Exemple de domaines d ’application

Quels points d’entrée pour le logiciel libre ?

Le candidat Linux pour l’avionique embarquée ?

Serge Goiffon (Airbus)

« Tout ce qui n ’est pas donné est perdu. » - Proverbe Soufi

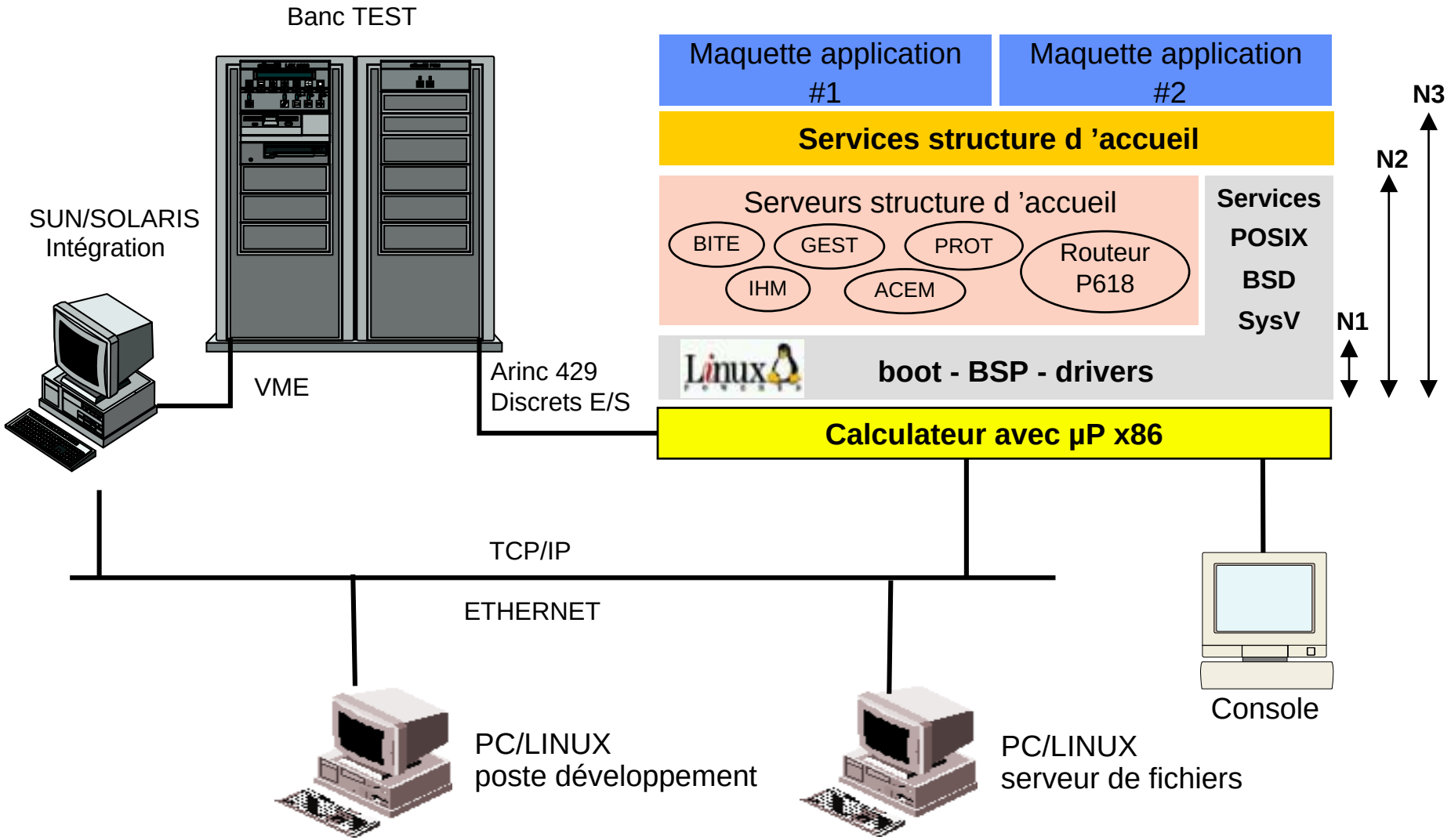
Contexte avionique embarquée

- **Stratégie industrielle basée sur le modèle « tout faire ou acheter »**
  - **Le « tout faire » n'est plus viable**
    - Complexité croissante du logiciel embarqué
    - Composants génériques (O/S, piles de communication, …) : expertise nécessaire
  - **L'incorporation de logiciels COTS n'est pas toujours possible**
    - Objectifs de certification pas toujours partagés par le fournisseur
    - Dépendance vis-à-vis du fournisseur

- **Le logiciel libre comme solution alternative**
  - Accès en modification au code source sans redevance
  - Communauté d'utilisateurs, de mainteneurs, de développeurs
  - Réutilisation de logiciel, accès à faible coût à la technologie
  - Modèle de développement coopératif, innovation

- **Questions posées : Linux peut-il …**
  - Être embarqué dans du matériel avionique
  - Être adapté pour répondre aux propriétés avioniques
  - Entrer dans un processus de certification DO178B

Embarquer LINUX : un calculateur embarqué sous LINUX



Embarquer LINUX : comment ?

