flux de données par les échanges d’entrées et sorties, flux de contrôles (activations, synchronisations). C’est une vue structurelle (statique).

Architecture fonctionnelle statique :

*Description des fonctions que le système doit exécuter pour remplir sa mission et satisfaire ses scénarios opérationnels.*

*Description des flux internes d'entrées/sorties (échanges entre les fonctions du système).*

*(voir exemple dans le sujet)*  Une image contenant texte, carte

Description générée automatiquement

Définition de la séquence d'exécution des fonctions :

Reprendre la description de chaque fonction (du tableau) et décrire formellement le comportement de chaque fonction, puis la séquence globale d’exécution des fonctions (vue comportementale - dynamique).

Architecture fonctionnelle dynamique : (optionnel)

*Scénarios et modèles dynamiques, incluant les fonctions précédentes avec leurs flux d'entrées sorties.*

*Le ou les modèles dynamiques montrent les flux de contrôles des fonctions et les flux d'entrées/sorties du système afin que le système exécute correctement sa mission et ses scénarios opérationnels.*

4 Architectures logique et physique

Décrit comment le système d'intérêt est organisé, sous forme de composants abstraits puis physiques, compte tenu du choix de Xenomai comme support d’exécution.

Choix et justification d’une organisation en constituants (découpage/regroupement des fonctions en tâches), caractérisation des tâches :

À l’aide du tableau suivant, identifier et caractériser un ensemble de composants (tâches) et de leurs interfaces logiques (entrées et sorties)

*(NB : Pensez à utiliser les règles de nommages proposées pour le TP)*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nom de la tâche | Rôle | Entrées | Sorties | Activation / Période | Priorité |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Allocation des fonctions aux composants :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tâche \ Fonction | Recevoir\_STM32 | Envoyer\_STM32 | Fonction i | Fonction j | Justification |
|  |  |  |  |  |  |
| T\_Recevoir\_STM32 |  |  |  |  |  |
| T\_Envoyer\_STM32 |  |  |  |  |  |
| … |  |  |  |  |  |

Exemple :

Une fois que vous avez vos fonctions, par exemple surveiller\_batterie, surveiller\_angle\_beta, et calculer\_couple, vous vous posez des questions :

- si par exemple vous surveillez la batterie et l’angle beta à la même fréquence, vous pouvez faire une seule et même tache périodique, sinon il en faudra deux ;

- pour calculer\_couple, si la tâche est activée dès qu’elle reçoit les informations utiles du STM32, alors elle ne sera pas périodique, mais sera en attente active (avec un timeout bloquant) sur réception d’une message queue (par exemple…) dans laquelle le STM32 envoie les informations….

Diagramme d’architecture logique statique :

Initier le diagramme d’architecture en positionnant les composants et leurs entrées/sorties.

Allouer les exigences système aux différents composants, en assurant la traçabilité.

Table des exigences sur chaque tâche :

|  |  |  |
| --- | --- | --- |
| Numéro exigence système | Concerne la tâche | Description |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Choix et justification des moyens de communication et de synchronisation

Consiste à identifier les moyens de communication et de synchronisation entre les tâches.

|  |  |  |  |
| --- | --- | --- | --- |
| Donnée échangée entre des tâches  (voir diagramme d’architecture logique statique) | Mode de communication (variable globale, messageQueue, …) et caractérisation  (Id, nom, taille, timeout, ..) | Protection (ou pas) par Mutex et Id du Mutex | Justification |
|  |  |  |  |
| Niveau de batterie | Variable Gobale :  Niveau\_Batterie | Mutex SemM\_Batterie | Accédée par les 3 tâches (envoi interface graphique, monitoring batterie et recevoir\_STM32) |
| Données envoyées au STM32 | File de message : NOM\_FILE | X | On souhaite envoyer le couple, l’angle, etc. sans ralentir le superviseur ni perdre de l’information |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Par exemple, si le niveau de batterie est partagé en lecture ou en écriture entre plusieurs tâches, on peut choisir de le représenter par une variable globale Niveau\_Batterie, protégée par un mutex SemM\_Batterie si on redoute des accès concurrents à la variable.

