Upgrading the ContrACT Scheduler with Useful Mechanisms for Dependability of Real-Time Systems

Bertrand Brun David Andreu Robin Passama

LIRMM, CNRS et Université Montpellier 2

10 mai 2012

Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
3. Proposition d'amélioration des schémas
4. Conclusion

Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
3. Proposition d'amélioration des schémas
4. Conclusion

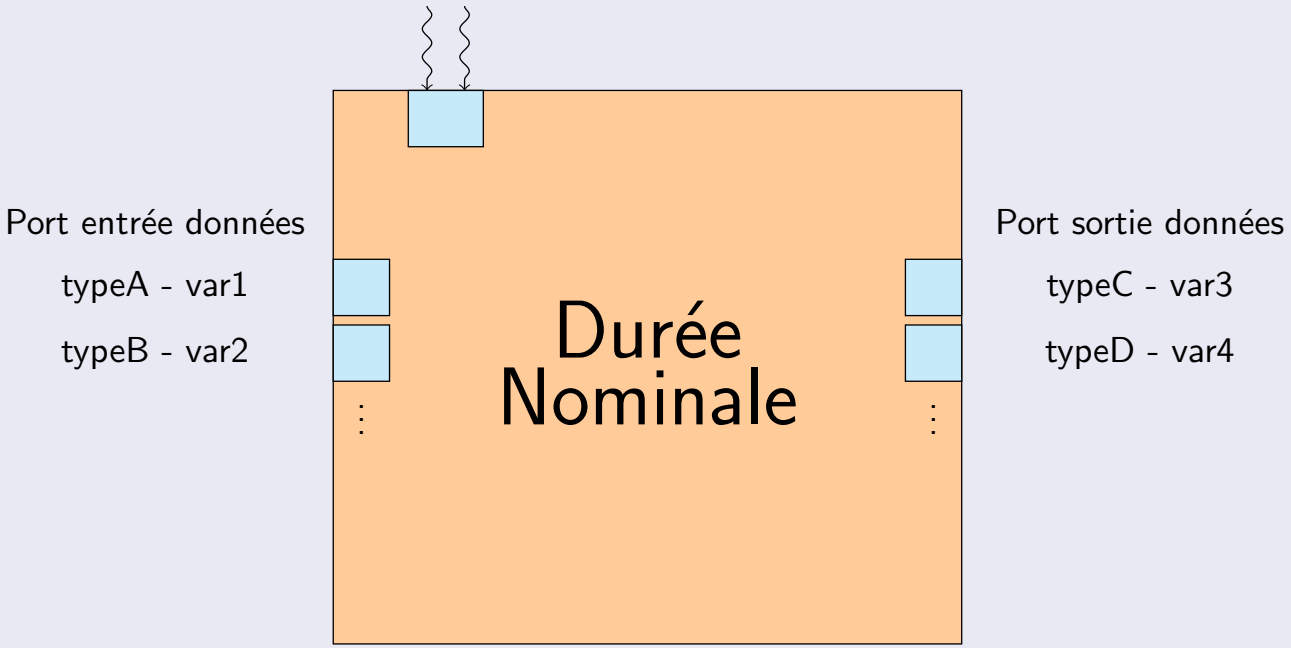
Le modèle architectural

Approche général

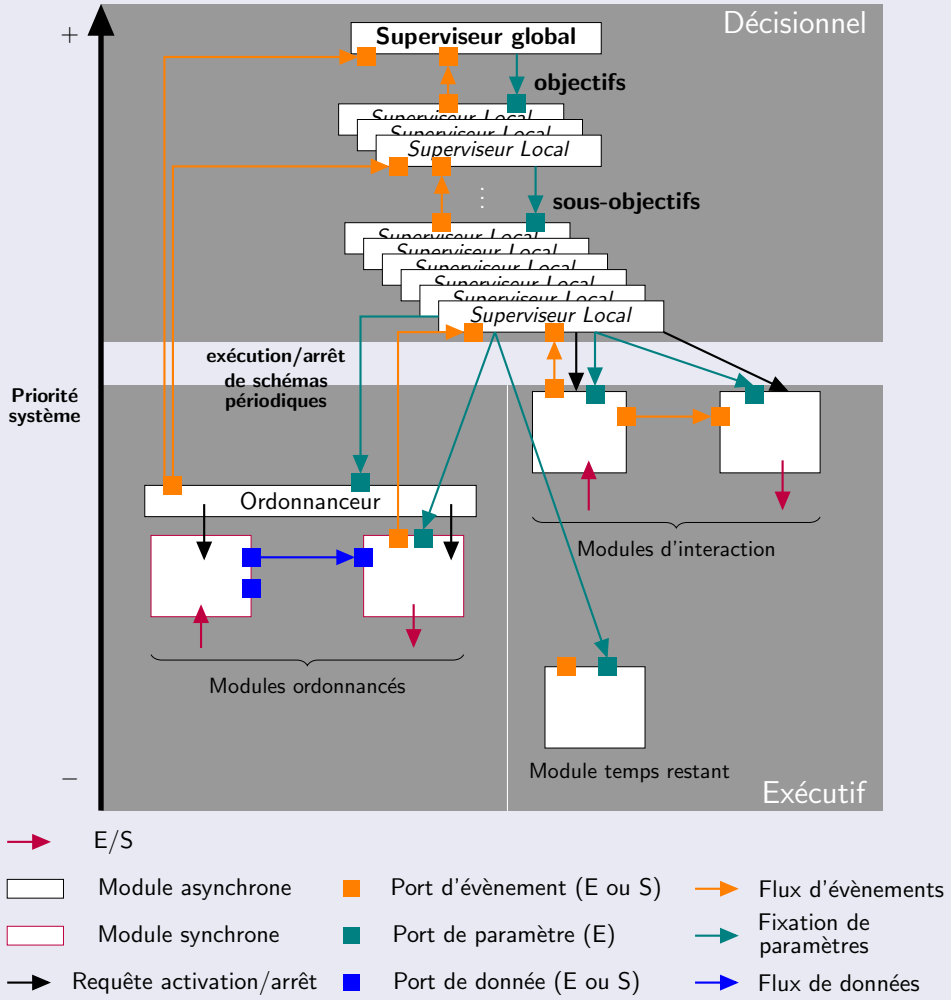
- Architecture modulaire : l'entité de base est un module correspondant à une tâche temps-réel
- Formes de composition :
  - Basée flot de données
  - Basée flot d'évènements
  - Basée requête
- 2 couches :
  - Décisionnelle (supervision « réactive »)
  - Exécutive (ordonnancement temps réel)

