La théorie de la supervision pour le développement des systèmes embarqués dans l'automobile

Jean-Pierre Elloy
Jean-Pierre.Elloy@irccyn.ec-nantes.fr
IRCCyN (UMR CNRS 6597) - Ecole Centrale Nantes

Sommaire

Sommaire

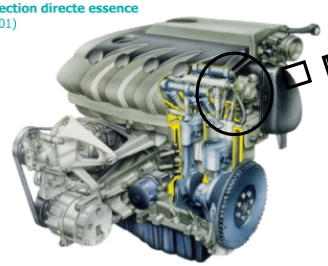
Le contexte

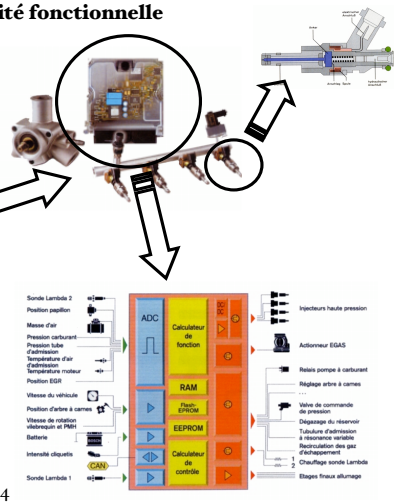
☆ Le contexte technologique

La complexité fonctionnelle

- *Lois de commande **multivariables***
- *Nombre de **grandeurs d'influence***
- *Nombre de **modes de fonctionnement***

Renault Moteur
Injection directe essence
(2001)



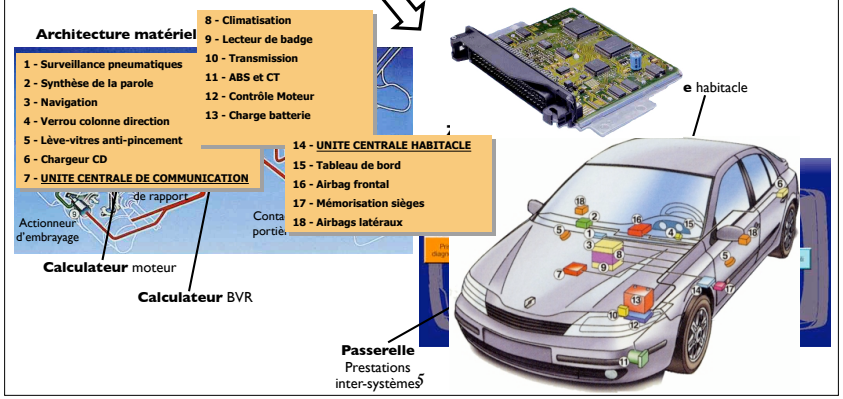


Le contexte

Le contexte technologique

La complexité architecturale

- *Nombre et hétérogénéité des* ***calculateurs***
- *Multiplicité des* ***réseaux***



Le contexte

Le contexte technologique

La complexité inter-systèmes

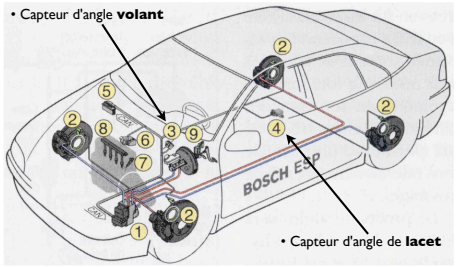
- *Coopération de fonctionnalités* ➜ ***prestations nouvelles***
- *Inclusion de fonctionnalités* ➜ ***fonctionnalités nouvelles***

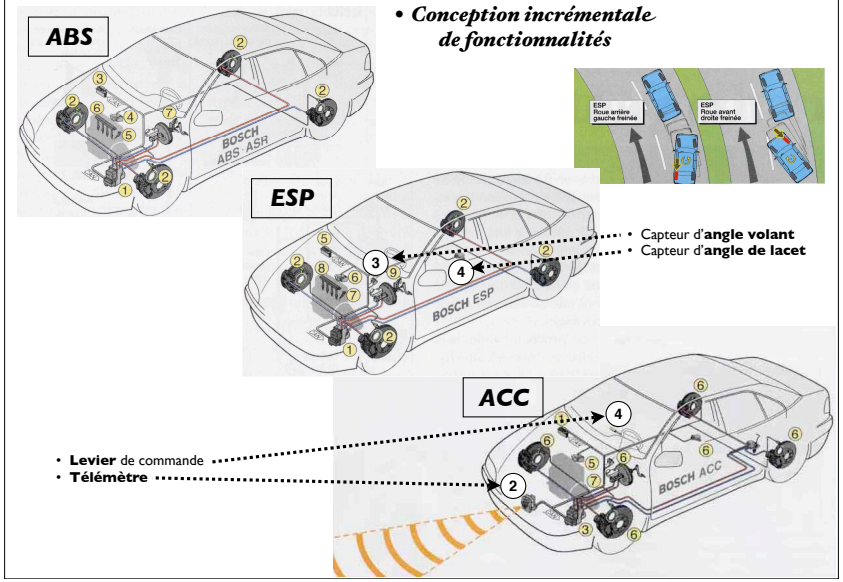
- **Eclairage** (feux de ville) *commandé par*
- **Essuyage** (continu)

ESP ⇩ ***= ABS + ...***

1 • Groupe hydraulique ABS **Calculateur** intégré
• Capteur de vitesse de rotation de roue 2
5 • **Calculateur** moteur (rampe d'injecteurs)
• Papillon motorisé 6
8 • Injecteurs
• Module d'allumage 7
9 • Capteur de position de pédale d'accélérateur

