



# Enseignement du temps réel avec Ada95

Christian.Carrez@cnam.fr

# Le contexte

- Systèmes Temps réels et leur programmation
    - DESS développement de logiciels sûrs
    - 2-3ème année ingénieur
  - 30h cours, 30h TD
    - Ordonnancement temps réel
    - Méthodologie
    - Programmation asynchrone
    - Distribution, tolérance aux pannes
- Ada95  
comme  
support*

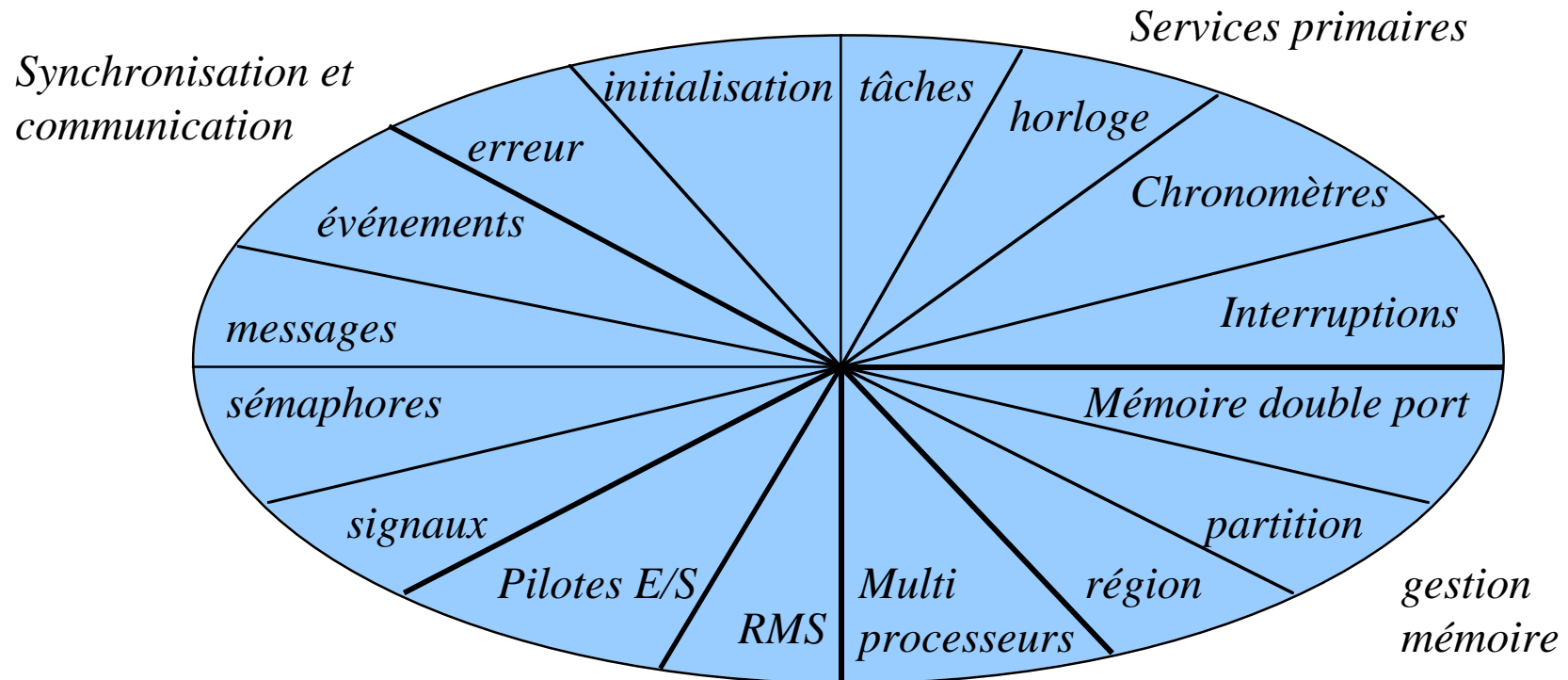
# Programming "in the large"