Module : tâche temps réel LXRT

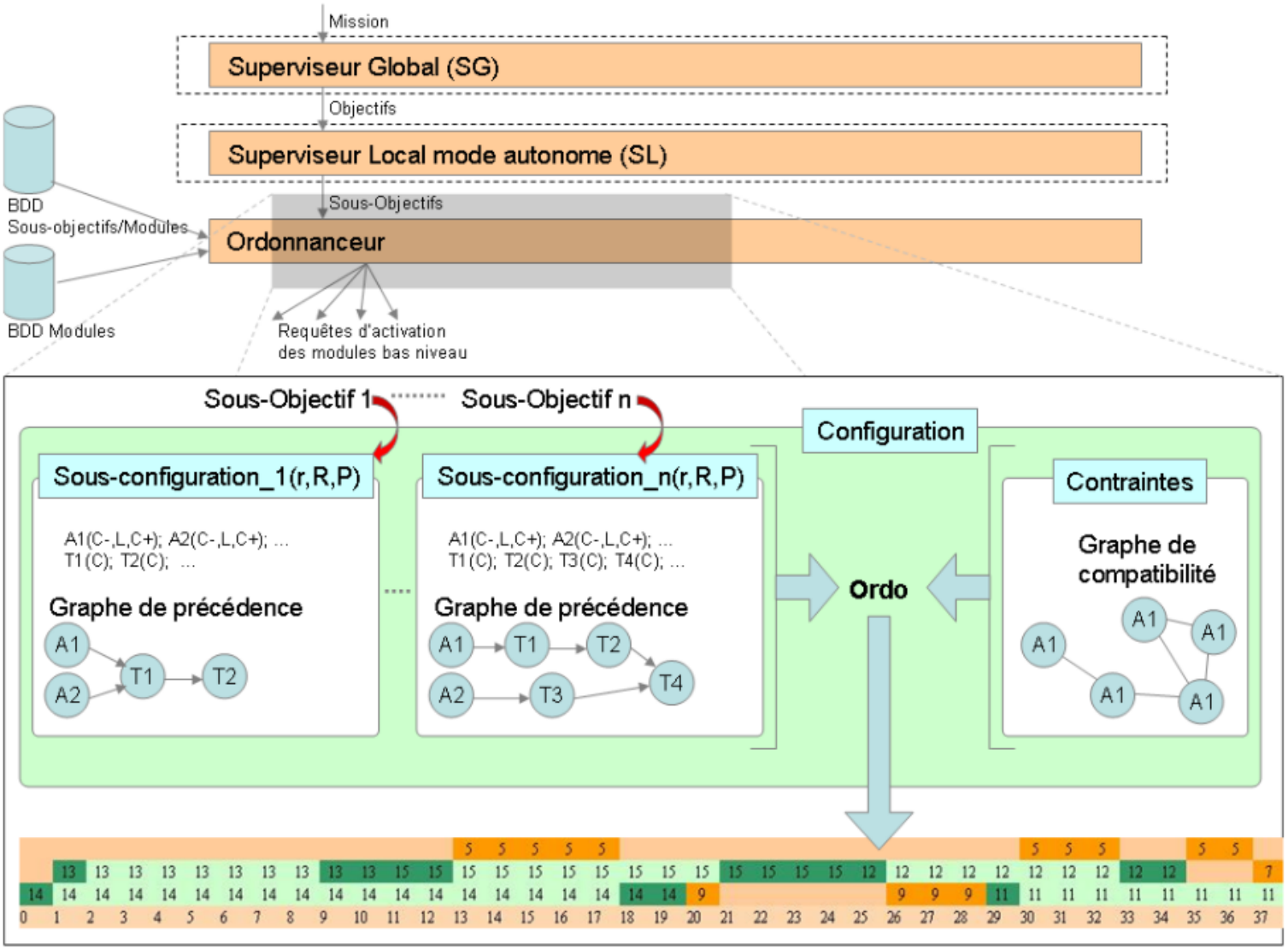
Entrée requêtes :
- Activation
- Arrêt
- Paramétrage
- Souscription (contextuelle) données
- Dessouscription (contextuelle) données
- Souscription (contextuelle) à un évènement
- Dessouscription (contextuelle) à un évènement



Architecture en 2 couches



L'ordonnanceur temps réel applicatif



Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
   - Adaptation des durées des modules
   - Niveau de priorités pour les schémas
   - Un système d'observation et de contrôle des évènements temporels
3. Proposition d'amélioration des schémas
4. Conclusion

Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
   - Adaptation des durées des modules
   - Niveau de priorités pour les schémas
   - Un système d'observation et de contrôle des évènements temporels
3. Proposition d'amélioration des schémas
4. Conclusion

Adaptation des durées des modules

Limites

- La durée d'exécution nominale est surestimée ;
- L'ordonnancement est sous-optimal.

Mise en œuvre :

- D'un système de statistique sur les durées d'exécutions des modules ;
- D'une formule pour adapter la durée d'exécution :

$$val_{estimated} = average + 1,5\sigma_n$$

- D'un algorithme d'adaptation.

Algorithme d'adaptation

Algorithme 1: Algorithme déterminant quand adapter

Entrées : M, le module à adapter

si $M.compteur \geq X$ **alors**
  si $M.tendance \nearrow$ **ou** $M.tendance \searrow$ **alors**
    si $M.moy > M.temps\_exec + \Delta$ **ou** $M.moy < M.temps\_exec - \Delta$ **alors**
      adapter
  $M.compteur \leftarrow 0$
sinon
  $M.compteur \leftarrow M.compteur + 1$

Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
   - Adaptation des durées des modules
   - Niveau de priorités pour les schémas
   - Un système d'observation et de contrôle des évènements temporels
3. Proposition d'amélioration des schémas
4. Conclusion

Niveau de priorités pour les schémas

Hiérarchisation des schémas

- Afin de ne pas avoir des schémas « égaux ».

Chemin suivi

- Mise en place d'un système de priorités à plusieurs niveaux ;
- Définition de la relation « plus haute priorité » notée : $\rightarrow$.

Example

Le schéma A est de plus haute priorité que le schéma B :

$$A \rightarrow B$$

Utilisation

Mise en place des priorités

1. Définition des priorités relatives entre les schémas ;
2. Création d'un graphe connexe avec les relations ;
3. Suppression des transitivités ;
4. Calcul des priorités globales en fonction de la hauteur de l'arbre.

Utilisation des priorités

1. Parmi les schémas de plus haute priorité :
   - Sélection d'un module avec la plus haute priorité (Algorithme ED) ;
2. Si on ne trouve pas refaire 1 avec un schéma de plus faible priorité.

Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
   - Adaptation des durées des modules
   - Niveau de priorités pour les schémas
   - Un système d'observation et de contrôle des évènements temporels
3. Proposition d'amélioration des schémas
4. Conclusion

Un système d'observation et de contrôle des évènements temporels

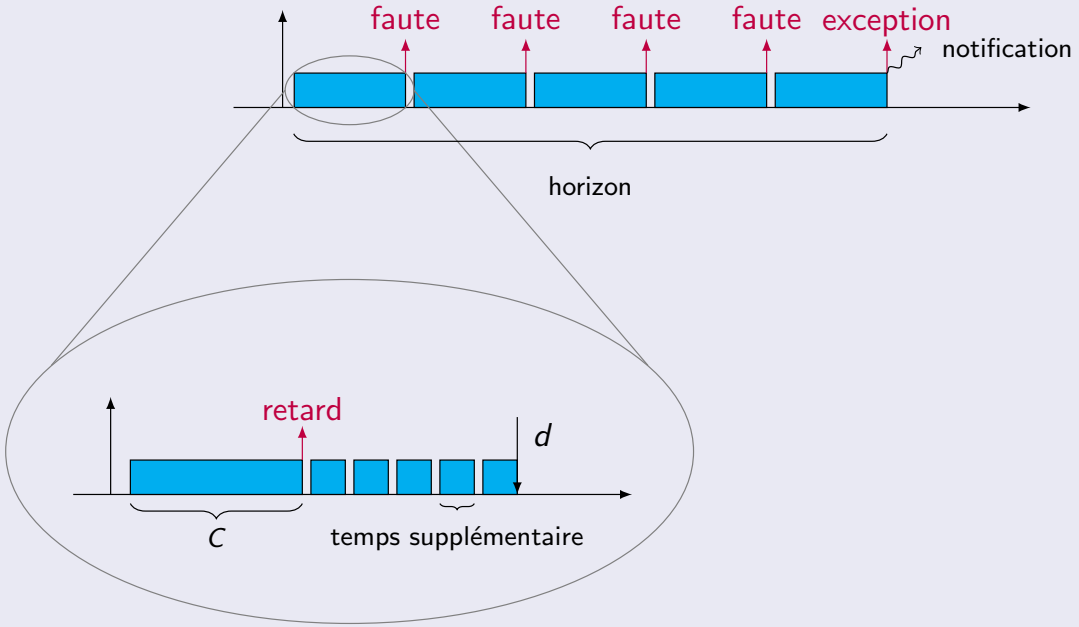
Limites

- Détections d'évènement basées sur la durée nominale du module ;
- Actuellement, ne permet pas une détection à grain fin :
  - des modules et des schémas impliqués
  - de la sévèrité de la faute

Propositions

- Durée nominale = Contrainte temporelle ;
- 3 niveaux d'évènements temporels :
  - Le retard ;
  - La faute ;
  - et l'exception.

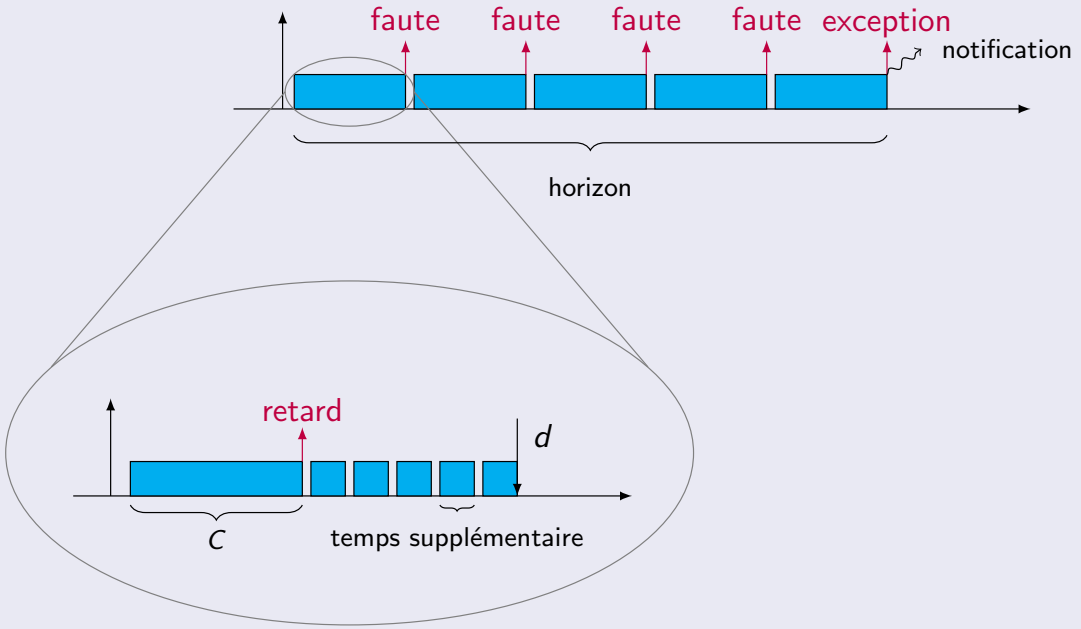
Schématisation des concepts de retard, faute et exception



La réaction aux retards

$$add\_time = (max - adapt) \times \frac{1}{delay\_allowed}$$

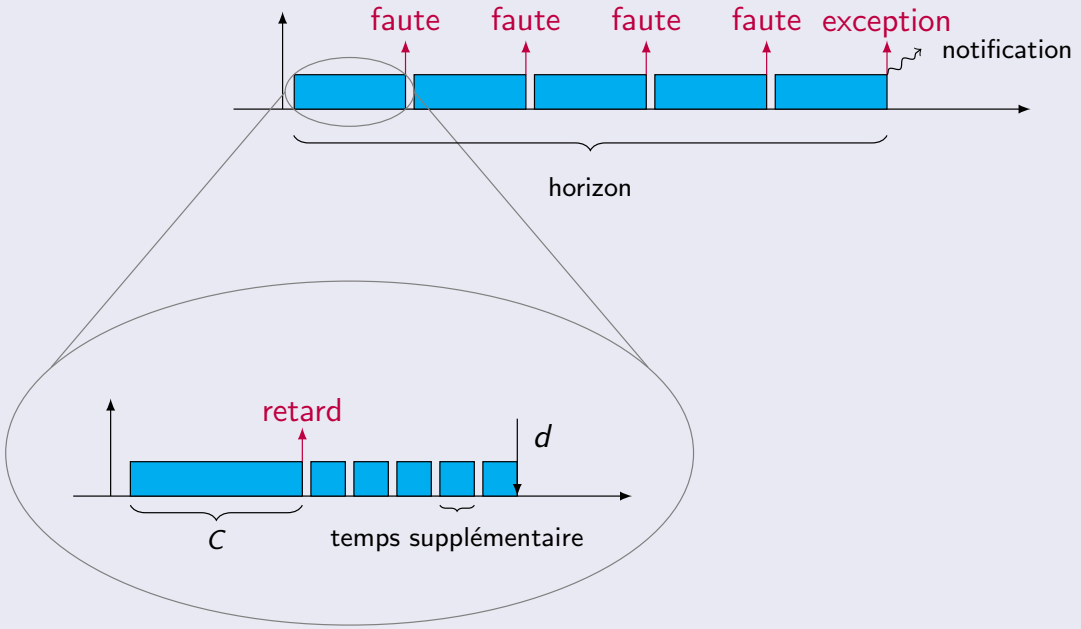
Schématisation des concepts de retard, faute et exception



Les réactions possible aux fautes

- Le module continue son exécution ;
- On arrête immédiatement l'exécution du module ;
- On arrête l'exécution juste avant la prochaine exécution du module.