• Capteur d'**angle volant** 3
4 • Capteur d'**angle de lacet**

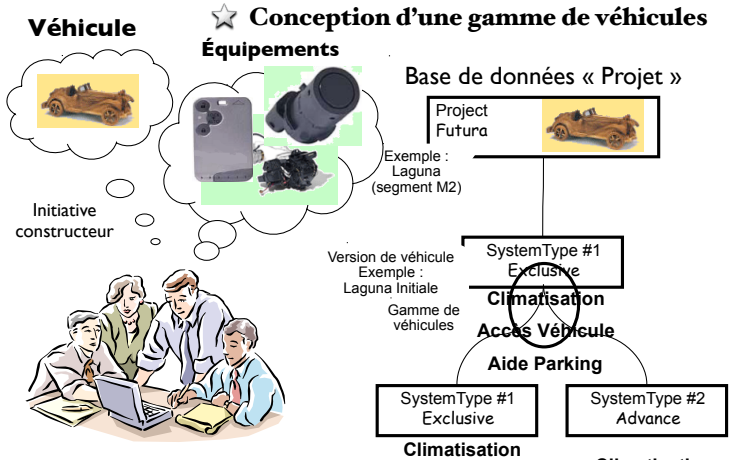




Sommaire

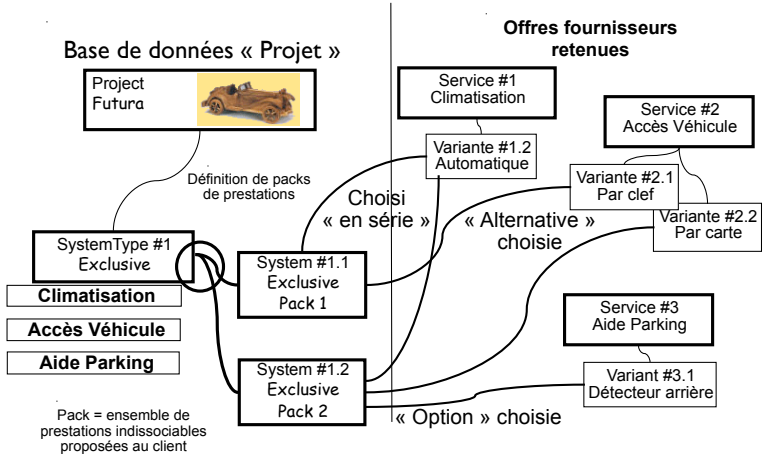
Projet

Conception d'une gamme de véhicules



Projet

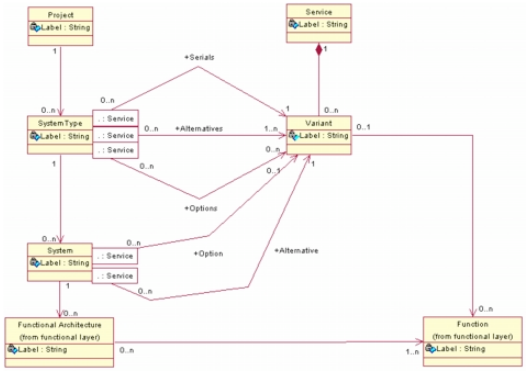
Les prestations



AIL
Vue Système

Projet

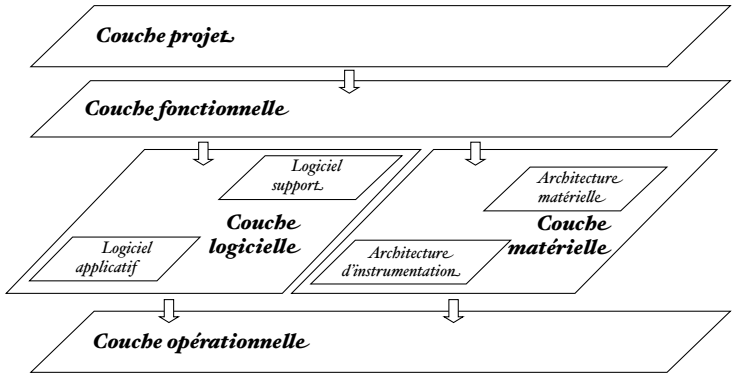
Le diagramme de classe UML de la couche Projet



Sommaire

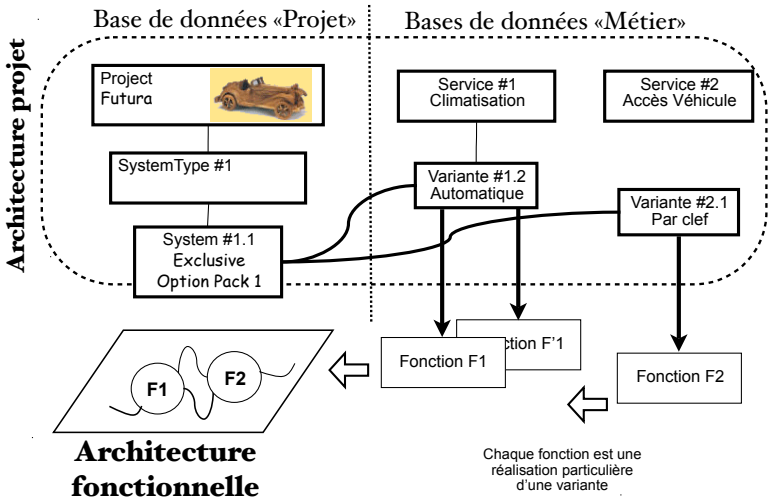
La structuration du système électronique

... se décline en couches ...