- masquage d'informations, compilation séparée, types abstraits, généricité, extensibilité
- contexte temps réel => garder un bon compromis expression <-> efficacité
- idée Ada: faire prendre conscience au programmeur des outils qu'il utilise en gardant une bonne efficacité

# Bibliothèque de composants

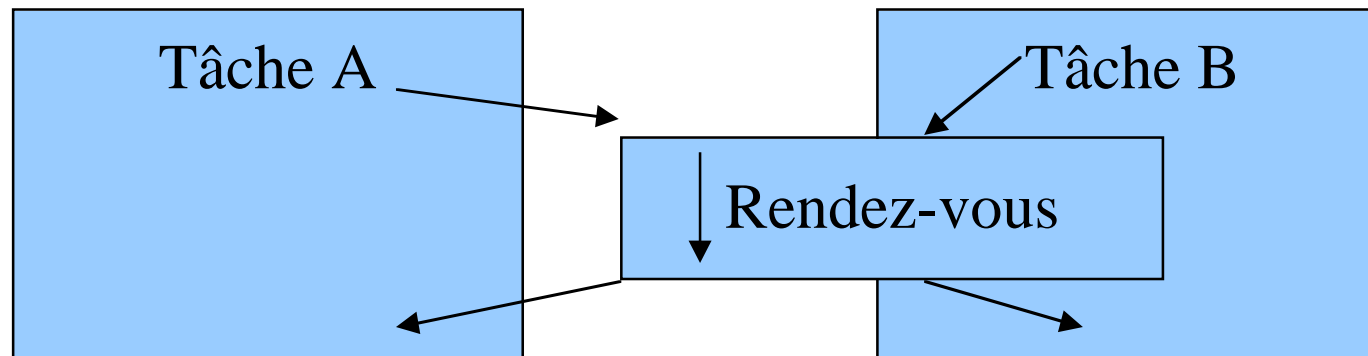
- règles de construction,
- portabilité,
- dépendances à l'implantation,
- gestion des composants