Schématisation des concepts de retard, faute et exception



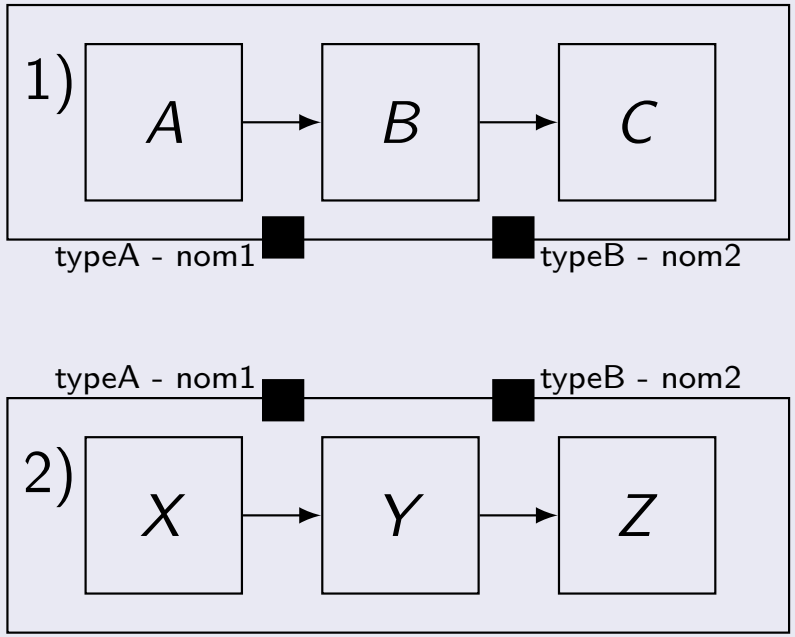
La réactions aux exceptions

- Notification auprès des superviseurs ;
- **notification** si **nombre faute** sur **horizon** ;

Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
3. Proposition d'amélioration des schémas
4. Conclusion

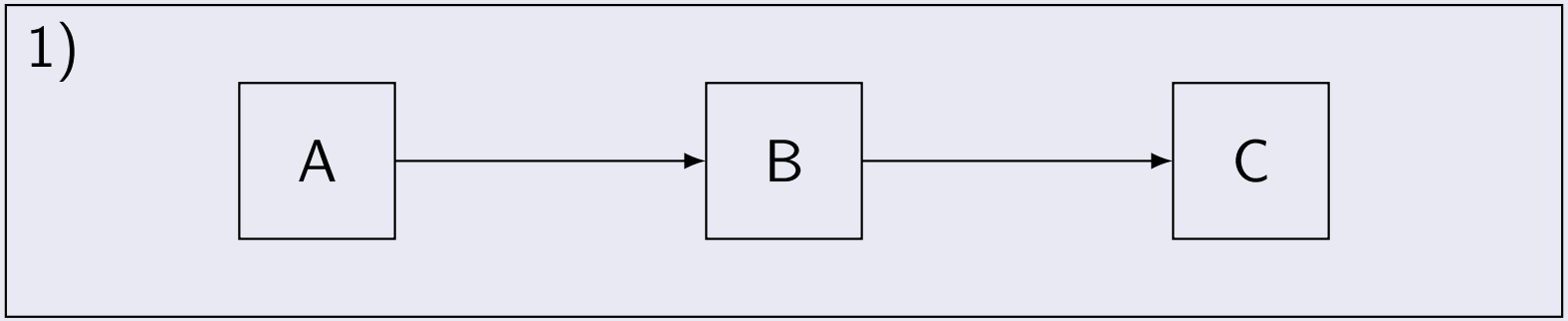
Schéma comme brique logicielle



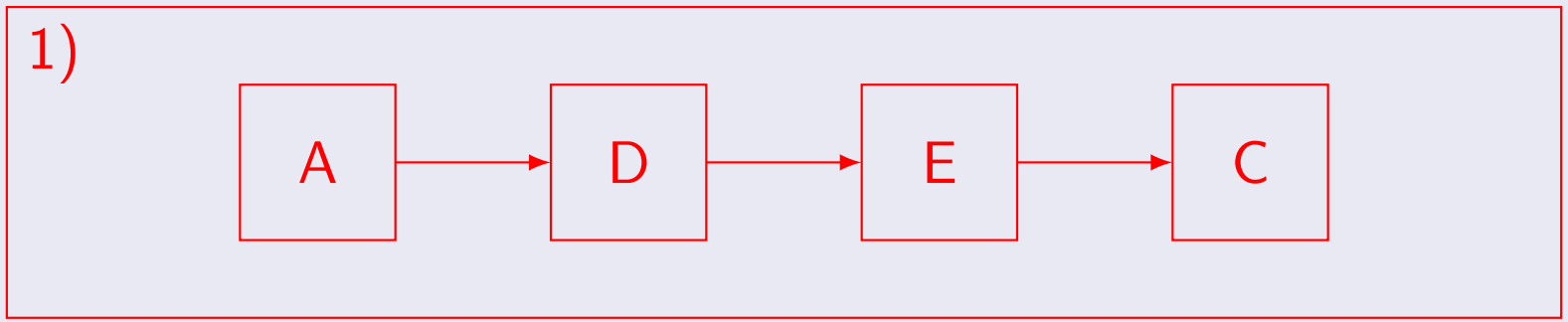
Les interfaces

- Même principe que les ports des modules :
  - direction (input ou output)
  - type de donnée
  - un nom (identifiant)

