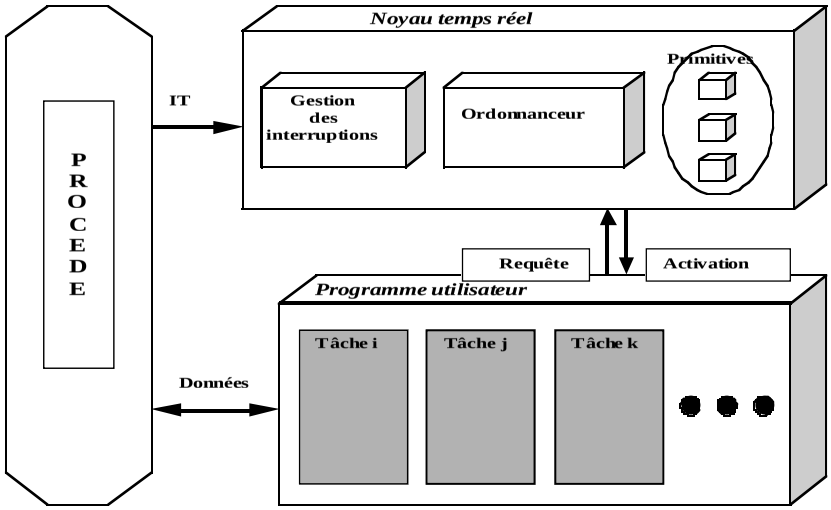
2. STRUCTURES D'ACCUEIL POUR LE TEMPS RÉEL

2.1.1. EXÉCUTIFS TEMPS RÉEL "EMBARQUÉS" SCHÉMA D'UN EXÉCUTIF TEMPS RÉEL.



Exécutifs de petite taille, basés sur les priorités, pour les applications embarquées

Références : **[Cottet 2000], [Kaiser 2001], [Trinquet 1999]**

2.1.2. EXÉCUTIFS TEMPS RÉEL "EMBARQUÉS"

• Gestion de l'interface avec le matériel (entrées-sorties industrielles, interruptions)

• Services de gestion du temps, de messages, de synchronisation entre tâches, de partage des ressources.

• Ordonnancement des tâches relativement simple (généralement avec des priorités fixes ; avec des priorités variables dans les exécutifs les plus récents) et dans un contexte monoprocesseur.

• De plus en plus : Traitement de l'inversion de priorité et de prévention de l'interblocage

• "Base de données temps réel" en mémoire

• Noyau préemptif et réentrant, découpé en ressources critiques

Exemples : VxWorks, pSOS, VRTX, QNX, iRMX, Lynx-OS,... Normalisation : POSIX Temps réel (P1003.4)

TYPE D'APPLICATION CONVENANT BIEN À LEUR UTILISATION :

fonctionnement déterministe analysable hors ligne (par ex. méthode RMA)

• Modèle de tâches simple et déterministe : date de déclenchement (périodique le plus souvent), durée d'exécution nominale (maximale), échéance de fin d'exécution.
• Tâches périodiques ou apériodiques = processus léger,
• Serveur périodique pour traiter les requêtes apériodiques
• Nombre fixe de tâches, créées au lancement de l'application
• Priorités fixes (RM, selon la période) ou variables (EDF, selon l'échéance).
• Peu ou pas d'allocation dynamique d'objets