# Les fonctions d'un exécutif temps réel (RTEMS)

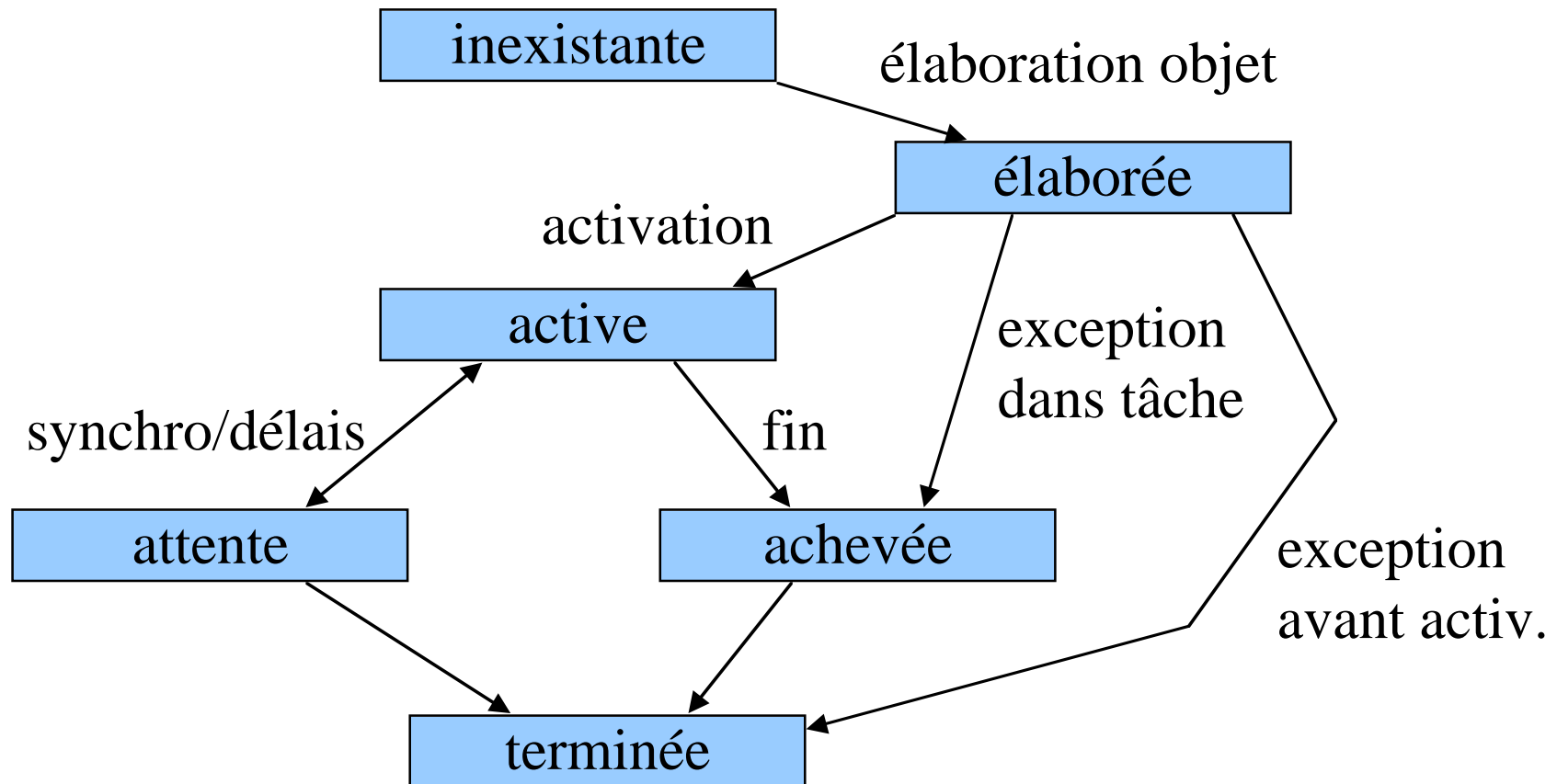


# concurrency, objets actifs, synchronisation

- Les tâches Ada et les rendez-vous
- Fonctionnement de type client-serveur
- Complexité de la création/termination des tâches