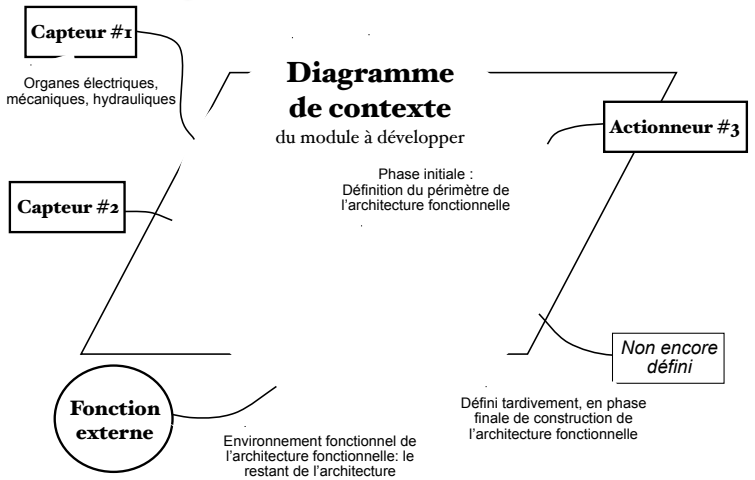
Sommaire

Fonction



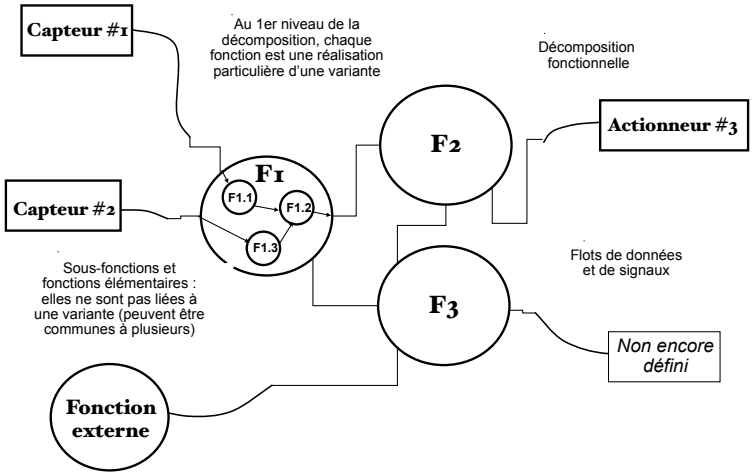
Fonction

☆ **Le diagramme de contexte**



Fonction

La décomposition hiérarchique

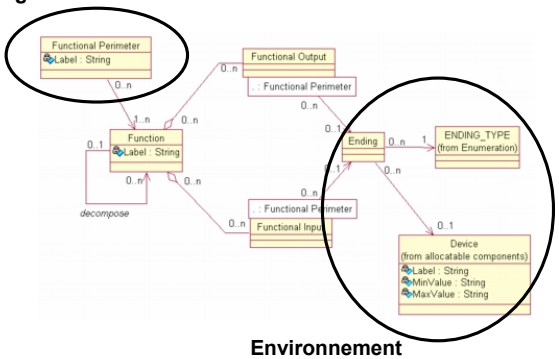


Fonction

AIL — Vue Système

Le diagramme de classe UML du diagramme de contexte

Diagramme de contexte

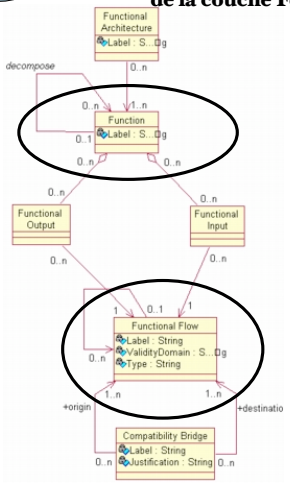


Environnement du diagramme

Fonction

AIL — Vue Système

Le diagramme de classe UML de la couche Fonctionnelle



Organisation des fonctions

Ports d'entrées-sorties

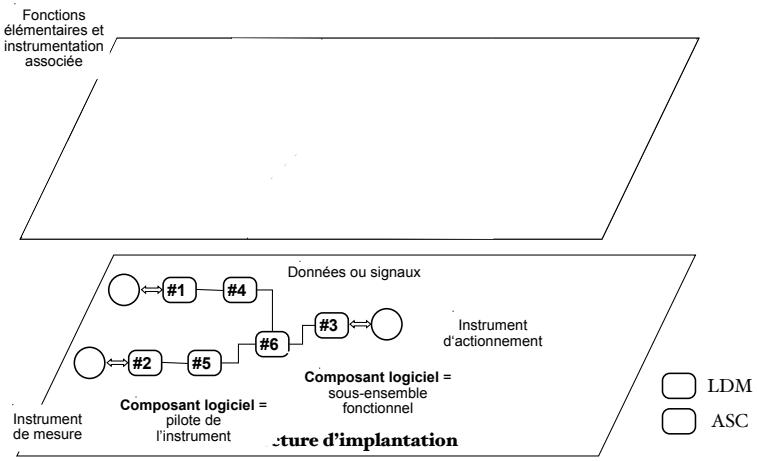
Flots de données (types des données échangées)

Compatibilité applicative (entre données de types ≠)

Sommaire

Logiciel

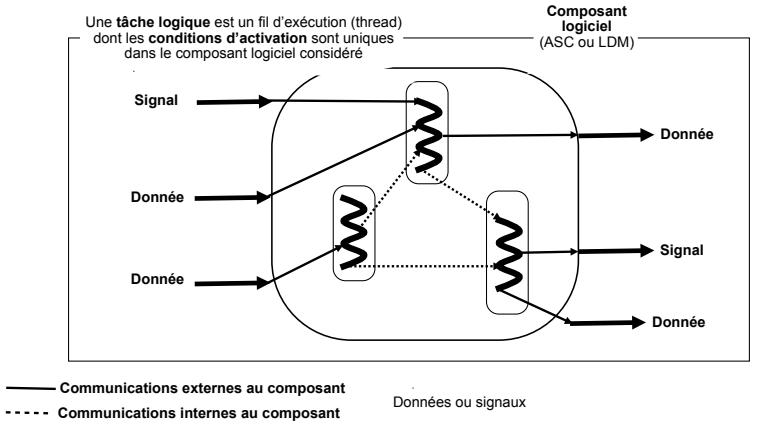
Décomposition des fonctions en composants



ture d'implantation

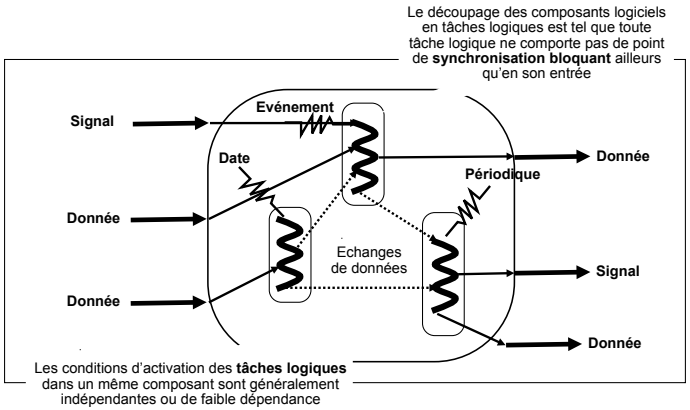
Logiciel

La décomposition en taches logiques



Logiciel

Conditions d'activation des tâches



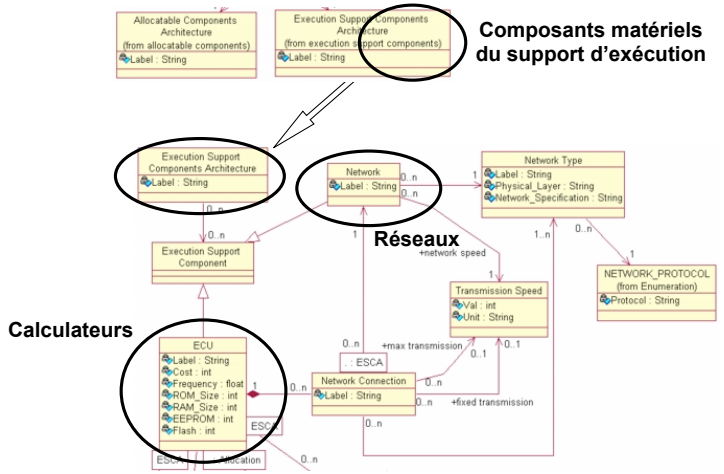
Sommaire

AIL

Vue Système

Matériel

Le diagrammes de classes du support d'exécution

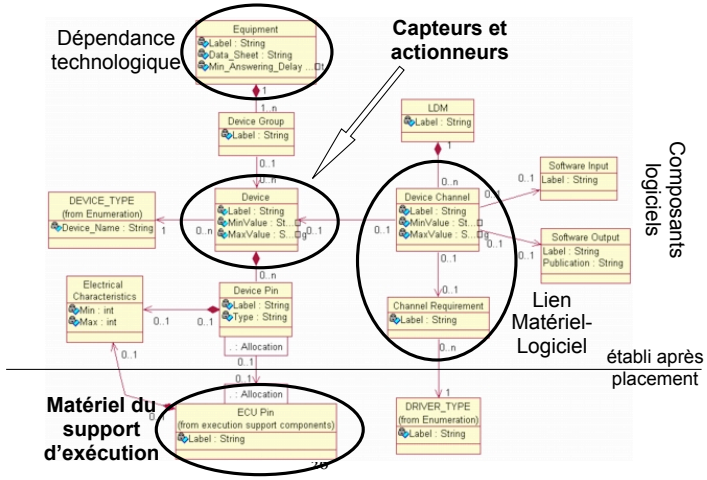


AIL

Vue Système

Matériel

Le diagrammes de classes de l'instrumentation

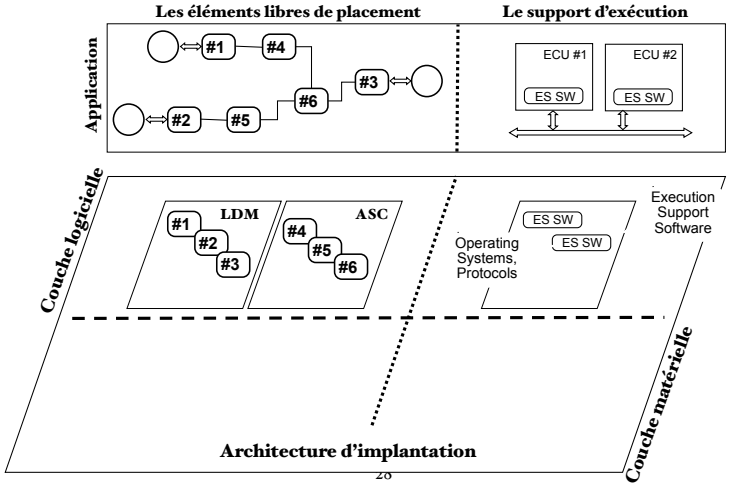


Sommaire

- Le contexte
- *La demande technique*
- Le projet véhicule
- *Les phases générales du développement*
- **La structuration de la partie électronique**
  - **Couche opérationnelle**
    - **Le placement du logiciel et du matériel**
    - **Construction des tâches exécutables**
    - **La structure opérationnelle d'un site**
    - **Le diagramme de classes opérationnel-tâches**
    - **Le diagramme de classes opérationnel-réseau**
- Démarches de développement du logiciel

