

MI11

Systèmes temps réel critique

Jérôme De Miras

Poste : 59 02
e-Mail : demiras@hds.utc.fr

MI11



1


Bibliographie

« Real-Time Systems : design principles for distributed embedded applications

Hermann Kopetz

Kluwer Academic Publishers

MI11



Printemps 2017

2

Panorama général

MI11



3

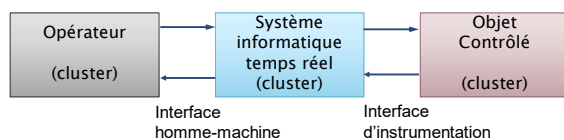
Définitions

Définitions de base

- Un système informatique temps réel doit être évalué non seulement sur la validité des résultats logiques qu'il produit mais aussi par rapport au moment où il est capable de les délivrer
- Un système temps réel doit répondre aux stimuli de son environnement dans un temps donné : l'**échéance**
- En fonction des contraintes sur les échéances, un système informatique temps réel peut être dur (hard) ou mou (soft)

Clusters

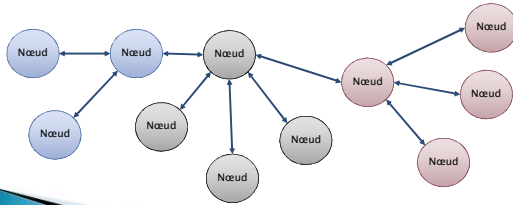
- Un système informatique temps réel n'est qu'une partie d'un système plus large qui dépend du temps et qui se compose de *clusters*.



Systèmes distribués

□ Un système temps réel peut être distribué :

- Nœuds
- Lien de communication temps réel



Exigences

Exigences

- A : Exigences fonctionnelles : les fonctions que doit remplir un système informatique temps réel
- B : Exigences temporelles : les contraintes de temps à respecter
- C : Exigences de fiabilité (dependability)

A

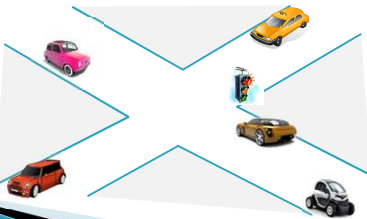
Exigences fonctionnelles

- ❑ Les fonctions que doit remplir un système informatique temps réel peuvent se grouper en trois classes :
 - ❑ collecte de données
 - ❑ pilotage de l'environnement
 - ❑ interactions homme machine

A

Collecte de données

- ❑ Un système change d'état en fonction du temps
- ❑ Si on arrête le temps, l'état peut être décrit par un certain nombre de valeurs (position, vitesse, couleur du feu, ...)



A

Entités TR Collecte de données

- ❑ Seul un sous ensemble des variables sont pertinentes pour l'application :
Entités Temps Réelles
- ❑ Chaque entité TR appartient à une sphère de contrôle où elle peut être modifiée
- ❑ En dehors de cette sphère, entité TR uniquement observable

A

Images TR

Collecte de données

Observer et collecter ces entités TR est l'exigence N°1 : **Images Temps Réelles**

Validité temporelle limitée des images TR

L'ensemble de images TR valides est appelé Base de données Temps Réelle

Mise à jour à chaque évolution des entités TR

Observation « Time Triggered »

Observation « Event Triggered »

MI11

utct

Printemps 2017

13

A

Données agréées

Collecte de données

Données

Traitements

BDTR

Conditionnement

Système de détection de faute

Image TR correcte = données agréée

Gestion d'alarmes (importance du temps pour aider l'opérateur à trouver l'origine du défaut)

MI11

utct

Printemps 2017

14

A

Relation au temps

Collecte de données

Toute propriété d'une entité TR qui reste valide pendant un intervalle est un **attribut d'état**

Un changement d'état est un **événement**

Une observation est un événement qui permet l'acquisition de l'état d'une entité TR à un instant donné

L'horloge découpe la ligne de temps en portion égales appelé **granules de temps de l'horloge**

Les granules de temps sont séparés par des événements : les **ticks d'horloge**

Un trigger est un événement qui déclenche une action

MI11

utct

Printemps 2017

15

A

Contrôle numérique

- Calculer le point de fonctionnement pour un actionneur en fonction de la BDTR
- Tâche répétitive et régulière
 - ⇒ problème de l'automatique
 - ⇒ exigences temporelles

A

Interactions homme-machine

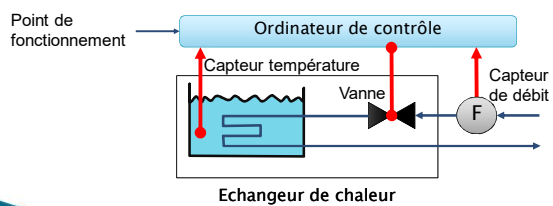
- Un système temps réel doit informer l'opérateur de l'état du système contrôlé
 - ⇒ Système critique majeur
- Une bonne partie des accidents sur des applications critiques ont eu lieu à cause de l'IHM

B

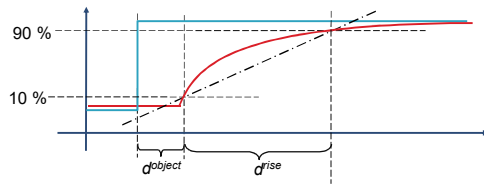
Exigences temporelles

- Elle proviennent en grande partie des besoins de performances des boucles de régulation