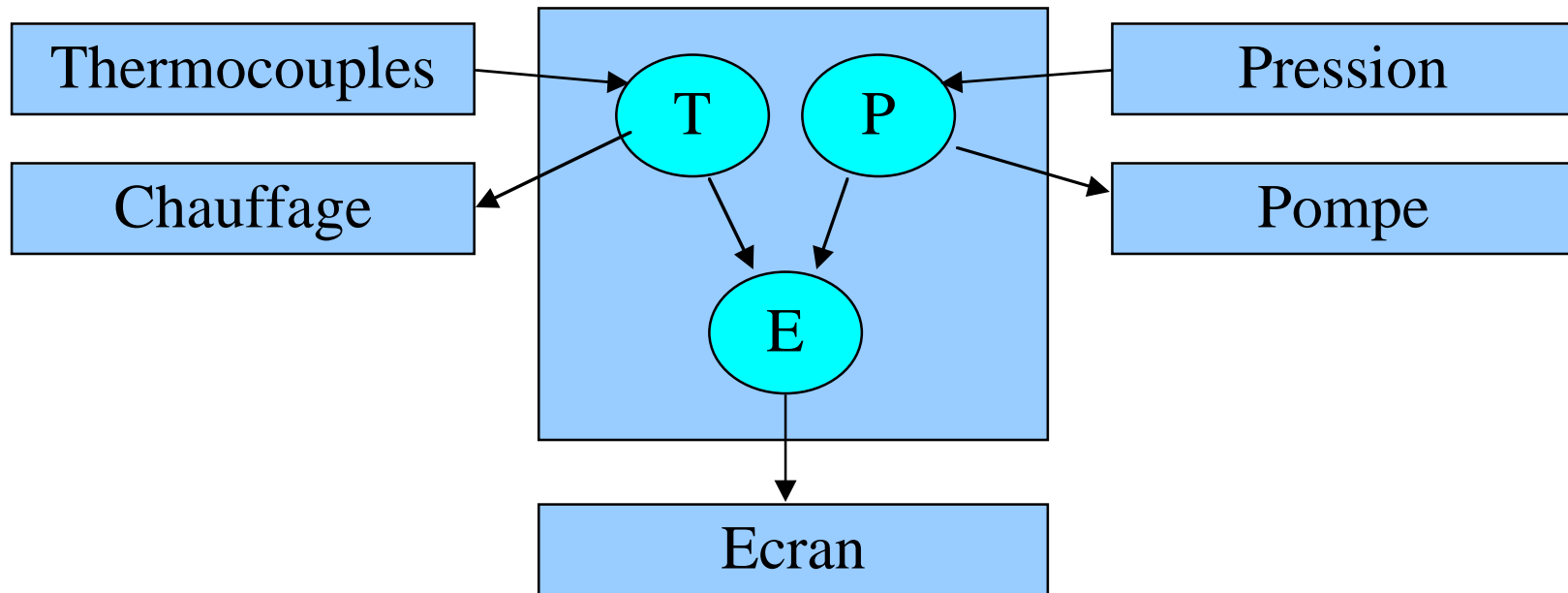


# Tâches: complexité de la création/terminaison



# exemple de contrôle de procédé,

- modules: types, E/S, modèles, tâches
- limites client serveur (non respect des échéances)