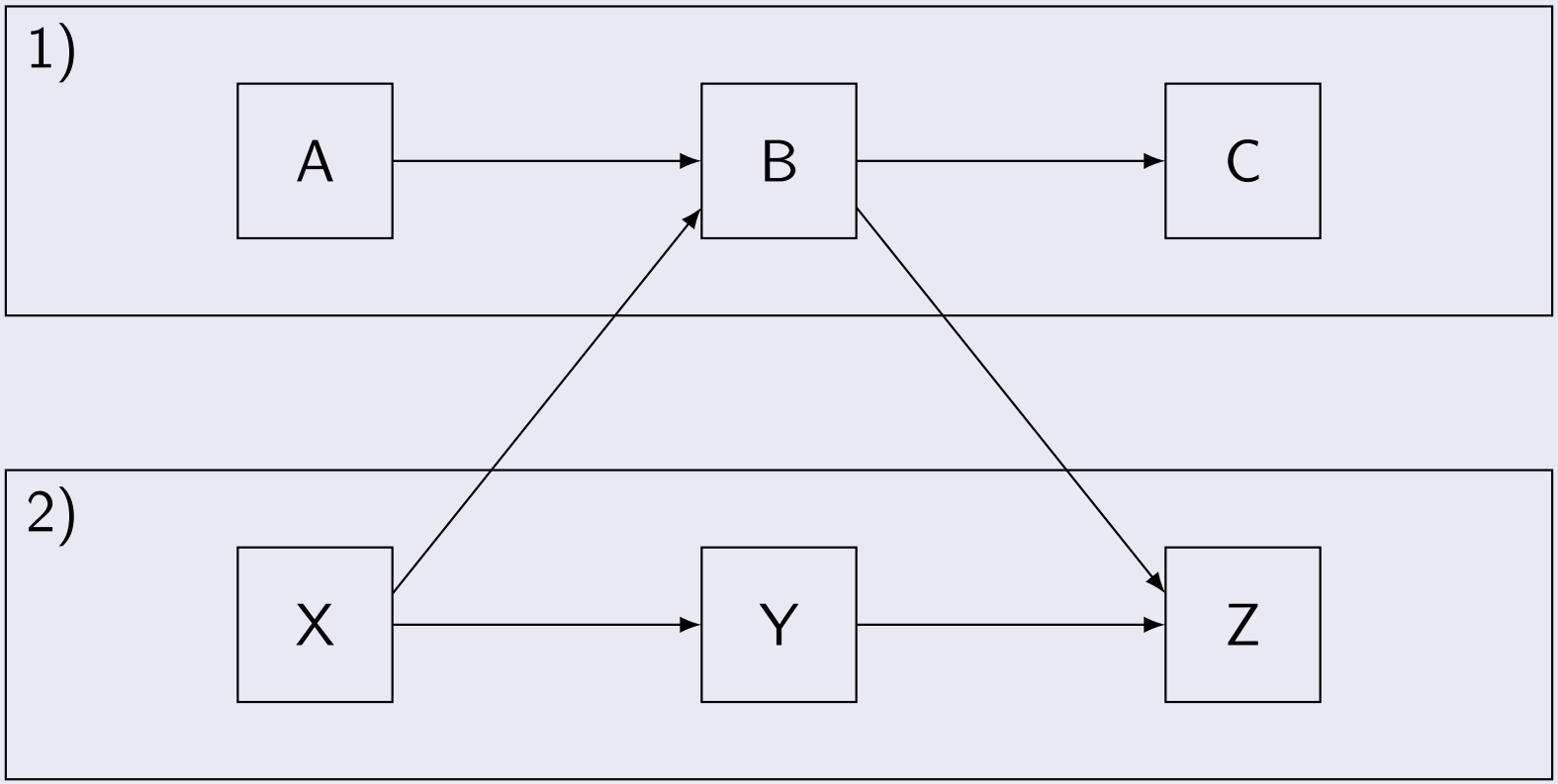
Commutation d'un schéma



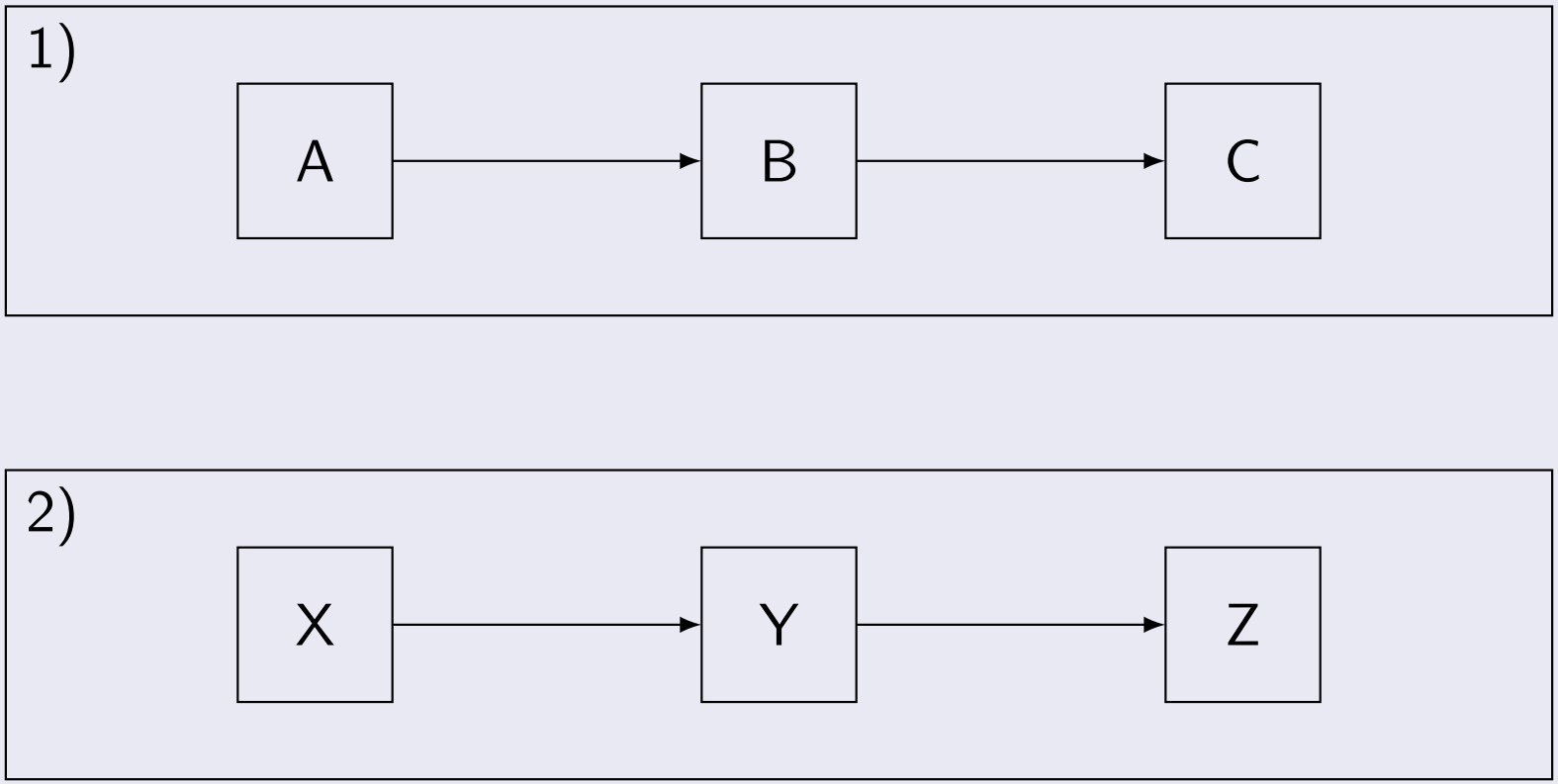
Commutation d'un schéma