- Exemple : échangeur de chaleur

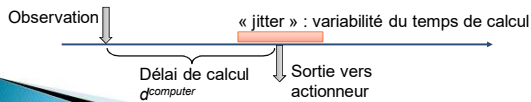


B Exemple : échangeur de chaleur



- Le contrôleur doit échantillonner la mesure

$$d_{\text{sample}} < d_{\text{rise}} / 10$$



B Résumé exigences temporelles

Variables	Sphère de contrôle	Relations
d_{object}	Délai de l'objet contrôlé	Process physique
d_{rise}	Temps de réponse	Process physique
d_{sample}	Période d'échantillonnage	$d_{\text{sample}} \ll d_{\text{rise}}$
d_{computer}	Temps de réponse calculateur	$d_{\text{computer}} < d_{\text{sample}}$
$\Delta d_{\text{computer}}$	Gigue sur le calcul	$\Delta d_{\text{computer}} \ll d_{\text{computer}}$
d_{deadtime}	Temps mort	$d_{\text{computer}} + d_{\text{object}}$

- Critères :
 - On suppose la gigue minimal car en contrôle on sait compenser les délais fixes
 - La gigue apporte du bruit sur les mesures
 - La détection d'une erreur doit se faire dans un temps compatible avec la période d'échantillonnage (minimiser la latence de détection qui dépend de la gigue)

⇒ minimiser les giges (jitter)

C Exigences de fiabilité

- Exigences de fiabilité (dependability)

- fiabilité (reliability)
- sûreté (safety)
- maintenabilité (maintainability)
- disponibilité (availability)
- Sécurité (security)

C

Fiabilité (reliability)

- $R(t)$: probabilité que le système fournisse le service au temps t sachant son état au temps t_0

$$R(t) = \exp(-\lambda(t - t_0))$$

λ : taux de pannes

$1 / \lambda$: MTTF (mean time to failure)

C

Sûreté (safety)

- Modes critiques des pannes
- Une panne critique est dite maligne
- Un système « safety critical » doit avoir un taux de panne extrêmement bas (10^{-9} panne/h)
- Certification du système :
 - Sous-système critique protégés par des interfaces stables qui empêchent la propagation d'une erreur
 - Tous les scénarios sont gérables sans argument probabiliste \Rightarrow design des ressources
 - L'architecture est compatible avec un processus de certification incrémental

C

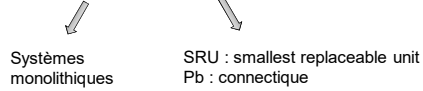
Sûreté (2)

- Propriétés requises :
 - Un modèle de sûreté peut être produit ou tous les paramètres non déduits analytiquement sont mesurables en un temps raisonnable
 - Le modèle de fiabilité n'inclut pas de transition d'état correspondant à un défaut de conception \Rightarrow à prouver analytiquement
 - Choisir un design qui minimise le modèle

C Maintenabilité (maintainability)

- Temps moyen de réparation après une panne
Probabilité $M(d)$
 d : intervalle de temps pour la réparation
MTTR : mean time to repair

Conflit entre fiabilité et maintenabilité



C Sécurité (security)

- Un système ne doit pas permettre l'intrusion
- Difficile à quantifier et à spécifier
- Intégrer des systèmes de cryptage

Classification des systèmes temps réel

« Hard real time » contre « Soft real time »

Caractéristique	Hard real time	Soft real time
Temps de réponse	sans compromis	violation possible
Performance sur pic de charge	prédictible	dégradation
Vitesse de conduite	environnement	calculateur
sûreté	souvent critique	non critique
Taille des fichier de données	petit / moyen	large
Type de redondance	active	points de restauration
Intégrité des données	court terme	long terme
Détection des erreurs	autonome	assistée

« fail safe » contre « fail-operationnel »

- ❑ Fail-safe :
 - ❑ en cas de panne le système peut atteindre un état sûr
 - ❑ caractéristique de l'objet et non du système informatique
 - ❑ la détection d'erreur doit avoir une couverture très large ⇒ la gestion TR améliore la disponibilité
- ❑ Fail-operationnel :
 - ❑ Fonctionner quoi qu'il arrive

« Réponse garantie » contre « meilleur effort possible »

- ❑ Le design exclu toute notion de réponse probable (fautes et pics de charge inclus)
⇒ Garantie sur la réponse
- ❑ Sinon on fait au mieux
⇒ Validation du design par des tests (économiquement plus attrayant)

« ressources adéquates » contre « ressources inadéquates »

- Un système TR critique a besoin d'être dimensionné pour le pire cas

⇒ Économiquement peu enviable

« Event triggered » contre « time triggered »

- Deux approches possibles en fonction du mode de déclenchement des communication et de l'activité de calcul
- Event triggered : le mécanisme de l'interruption permet de répondre à l'événement
⇒ ordonnancement dynamique
- time triggered : seule la progression du temps fait évoluer le système
⇒ disposer d'un temps global (granularité ?)

Systèmes temps réel embarqués

- Eléments pertinents pour le design d'un système
 - Production de masse
 - Structure statique
 - Interface homme machine
 - Support de stockage du logiciel
 - Stratégie de maintenance
 - Capacité de communication

Trois exemples

- ❑ Régulateur de débit
- ❑ Moteur thermique
- ❑ laminoir