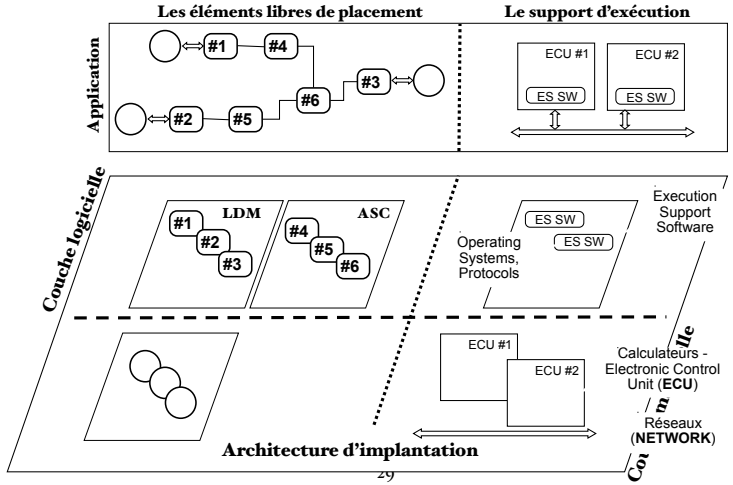
Opérationnel

Le placement du logiciel

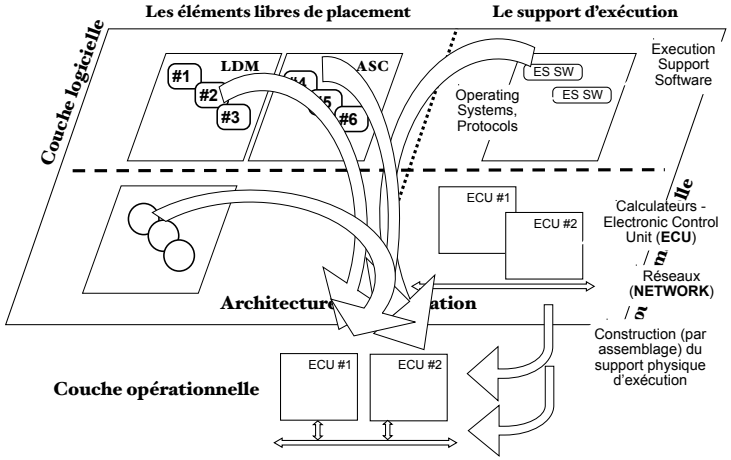


Opérationnel

Le placement du matériel

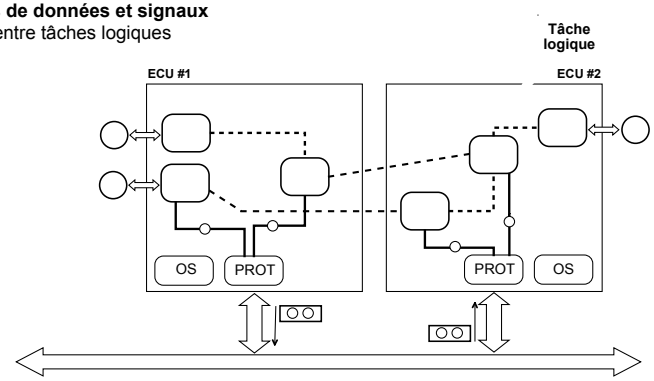


Opérationnel



Opérationnel

Flots de données et signaux
entre tâches logiques



Opérationnel

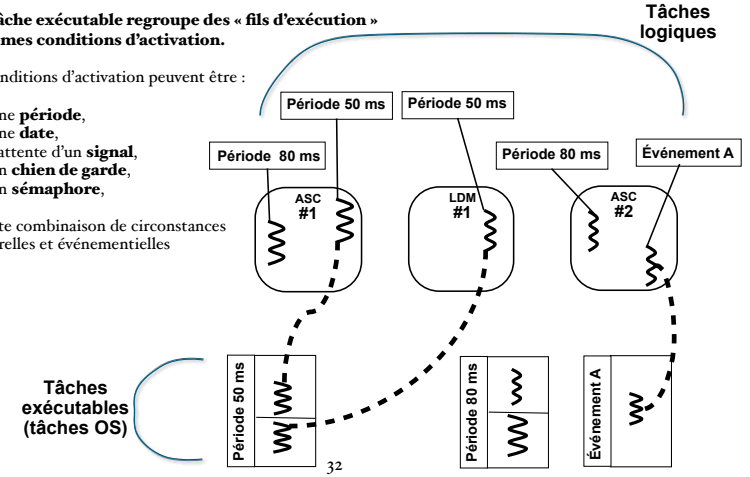
Construction des tâches exécutables (OS)

Une tâche exécutable regroupe des « fils d'exécution » de mêmes conditions d'activation.

Ces conditions d'activation peuvent être :

- une **période**,
- une **date**,
- l'attente d'un **signal**,
- un **chien de garde**,
- un **sémaphore**,

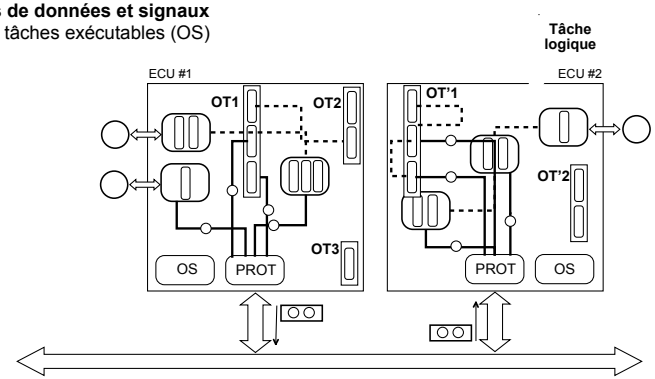
ou toute combinaison de circonstances temporelles et événementielles



Opérationnel

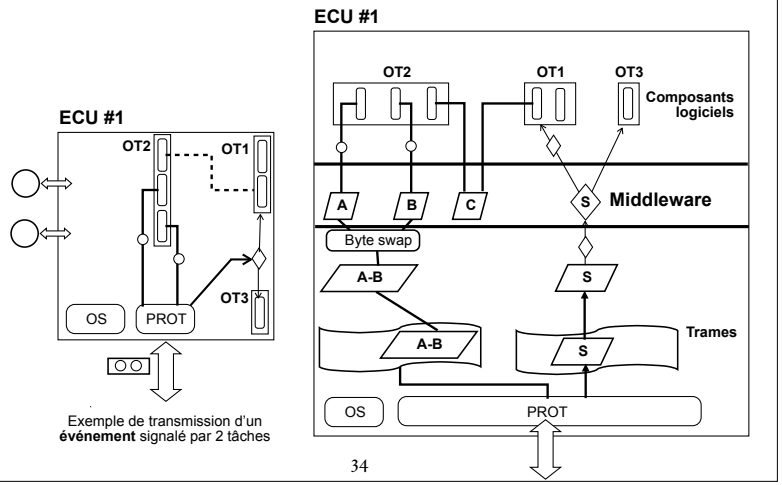
- **La construction des échanges entre taches OS**

Flots de données et signaux
entre tâches exécutables (OS)