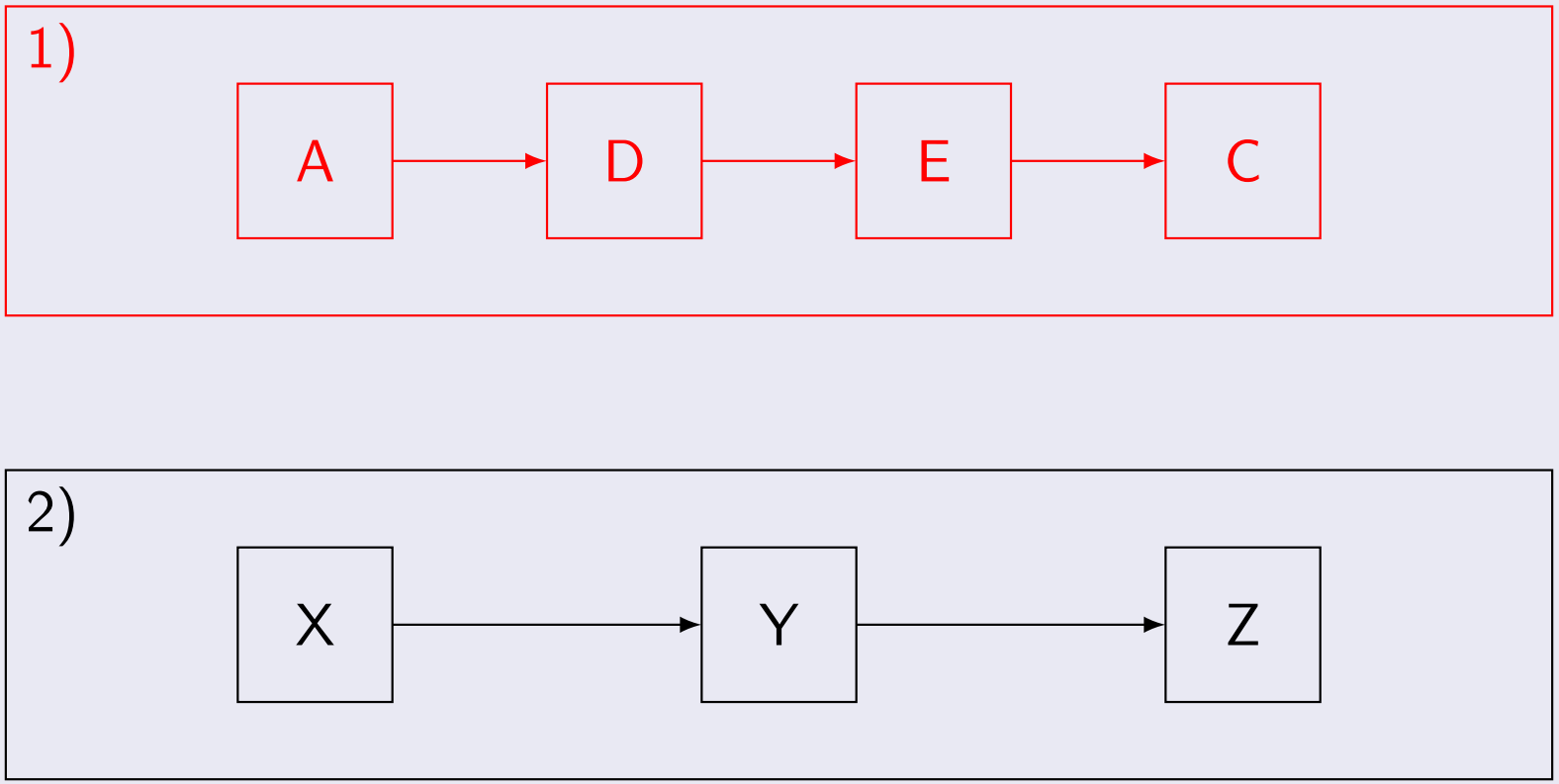
Commutation multischémas



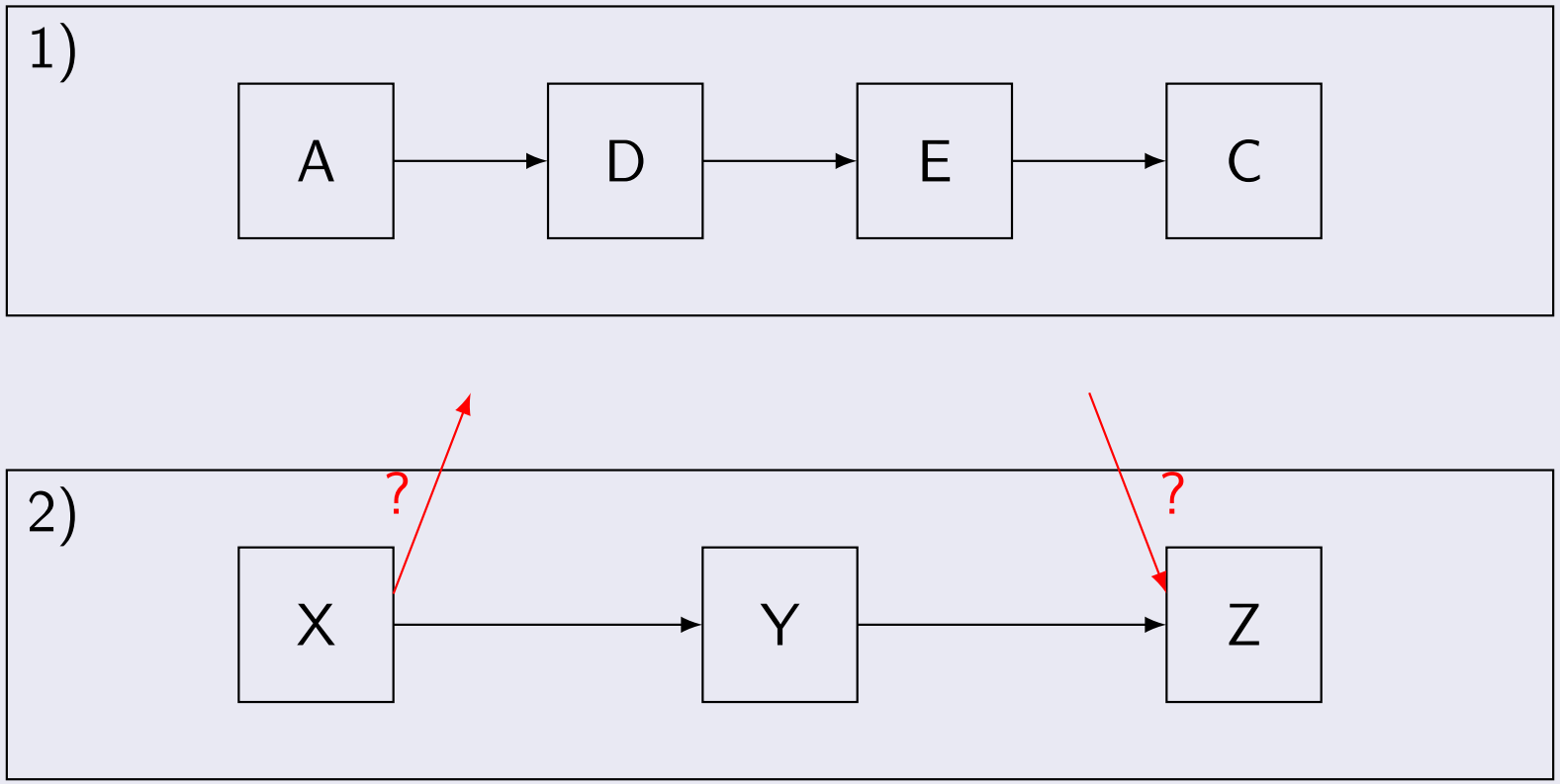