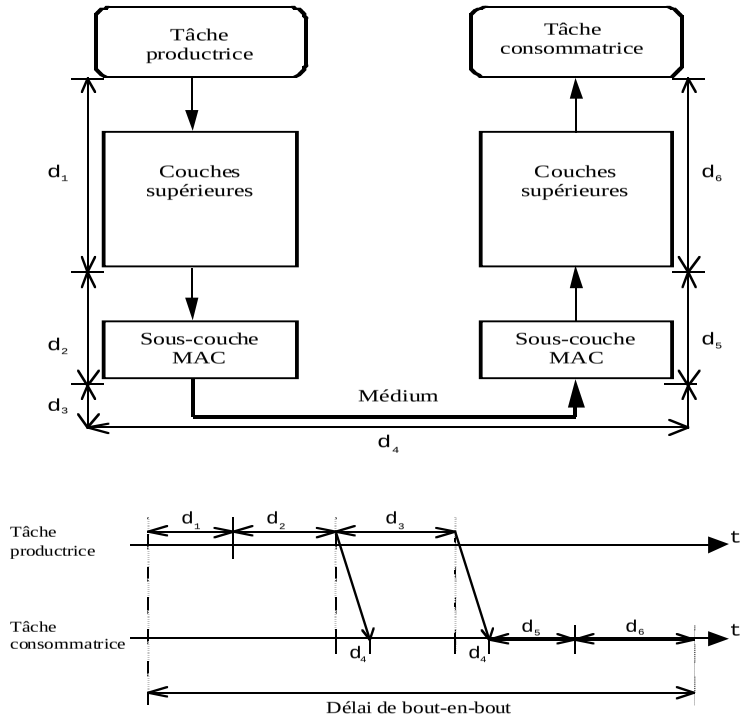
2.2. RÉSEAUX LOCAUX TEMPS RÉEL

• Réseau de type bus à jeton (anneau logique et trame spéciale appelée jeton, norme IEEE 802.4, MAP),
• Réseau FIP (bus de terrain avec arbitre central et schéma producteur-consommateur),
• Réseau CAN (bus pour véhicules, réseau synchrone avec rythme fourni par le bus),
• Réseau de type FDDI (boucle optique avec jeton circulant).
• Protocoles spécifiques (MAC, "Medium Access Control").
Il existe des stratégies déterministes (ex CSMA/CA , "collision avoidance" ou CSMA déterministe).
• Priorités pour messages

• On sait évaluer les pires cas de transfert d'un message prioritaire [Cottet 2000].

délai de communication d'un message entre tâches distantes = d1 + d2 + d3 + d4 + d5 + d6
d1 : délai de traversée des couches supérieures
d2 : délai d'attente au niveau de la couche MAC (medium access control) de la machine émettrice
d3 : délai de transmission (émission ou réception) des bits du message
d4 : délai de propagation de chaque bit sur le médium (entre la source et la destination)
d5 : délai de réception et d'attente au niveau de la couche MAC (medium access control) de la machine réceptrice
d6 : délai de traversée des couches supérieures

Cela autorise un ordonnancement conjoint de tâches et de messages



Délais intervenant dans la communication entre deux tâches distantes.

2.4. TENDANCES ACTUELLES

EXTENSIONS TEMPS RÉEL DES SYSTÈMES CLASSIQUES

Les défauts des systèmes classiques
• Trop grande variabilité dans le fonctionnement => incertitudes temporelles:
• Ordonnancement équitable en moyenne, allocation dynamique et synchronisation, E/S engendrant des temps morts, gestion des interruptions non optimisées, gestion de mémoire virtuelle,
• Résolution temporelle insuffisante pour les temporisateurs ("timers")
• Temps de réponse à un événement externe non prévisible, car le noyau n'est ni préemptif, ni reentrant.
• Optimise l'utilisation des ressources et non le temps de réponse des clients qui ont des contraintes de temps. Ne sait pas reconnaitre ces clients.
• Appel système non préemptif, suppose un disque, création dynamique d'espace d'adressage (process)

• Système temps réel par réécriture d'un noyau préemptif et réentrant, découpé en ressources critiques
réécriture partielle : RT-Unix, RT-Mach, RT-Linux
réécriture totale : Chorus, LynxOs
On retrouve la norme POSIX temps réel P1003.4

UTILISATION DES RÉSEAUX GÉNÉRAUX AVEC ADAPTATIONS POUR LE TEMPS RÉEL

" Best effort" insuffisant pour le temps réel
• Approche qualité de service QoS : contraintes sur le délai de transfert, sur la gigue et sur le taux de perte
Avec ATM : CBR "constant bit rate", RT-VBR "real-time variable bit rate"
Avec TCP/IP, protocoles RTCP "real-time control protocol", RSVP "resource reservation protocol", Diff-Serv, Int-Serv, QoS routing...
Ce n'est cependant pas la panacée pour le temps réel strict avec contrôle d'échéance de bout en bout

UTILISATION DES COMPOSANTS, SYSTÈMES ET PLATEFORMES COMMUNES (COTS)