Opérationnel

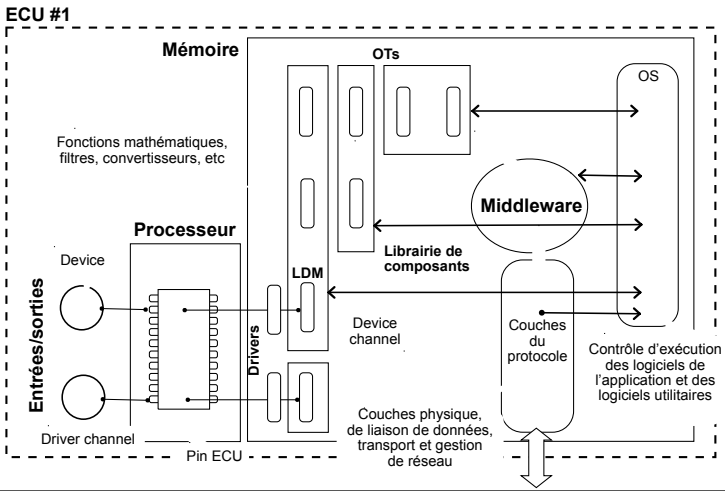
La structure opérationnelle d'un site



Exemple de transmission d'un **événement** signalé par 2 tâches

Opérationnel

La structure technique d'un site



Sommaire

- Le contexte
- *La demande technique*
- Le projet véhicule
- *Les phases générales du développement*
- La structuration de la partie électronique
- **Démarches de développement du logiciel**
  - **La modélisation dynamique et la vérification**
  - **Modélisations fonctionnelle et modale**
  - **La théorie de la supervision**
  - **Superviser dans les applications embarquées**

Sommaire

- Le contexte
- *La demande technique*
- Le projet véhicule
- *Les phases générales du développement*
- La structuration de la partie électronique
- **Démarches de développement du logiciel**
  - ☆ **La modélisation dynamique et la vérification**
  - ☆ Modélisations fonctionnelle et modale
  - ☆ La théorie de la supervision
  - ☆ Superviser dans les applications embarquées

Modèles

☆ Modéliser les aspects dynamiques

Modéliser pour analyser

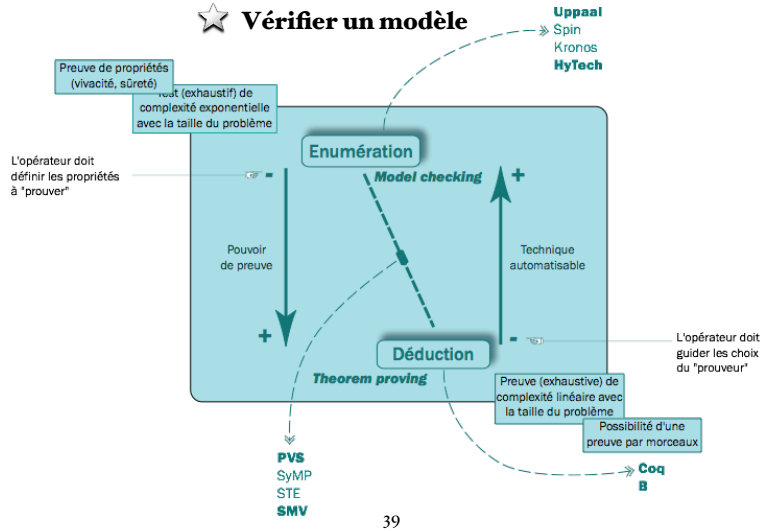
- Scénarios de tests, modèles non-formels ← **Analyse des exigences**
- Modèles analytiques pour l'ordonnançabilité, Automates hybrides, Réseau de Petri temporels ← **Analyse temporelle**
- Arbres de fautes, graphes bayésiens ← **Analyse de la sûreté de fonctionnement**
- **Systèmes états - transitions** ← **Analyse comportementale**

Représentation du parallélisme et de la gestion des ressources

- **Réseaux de Petri** (généralisés, colorés, temporels, temporisés)
- **Automates ou Systèmes de transitions** — **AEF ou FSM**

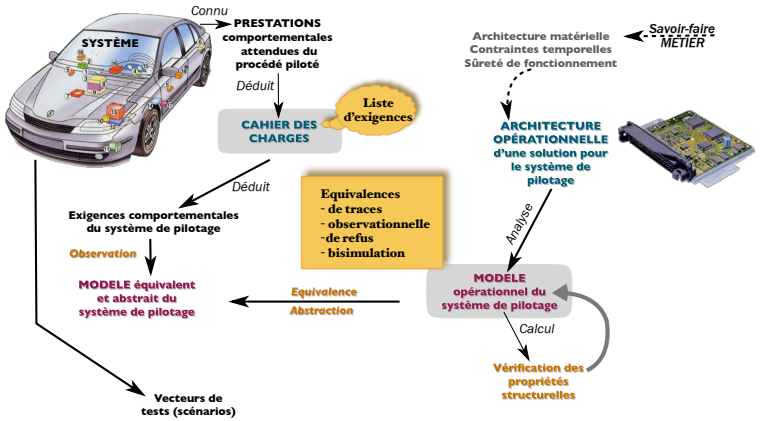
Modèles

☆ Vérifier un modèle



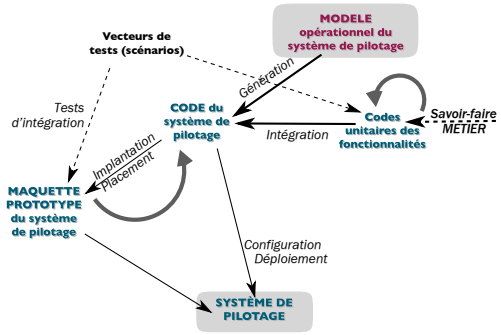
Modèles

☆ La démarche de développement (1)



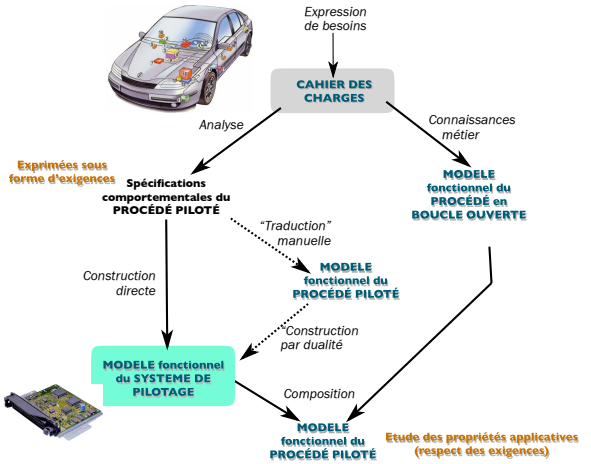
Modèles

La démarche de développement (2)



Modèles

La démarche de développement (3)



Sommaire

- Le contexte
- *La demande technique*
- Le projet véhicule
- *Les phases générales du développement*
- La structuration de la partie électronique
- **Démarches de développement du logiciel**
  - La modélisation dynamique et la vérification
  - **Modélisations fonctionnelle et modale**
  - La théorie de la supervision
  - Superviser dans les applications embarquées

Modes

Généralités sur les modes de fonctionnement

Deux grandes catégories de modes de fonctionnement

Les différentes types de modes

Tolérance aux fautes → **Modes dégradés**

Fonctionnalités à assurer → **Modes d'utilisation**

Modes dégradés

* **Services réduits, mais acceptés (spécifiés)**
→ changement de fonctionnalité

* **…intenus, mais moins sécurisés**
…la fonctionnalité avec modification
…e réalisation

Au dernier stade, le mode « refuge »

Modes d'utilisation

* **Services spécifiés différents en fonction du contexte**
→ changement de fonctionnalité avec …
schéma de réalisation (interactions, …

Considérés comme des modes « nominaux »

Modes transitoires

Basculement avec maintien de l'intégrité

Modes de « recouvrement » (confirmation du défaut avec suspension de la fonction)

Modes

☆ Généralités sur les modes de fonctionnement

Les propriétés des modes

***Principe fondamental** adopté pour la modélisation des modes de fonctionnement.*

Les modes de fonctionnement d'une même entité fonctionnelle **s'excluent mutuellement** (ce qui signifie qu'une même entité ne doit pas être dans 2 modes différents au même instant).