- **Base noyau Linux 2.2.12-20 sans patch temps réel**
- **Adaptations majeures**
  - Code de démarrage et BSP réécrit (pas de BIOS, boot sur un SGF FLASH)
  - Fonctionnalité XIP ajoutée (exécution code depuis la FLASH)
  - Universalité des drivers Unix

    => facile de migrer de LynxOS à Linux (A429, E/S discrète, Flash, EEPROM, Ethernet, …)
  - Standard d'interface (POSIX.1, Sys V, BSD, multithreading)

    => faible impact sur le code applicatif (simple recompilation)
- **Validation**
  - Tests existants sur les drivers, OpenGroup POSIX Test Suite, lmbench
  - Commandes Unix et services réseau
  - Portage des applications maquettes

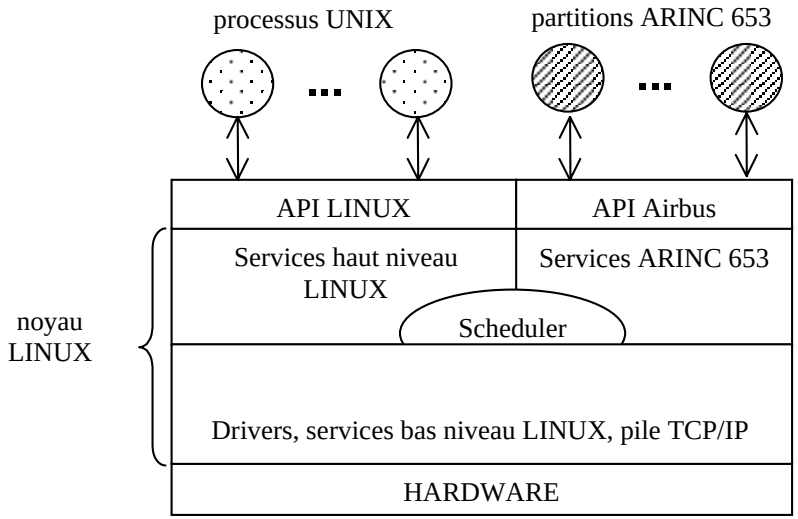
Adapter LINUX : contexte avionique nouveau A380

- **Contraintes imposées au niveau du programme avion**
  - Banalisation ressources de calculs (moins d'éléments de rechange)
  - Réseau avion basé sur AFDX (Ethernet commuté déterministe)
  - Norme avionique ARINC 653 comme standard d'interface
- **Propriétés requises**
  - Partitionnement mémoire
  - Partitionnement temporel
  - Partitionnement E/S
- **A653 => perte d'interopérabilité, nécessite développement d'outils spécifiques (réutilisation difficile)**

➔ **Utiliser Linux comme plate-forme d'accueil d'applications avioniques assurant l'interopérabilité et garantissant les contraintes de partitionnement**

Adapter LINUX : la maquette PC LINUX/IMA

- **Modification du noyau Linux 2.4.17**
  - 1 partition = 1 processus Unix + contraintes temporelles
  - Modification de l 'ordonnanceur (partitionnement temporel)
  - Ajout dans le noyau des services définis par la norme A653 => efficacité par utilisation des services internes Linux pour la synchronisation, allocation mémoire, etc...



- **Adaptation outils**
  - Analyse du comportement des applications avioniques : intégration d'événements nouveaux dans l'outil de trace Linux Trace Toolkit

Bilan des études

- **Points forts de Linux**
  - Bonne lisibilité du code du noyau, bonne expertise (support de la communauté)
  - Points communs avec LynxOS
  - Interopérabilité (standards industriels)
  - Environnement d'outils GNU complet pour le développement

- **Du point de vue de la certification …**
  - Pas de points techniques bloquants vis-à-vis des contraintes de l 'avionique embarquée
  - Nécessité d'un contrôle d'allocation des ressources système et du remplacement des mécanismes non « avionables »
  - Mise en conformité du code vis-à-vis des standards de codage avioniques, peu de document de conception du système, documentation variée, mais incomplète, éparse, et pas forcément à jour
  - Coût d 'appropriation non négligeable, nécessite un effort de « reverse engineering » pour recréer le matériel de certification