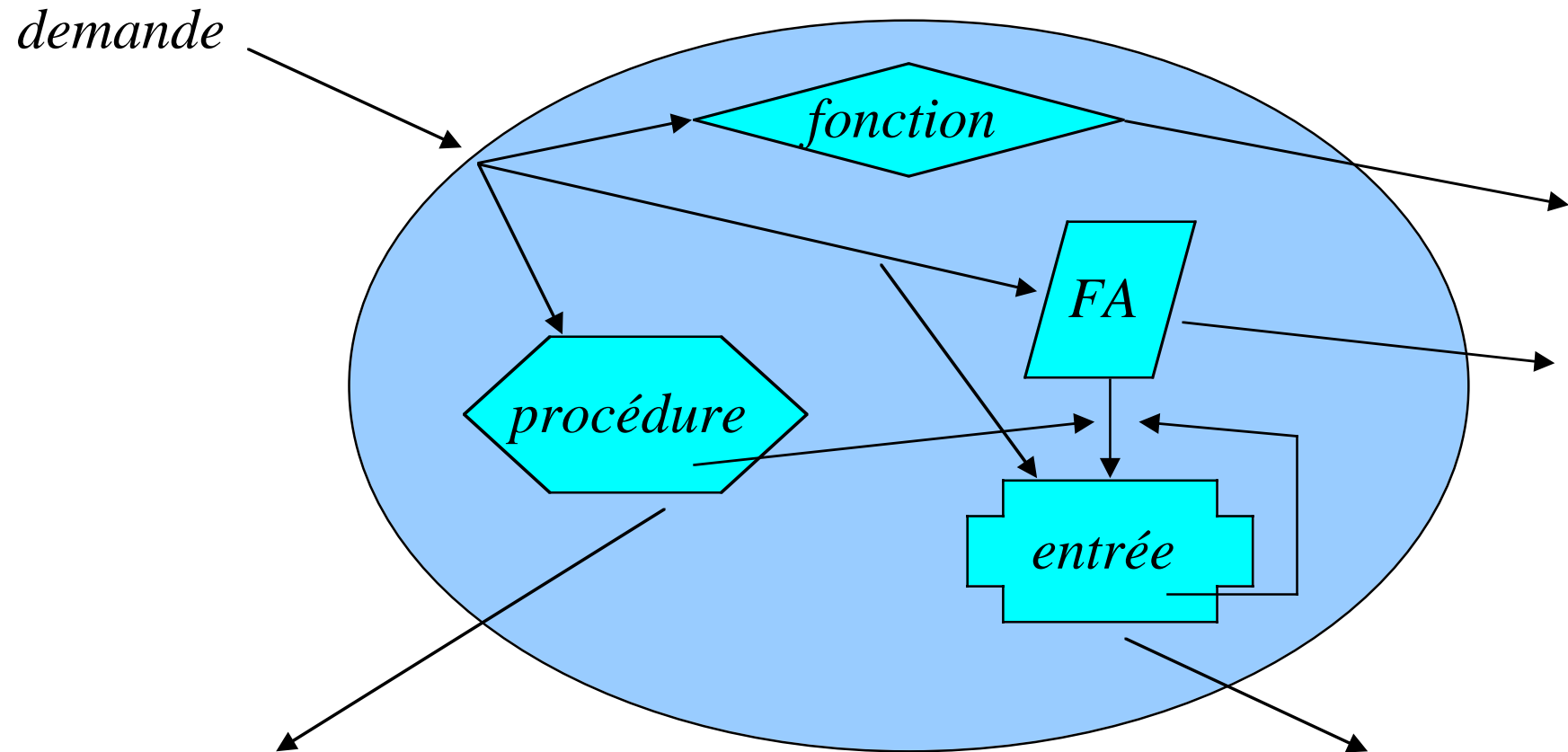
# Partage de données, objets passifs

- rappels de mécanismes
  - sémaphores, événements, moniteurs...
- objets protégés = mécanisme de haut niveau
- principe de fonctionnement = sémantique claire

# Objets protégés

- exemple d'utilisation en contrôle de procédé
  - reprise de l'exemple précédent
- règles de fonctionnement
  - schéma lecteur-rédacteur
  - pas d'opération potentiellement bloquante
  - réévaluation des barrières avant libération
  - durée d'exécution devrait être brève

# Principe de fonctionnement



# Contrôle de ressources

- encapsulation dans des modules => serveurs
- prise en compte des paramètres par "requeue"
- prise en compte de la priorité du client
- interblocage et famine

# Prise en compte des contraintes temps réel (1)

- modèle interruption
- priorité des tâches, héritage de priorité
- distinction entre mécanisme et politique
- mécanisme: allocation processeur à la tâche en tête de la file la plus prioritaire
- politique: gestion des files, priorité plafond

# Prise en compte des contraintes temps réel (2)

- reprise de l'ordonnancement
  - rate monotonic, inverse deadline
  - earliest deadline first
- changement dynamique de priorité  
(=>quand)
- prise en compte des tâches apériodiques

# Prise en compte des contraintes temps réel (3)

- contrôle synchrone (sémaphore privé)
- contrôle de tâche asynchrone
  - Asynchronous\_Task\_Control
- transfert asynchrone
  - select ... then abort ...
- besoin d'une horloge monotone
- besoin de mesure de temps CPU consommé

# Restrictions du langage

- justifier l'usage des restrictions
- donner des exemples
- profil Ravenscar

# Distribution

- Partitionnement (pré, post)
- Nœud virtuel
- Communications entre les nœuds
- Ordonnancement et répartition
- La distribution au travers de Ada95

# Techniques d'implantations en exercice

- Contraintes relatives: gestion d'une ligne série
- Implantation de tâches périodiques et apériodiques à contraintes strictes
- Implantation d'un serveur sporadique
- Vérification du respect des échéances

# Pourquoi Ada95

- Utilisé dans plusieurs enseignements du Cnam
- Bon langage support
- Permet de s'appuyer sur le "rationale"
- Dans chaque cas, l'exposé de la sémantique Ada95 permet de montrer les problèmes