Commutation multischémas



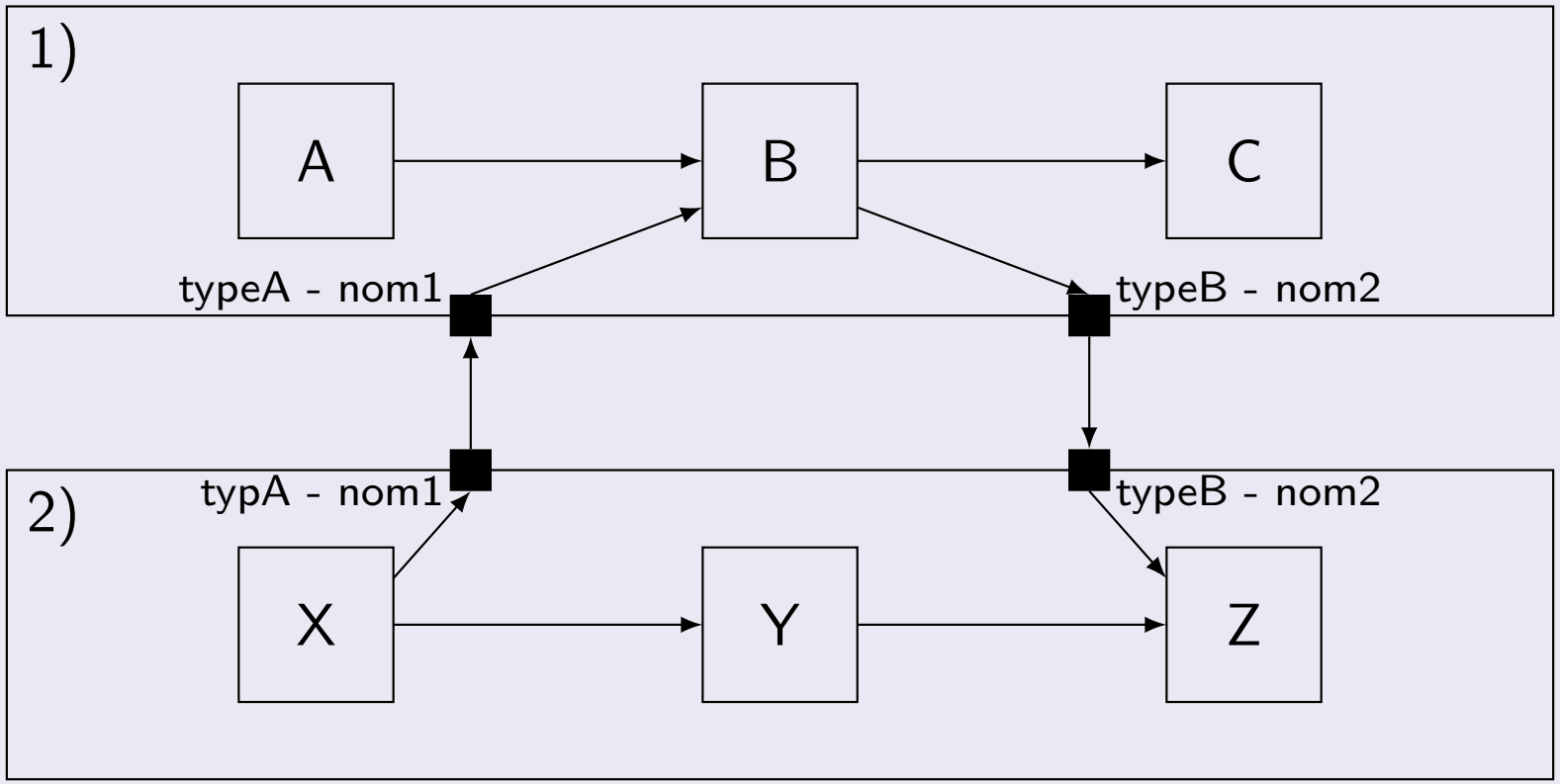
Commutation multischémas



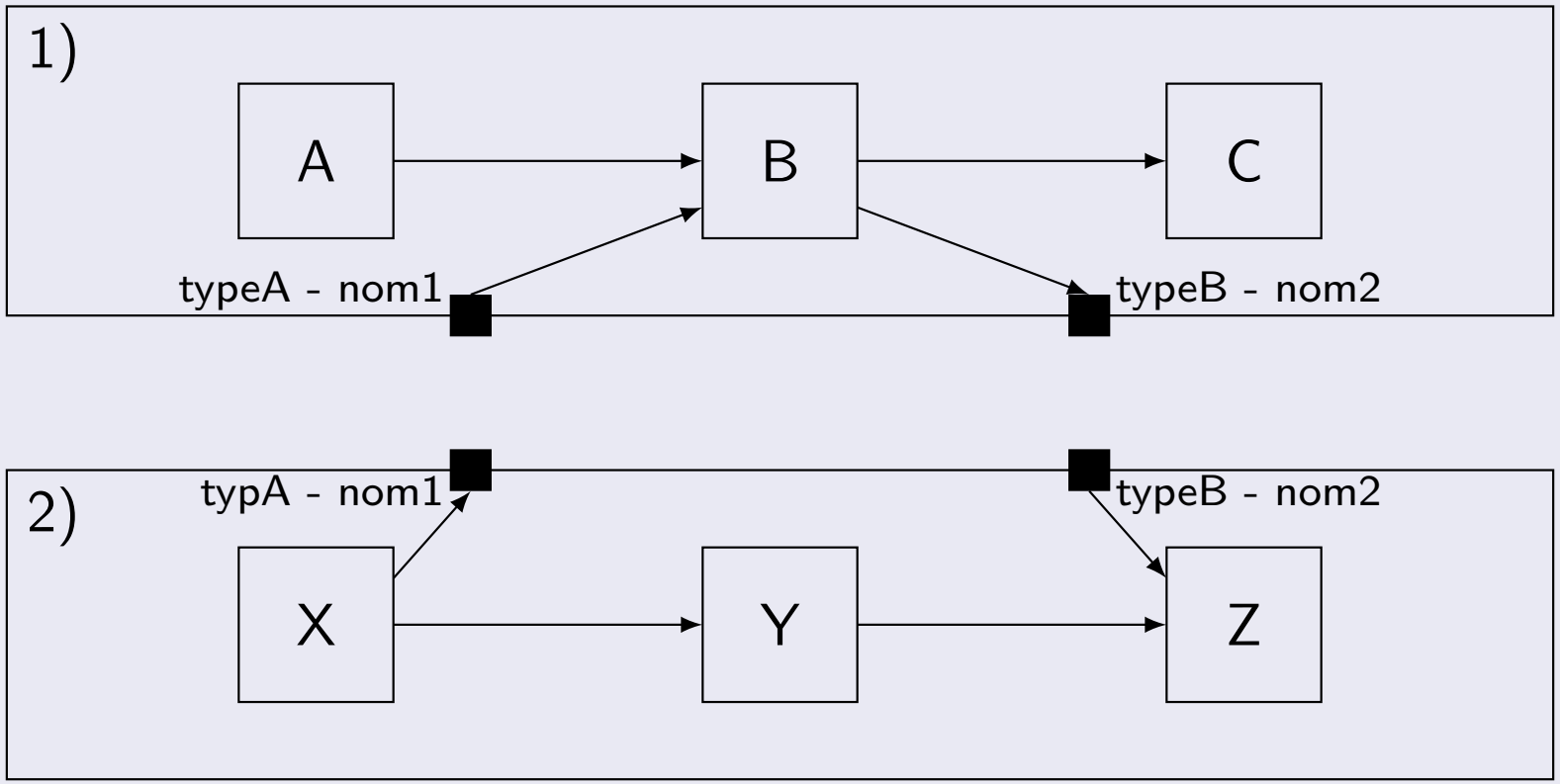