***L'entrée** dans un mode de fonctionnement (idem pour la **sortie** d'un mode)*

Une entité fonctionnelle est activée dans un mode :
– suite à la **détection** par elle-même des conditions de basculement
– suite à sa réception de **données modales** qui signalent l'activation d'un nouveau mode
– par le support d'exécution qui (**autoritairement**) force l'activation de l'entité
– par un **gestionnaire des modes** qui détecte et orchestre la commutation.

Auto-commutation de l'EF

Commutation externe de l'EF

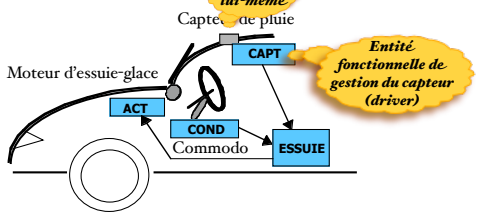
La commutation par gestionnaires peut être « autoritaire » ou coopérative avec les EF

La construction par gestionnaires est : – incrémentale, – réutilisable, – ouverte inter-systèmes

Modes

☆ Exemple : la fonction essuyage

Le capteur lui-même



Entités fonctionnelles (EF) de l'application

Drivers des capteurs

Loi de commande

* **ACT** est l'EF qui **émet la commande du moteur** d'essuie-glace
* **CAPT** est l'EF qui relève la mesurande du **capteur de pluie** et la transforme en donnée
* **COND** est l'EF qui détecte la **position du commodo** et la transforme en donnée. Le commodo a 3 positions (*haute* ⇨ **Arrêt**, *médiane* ⇨ **Balayage manuel**, *basse* ⇨ **Balayage auto**)
* **ESSUIE** est l'EF qui **calcule la commande** de l'essuie-glace à partir de
  - la consigne = position du commodo transmise par COND
  - la mesure de l'humidité sur le pare-brise transmise par EF-CAPT

Modes

☆ Exemple : la fonction essuyage

Les modes de l'application

MODES	ACT	CAPT	COND	ESSUIE
1 AUTO	★	nomi…	★	auto
2 FIXE (2a, 2b)	★	nominal ou …ne	★	manuel
3 STOP (3a, 3b)		…nal ou panne	★	arrêt

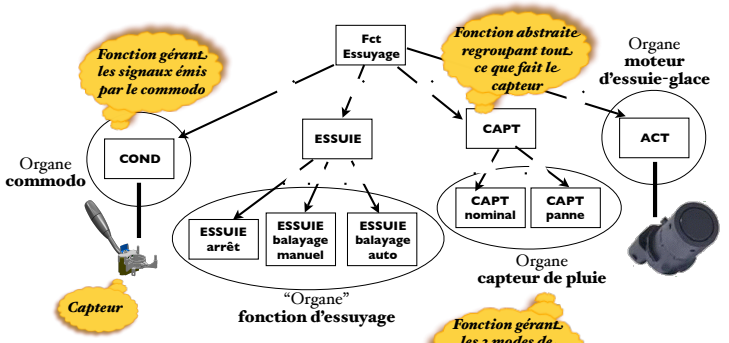
Modes de fonctionnement du capteur

« 2b » est un mode nominal, ou le mode dégradé de « 1 » si le capteur est en panne

Modes

☆ Essuyage : modélisation fonctionnelle

Décomposition fonctionnelle ORGANIQUE



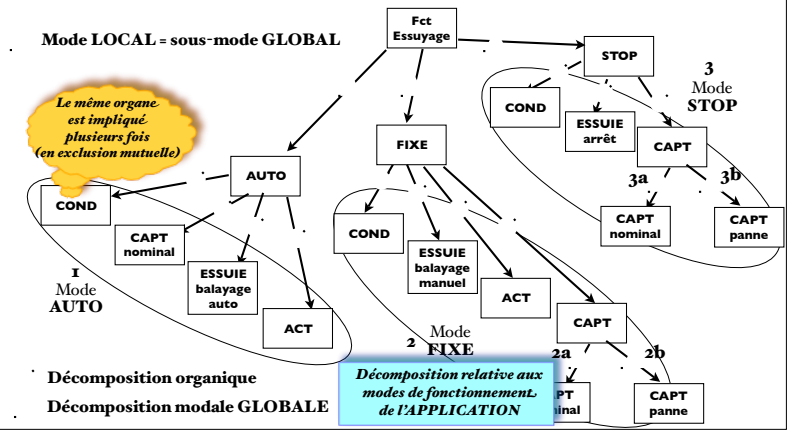
Décomposition organique

Décomposition modale LOCALE

Modes

☆ Essuyage : modélisation fonctionnelle

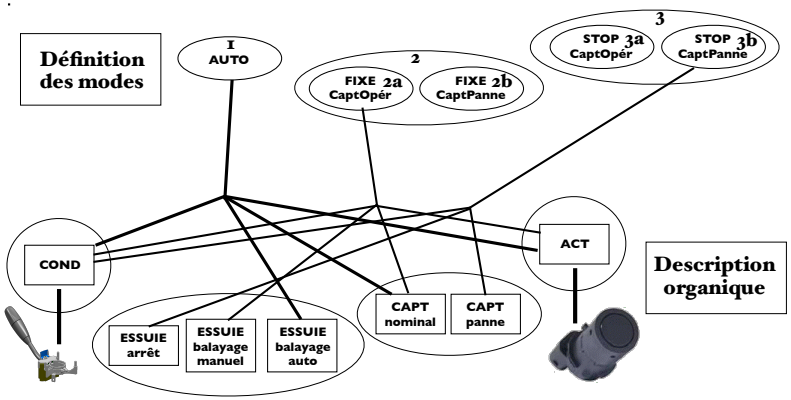
Décomposition fonctionnelle MODALE



Modes

☆ Essuyage : modélisation fonctionnelle

Description ORGANIQUE et MODALE



Modes

☆ Essuyage : modélisation fonctionnelle

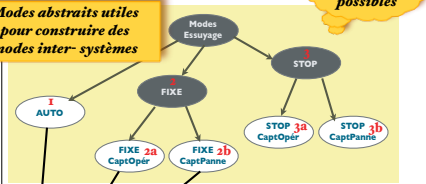
Architecture des modes et EF modaux

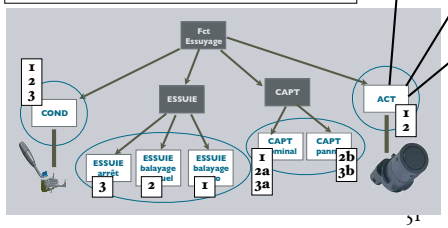
Principe de développement *d'une architecture fonctionnelle modale*

- La structure **fonctionnelle** est développée organe par organe indépendamment de leurs interactions
- Les **modes de fonctionnement** de l'application sont définis et **hiérarchisés** indépendamment des entités fonctionnelles

Modes abstraits utiles pour construire des modes inter- systèmes

D'autres hiérarchies sont possibles





- Des **relations statiques** entre structure fonctionnelle et structure des modes fixent les diagrammes fonctionnels par mode
- La gestion des **commutations de mode** est confiée aux modes abstraits qui prennent le rôle de **superviseur**