• Simplifier la réalisation des systèmes coopérants : utiliser l'invocation à distance

• Construire des applications temps réel par coopération de systèmes embarqués hétérogènes via internet

• Augmenter la productivité de développement et diminuer le temps de cycle : intégration de composants

• Utiliser les matériels et logiciels ou de standards généraux sur étagères ("common-off-the-shelf" : COTS))

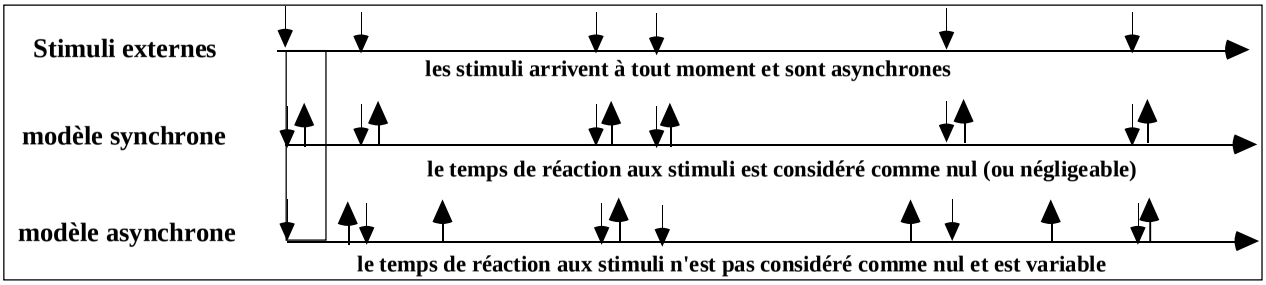
• Réduire les coûts de maintenance et d'évolutivité : profiter du potentiel d'évolution des composants

GRANDES INSUFFISANCES DES PLATES-FORMES CONVENTIONNELLES

•• Pas de moyens de programmation temps réel
•• Performances inadéquates et absence de déterminisme (prédictivité des opérations)
•• Pas de service de bout en bout intégrant réseaux et systèmes
•• Pas d'interfaces pour spécifier la QoS et l'allocation ou réservation de bande passante
•• Inefficacité de gestion des interfaces, des E/S et des protocoles
•• Coûts de gestion de l'hétérogénéité (temps d'attente dans les files, pertes de messages)
•• Inversions de priorités à cause des ordonnanceurs à priorités fixes ou du partages de connexions TCP
•• allocation dynamique globale de mémoire
•• grand nombre d'accès mémoire, de défauts de cache, d'allocation et désallocation dans le tas (allocation dynamique de mémoire) et de commutations de contexte.
•• pas de priorité pour l'établissement des connexions ni dans les messages, service à l'ancienneté ("FIFO")
•• coût du glanage (ramasse-miettes, "garbage collector) dû l'allocation dynamique de mémoire
••••••• néanmoins, il y a des essais de définitions de normes pour ADA, JAVA, CORBA

2.5. LES LANGAGES DE PROGRAMMATION POUR LE TEMPS RÉEL

HYPOTHÈSES SUR LA CONCORDANCE DES TEMPS DU PROCÉDÉ ET DU SYSTÈME DE CONTRÔLE



MODÈLES ET LANGAGES RÉACTIFS SYNCHRONES

langages impératifs et textuels : CSML (1991), ESTÉREL(1991),
langages déclaratifs à flots de données textuels ou graphiques : LUSTRE (1991), SIGNAL(1991)
outils de modélisation : GRAFCET (1982), STATECHARTS(1987)