|  |  |  |  |
| --- | --- | --- | --- |
| Tâches à synchroniser | Événement à signaler | Id du sémaphore binaire qui représente l’événement | Justification |
|  |  |  |  |
| La tâche de monitoring de la batterie et la tâche de gestion de l’arrêt d’urgence | Niveau de batterie faible | SemB\_Batterie\_Faible | On souhaite que l’arrêt d’urgence soit déclenché dès qu’on détecte un niveau de batterie trop faible |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Par exemple, si vous avez 3 tâches de monitoring (angle Beta, niveau de batterie et chute du pilote), dans un premier temps, chaque tâche peut alerter une tâche qui déclenche un arrêt d’urgence, indépendamment de la raison de la demande d’arrêt ; vous pouvez pour signaler la demande d’arrêt d’urgence utiliser un sémaphore binaire, qui est libéré par l’une des 3 tâches et sur lequel la tâche de déclenchement de l’arrêt d’urgence est en attente.

Transformer le diagramme d’architecture logique en un **diagramme d’architecture physique**.

NB :

* voir les règles de nommages suggérées dans le fichier « Annexes TP » sous Moodle
* voir le formalisme et l’éditeur proposés sous Moodle pour représenter les sémaphores et files de messages

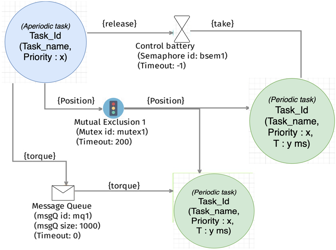
Architecture physique statique :

*Reprendre le diagramme d’architecture logique statique décrivant les constituants du système étudié (ici les tâches) et modifiez-le pour préciser les choix d’implantation physique (dépendant de la plateforme d’exécution) pour supporter les liens d’interface internes et externes (entrées/sorties des tâches) au vu des choix d’implantation opérés pour les moyennes de communication et de synchronisation.*

Décrire le comportement de chaque tâche ainsi que le comportement global dans le cas des scénarios décrit lors de l’analyse opérationnelle.

Architecture physique dynamique :

*Scénarios et modèles dynamiques, représentant dans le temps l’exécution des tâches avec leurs flux de contrôles et flux d'entrées/sorties.*

*(voir exemple dans le sujet)* 

5 Codage et livraisons incrémentales

Reprendre la conception (diagramme d’architecture physique et justifications) après retro-ingénierie du code fourni pour intégrer les contraintes de plateforme.

Diagramme d’architecture physique revu en fonction des contraintes de plateforme :

Pour le développement et la validation de l’application, vous devez décrire :

* La stratégie de codage, vérification et intégration,
* Le partage du travail dans l’équipe,
* Les livrables.

Décrire la stratégie de codage et intégration :

Stratégie de codage et d’intégration

6 Analyse et validation du logiciel par rapport aux exigences

Pour chacune des exigences attendues, indiquez si elle a été réalisée. Ajoutez d’éventuelles remarques si le fonctionnement n’est pas exactement celui attendu, ou expliquant les difficultés rencontrées, justifiez le respect des propriétés temporelles.

|  |  |  |
| --- | --- | --- |
| Numéro exigence | Description de l’exigence | État |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

7 Bonus

Une fois que vous aurez assuré cette première version du superviseur, vous allez réfléchir à une gestion un peu plus ‘fine’ des cas d’arrêt d’urgence. Par exemple, nous allons éviter, quel que soit le scénario, d’arrêter brutalement les moteurs. Vous pouvez ainsi envisager, pour un dépassement de l’angle Beta, de mettre en place une tâche qui va ramener progressivement l’angle dans une plage acceptable, en prenant la main sur la loi de commande, et en suspendant l’exécution de la calcul de couple provisoirement ; pour la supervision du nouveau de batterie, vous pouvez imaginer, à 50% de la charge de passer d’autorité dans une mode de conduite économe en énergie, puis, en deçà de 15%, de décélérer progressivement pour permettre au pilote d’amener le gyropode à la station de recharge.

Vous allez ainsi remettre en question une partie de votre architecture pour mettre en œuvre ces nouvelles stratégies.

8 Commentaires

Dans cette partie indiquer toute remarque relative au déroulement du projet.