Modes

☆ Essuyage : modélisation fonctionnelle

La gestion des modes

Le rôle du gestionnaire de mode *(système de transitions intégré au mode abstrait global de l'application)*

- Les **flots de données et de signaux** échangés entre des EF dans des modes différents peuvent être différents : le gestionnaire des modes non seulement **l'activité des EF** mais gère aussi les **ports des EF**
- La **détection** des changements de mode est effectuée par le gestionnaire de mode. Pour cela, il doit pouvoir observer l'état de l'environnement (le procédé piloté) ainsi que l'évolution des EF
- **L'action** du gestionnaire sur **l'activité** des EF peut être **autoritaire** (appel à des services de gestion de tâches) ou **coopérative** (synchronisation et pilotage par signaux)
- La gestion des modes d'une application peut être éclatée et **répartie** dans plusieurs gestionnaires hiérarchisés dans un même arbre de modes ➤ construction incrémentale et passerelle inter-systèmes
- La **commutation de modes** doit contrôler tous les états intermédiaires et forcer le passage par des modes **transitoires légaux**

Intérêt de la théorie de la supervision

Sommaire

- Le contexte
- La demande technique
- Le projet véhicule
- Les phases générales du développement
- La structuration de la partie électronique
- **Démarches de développement du logiciel**
  - ☆ La modélisation dynamique et la vérification
  - ☆ Modélisations fonctionnelle et modale
  - ☆ **La théorie de la supervision**
  - ☆ Superviser dans les applications embarquées

Supervision

☆ Le principe de la théorie

Origine

- Elle est fondée sur les langages formels (Hack, 76) (Hopcroft & Ullman, 79), générés par un automate d'états fini

Principe (Ramadge & Wonham)

- « Soit un procédé ***G*** et une spécification de comportement ***C***, la théorie de la commande supervisée produit la construction d'un superviseur ***S*** tel que le comportement de ***G*** supervisé par ***S*** satisfait la spécification ***C*** »

$$G \wedge S \Rightarrow C$$

Plus précisément

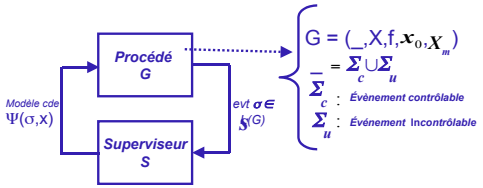
- Le procédé ***G*** et le superviseur ***S*** sont considérés comme les générateurs de deux langages : L(***G***) et L(***S***). La spécification ***C*** est aussi supposée exprimée dans un langage formel K

  Le superviseur ***S*** doit être construit pour que le système en bouclé fermée ***S*** || ***G*** génère le langage K

Supervision

☆ La supervision par les événements

Superviseur



- Le superviseur S et le procédé G sont modélisés par des systèmes de transition **évoluant en parallèle**
- G ne peut pas engendrer d'événement **sauf si celui ci appartient à S**
- Les évènements **contrôlables** de G sont **dynamiquement** autorisés ou neutralisés par S pour que le procédé G génère un langage conforme aux spécifications

Supervision

☆ Application de la méthode

Une façon de procéder

- Modéliser le comportement du procédé non supervisé (ensemble de tous les comportements possibles en boucle ouverte)
- « Surligner » dans ce modèle les comportements du procédé qui sont conformes aux spécifications
- Synthétiser le superviseur qui contraint le procédé à suivre ces comportements attendus

Extensions

- *Supervision modulaire*
- *Supervision hiérarchique*
- *Observation partielle des événements*

Limites

- *Explosion combinatoire*
- *Description des spécifications en langage formel*

Sommaire

- Le contexte
- La demande technique
- Le projet véhicule
- Les phases générales du développement
- La structuration de la partie électronique
- **Démarches de développement du logiciel**
  - La modélisation dynamique et la vérification
  - Modélisations fonctionnelle et modale
  - La théorie de la supervision
  - **Superviser dans les applications embarquées**

Superviser

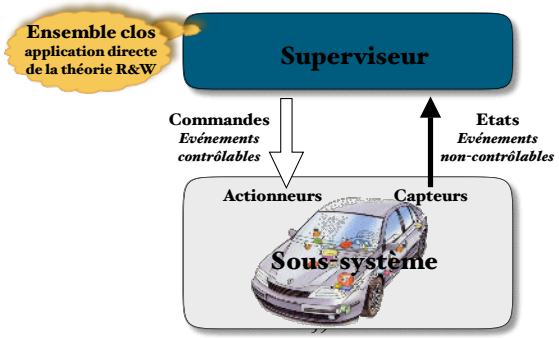
Rôles du superviseur en embarqué

	Superviseur			
	Construire une nouvelle commande			
Sous-système en boucle ouverte	Pour des sous-systèmes de taille limitée			
Procédé				

Superviser

Rôles du superviseur en embarqué

- **Construire une nouvelle commande**



Superviser

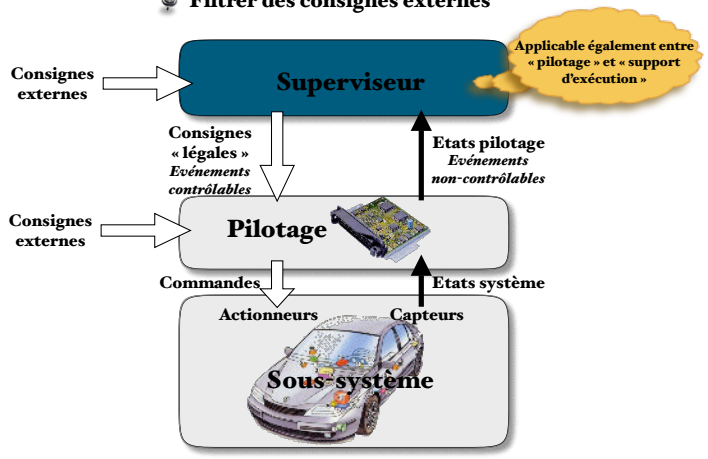
Rôles du superviseur en embarqué

	Superviseur			
	Construire une nouvelle commande	**Sécuriser une commande**		
Sous-système en boucle ouverte	Pour des sous-systèmes de taille limitée	Pour filtrer les commandes brutes		
Sous-système en boucle fermée		Pour filtrer des consignes externes		
Procédé				

Superviser

Rôles du superviseur en embarqué

Filtrer des consignes externes



Superviser

Rôles du superviseur en embarqué

Superviseur

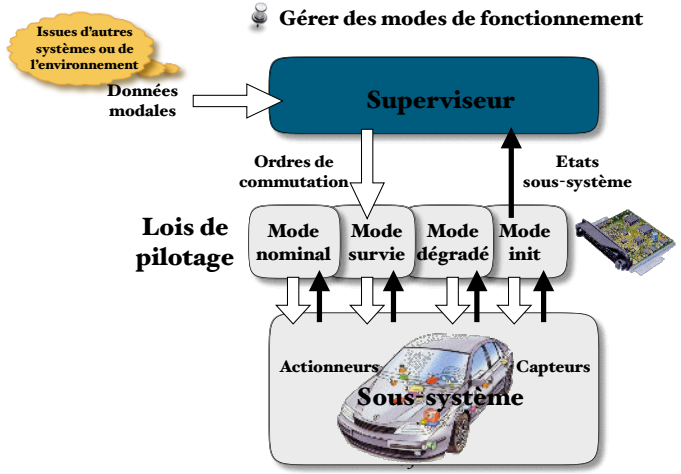
	Construire une nouvelle commande	Sécuriser une commande	Gérer les modes de fonctionnement	
Sous-système en boucle ouverte	Pour des sous-systèmes de taille limitée	Pour filtrer les commandes brutes		
Sous-système en boucle fermée		Pour filtrer des consignes externes	Pour le cas où la commande est décomposée en modes conditionnés par le système et l'environnement	