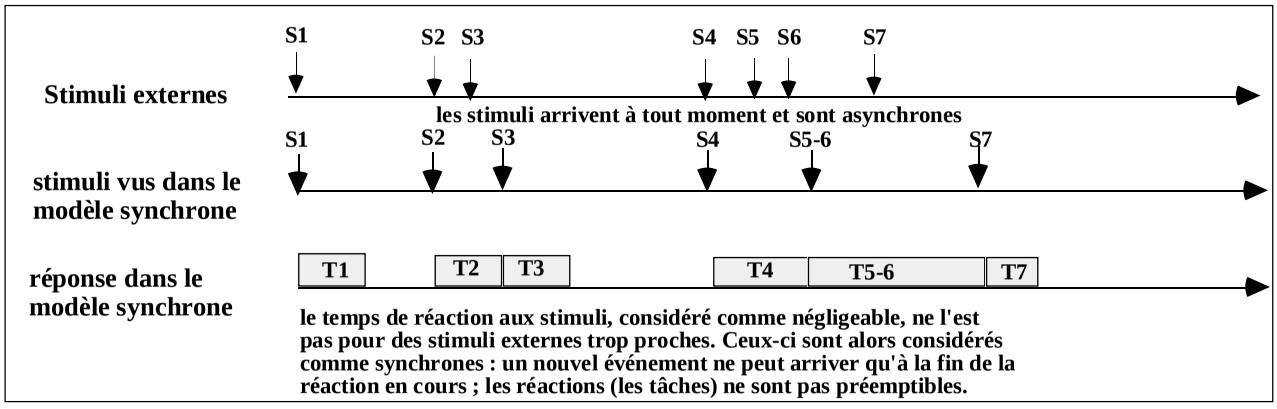
MODÈLES ET LANGAGES ASYNCHRONES

langage normalisé CCITT pour les télécommunications : CHILL
langage normalisé ISO : ADA
langage réactif impératif et textuel : ELECTRE (1985)
outil de modélisation : RÉSEAUX DE PETRI (1982)

LANGAGE SYNCHRONE OU LANGAGE ASYNCHRONE ?

Le synchrone est plus simple à programmer quand les ressources sont allouées seulement pour la durée d'une réaction (pas de préemption, pas d'attente de ressource occupée). On peut faire des preuves plus facilement.
Le compilateur génère un automate à états finis. On peut mesurer la transition la plus longue.

Mais si des stimuli nouveaux arrivent pendant la réponse du stimulus précédent, on n'est plus dans l'hypothèse de temps nul et on peut ne plus suivre d'assez près la dynamique du procédé (instabilités dues au retards).



Le choix dépend beaucoup du contexte (durée de calcul de la réponse, durée des messages). Le modèle synchrone n'est pas toujours applicable. La réalité est asynchrone.

PRINCIPAUX ASPECTS TEMPS RÉEL DU LANGAGE ADA

MANUEL DE RÉFÉRENCE ET ANNEXE TEMPS RÉEL (révision et normalisation ISO de 1995)

• Expression de la concurrence et du parallélisme des processus

Composants actifs : tâche, rendez-vous, et aussi partition appelée par RPC,
Composants passifs partagés : objets protégés avec entrées gardées (barrières de synchronisation) et réponse possible par étapes ("requeue"), priorités entre les entrées et dans chaque file d'attente associées
Ordonnancement par priorités fixes, traitement de l'inversion de priorité (par priorité plafond)

• Expression fine des temps : horloges, comptes-temps
Date : type time. Possibilité de définir des horloges et des types de temps plus précis
Durée : type duration
instruction "delay durée" ou "delay until date"

• Utilisation précise de l'architecture matérielle
Placement et représentation des registres de contrôle du matériel et des interruptions.
Programmation des interruptions et de leurs priorités ("attach").
Interface avec d'autres langages évolués ou des environnements standard (Posix).

• Outils avancés de génie logiciel
Programmation objet, héritage, classes, pointeur par type accès aux objets. Mécanisme d'exceptions
Bibliothèques hiérarchiques avec règles de visibilité, partitions et systèmes distribués
Contrôle utilisateur de l'allocation dynamique et de la désallocation des objets

PRINCIPAUX ASPECTS TEMPS RÉEL DU LANGAGE ADA

PROFIL RAVENSCAR

Pour les applications critiques qui doivent être validées ou certifiées
Interdire toute allocation dynamique et files d'attente qui créent de l'indéterminisme.

• Normalisation ISO, en 2000, de restrictions sur l'expression de la concurrence.
Compilateurs les vérifiant.

• Interdiction de création et d'allocation dynamique des objets protégés et des tâches. Une seule entrée par objet protégé et pas de file d'attente. Pas de priorités statiques.

• Pas d'attachement dynamique des interruptions,...