Procédé

Superviser

Rôles du superviseur en embarqué

Gérer des modes de fonctionnement



Superviser

Rôles du superviseur en embarqué

Superviseur

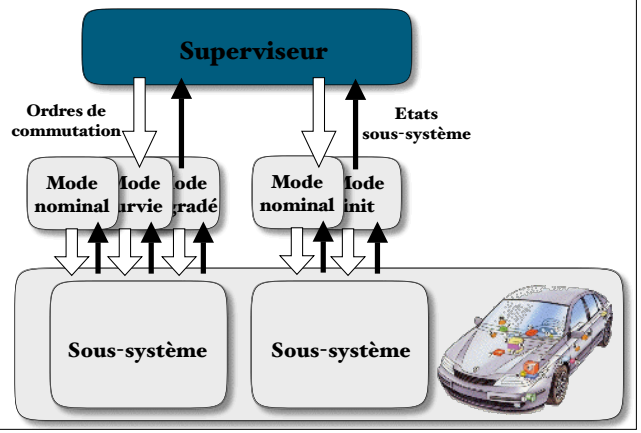
	Construire une nouvelle commande	Sécuriser une commande	Gérer les modes de fonctionnement	Coordination inter-système
Sous-système en boucle ouverte	Pour des sous-systèmes de taille limitée	Pour filtrer les commandes brutes		
Sous-système en boucle fermée		Pour filtrer des consignes externes	Pour le cas où la commande est décomposée en modes conditionnés par le système et l'environnement	
Sous-systèmes supervisés				Prestation nouvelle

Procédé

Superviser

Rôles du superviseur en embarqué

Prestation nouvelle



Superviser

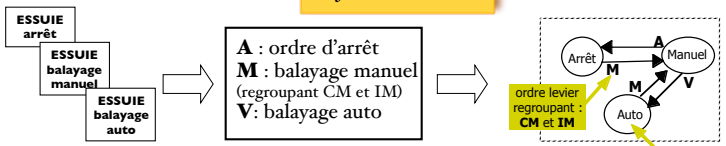
Modéliser la commutation des modes locaux

Organe par organe

"Organe"
Fonction d'essuyage

Modélisation comportementale sous forme d'automate d'états fini

... ou sous toute autre forme de système de transition



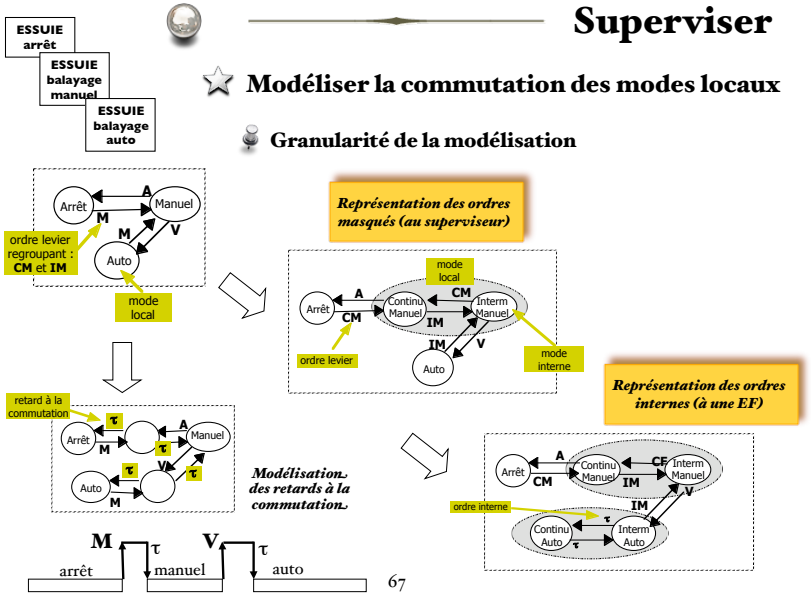
Définition des événements qui provoquent la commutation de mode local. Ces evts sont propres à chaque "organe" et sont indépendants des modes glob...

La réalisation effective de ces événements est décidée lors du codage (variable partagée, commande gardée, déclenchement sur signaux, primitives OSEK,...)

Superviser

Modéliser la commutation des modes locaux

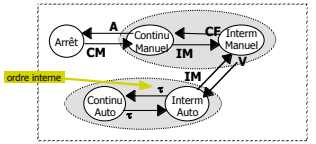
Granularité de la modélisation



Superviser

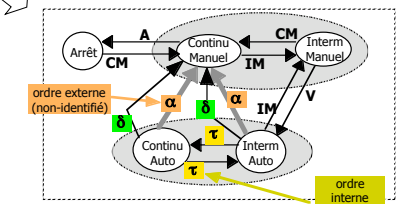
Modéliser la commutation des modes locaux

Introduire les ordres et états pour la supervision



Le superviseur peut utiliser les ordres (**A, CM, IM, V**) en lieu et place de leur origine **naturelle** (le commodo) : il intercepte donc ces ordres naturels pour les contrôler

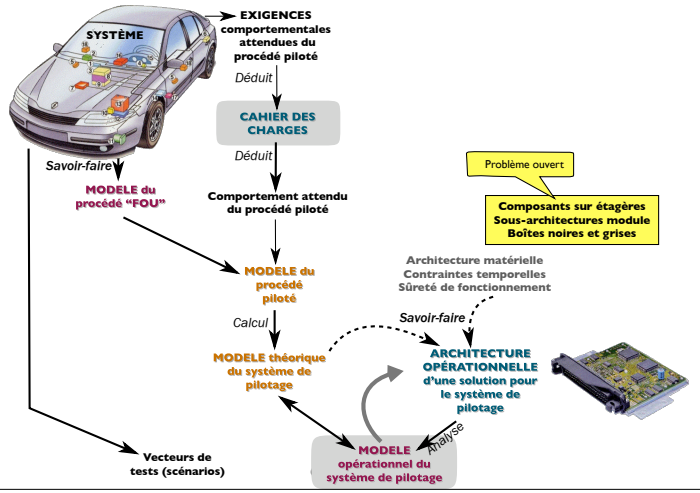
L'origine des ordres n'est pas identifié à ce stade



Le superviseur peut utiliser les ordres rajoutés **(α, δ)** pour contrôler le comportement de l'application pour obtenir les modes globaux spécifiés.
Ici **α et δ** symbolisent des ordres qui permettent de basculer dans le **mode FIXE** en cas de panne capteur ou en cas de détection d'erreur logicielle survenues **en mode AUTO**.

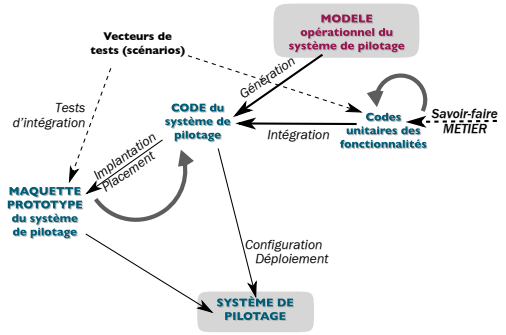
Superviser

La démarche de développement (1)



Superviser

La démarche de développement (2)



Superviser

La démarche de développement (3)