Commutation multischémas



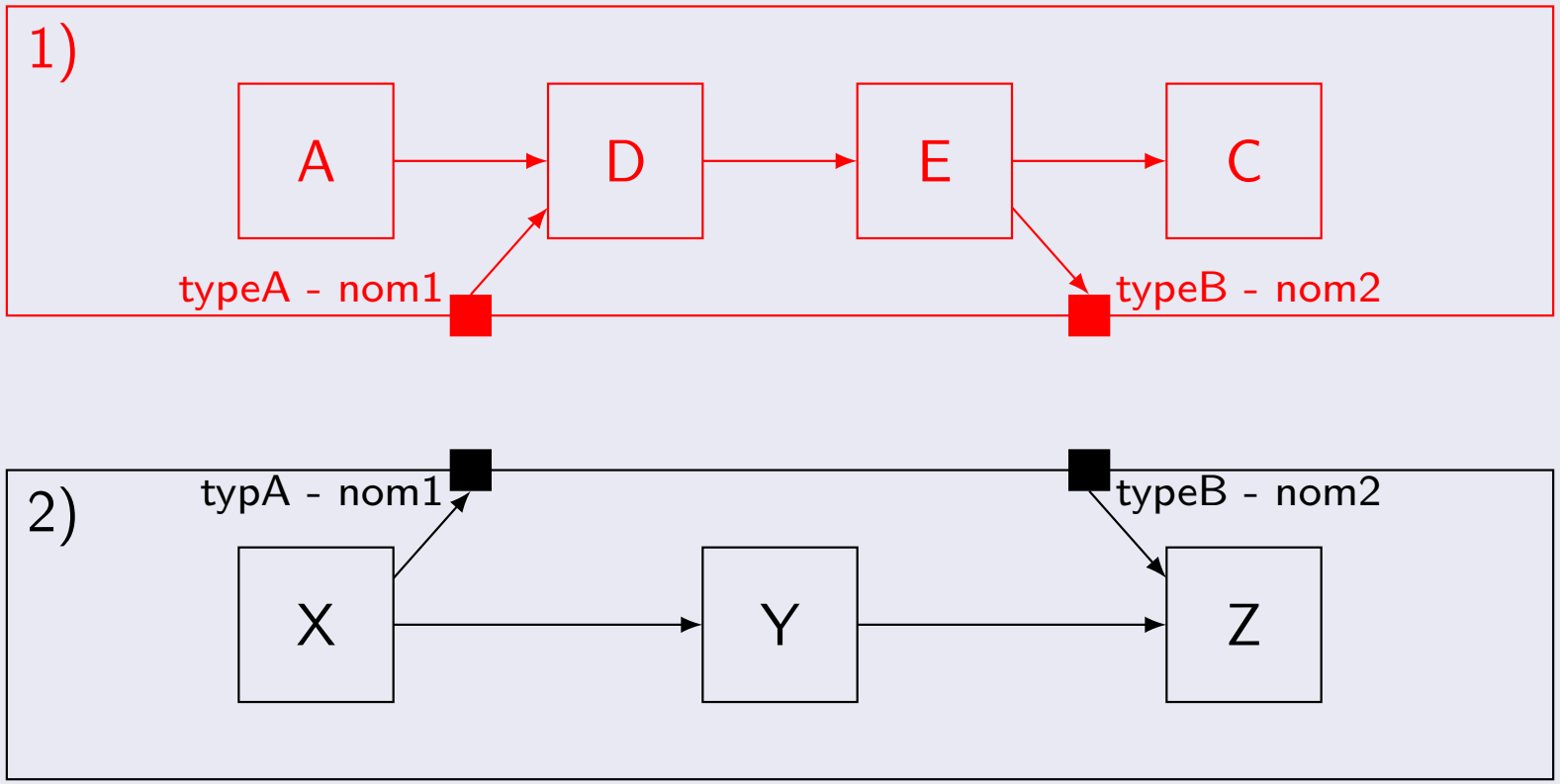
Commutation multischémas : Nouvelle approche



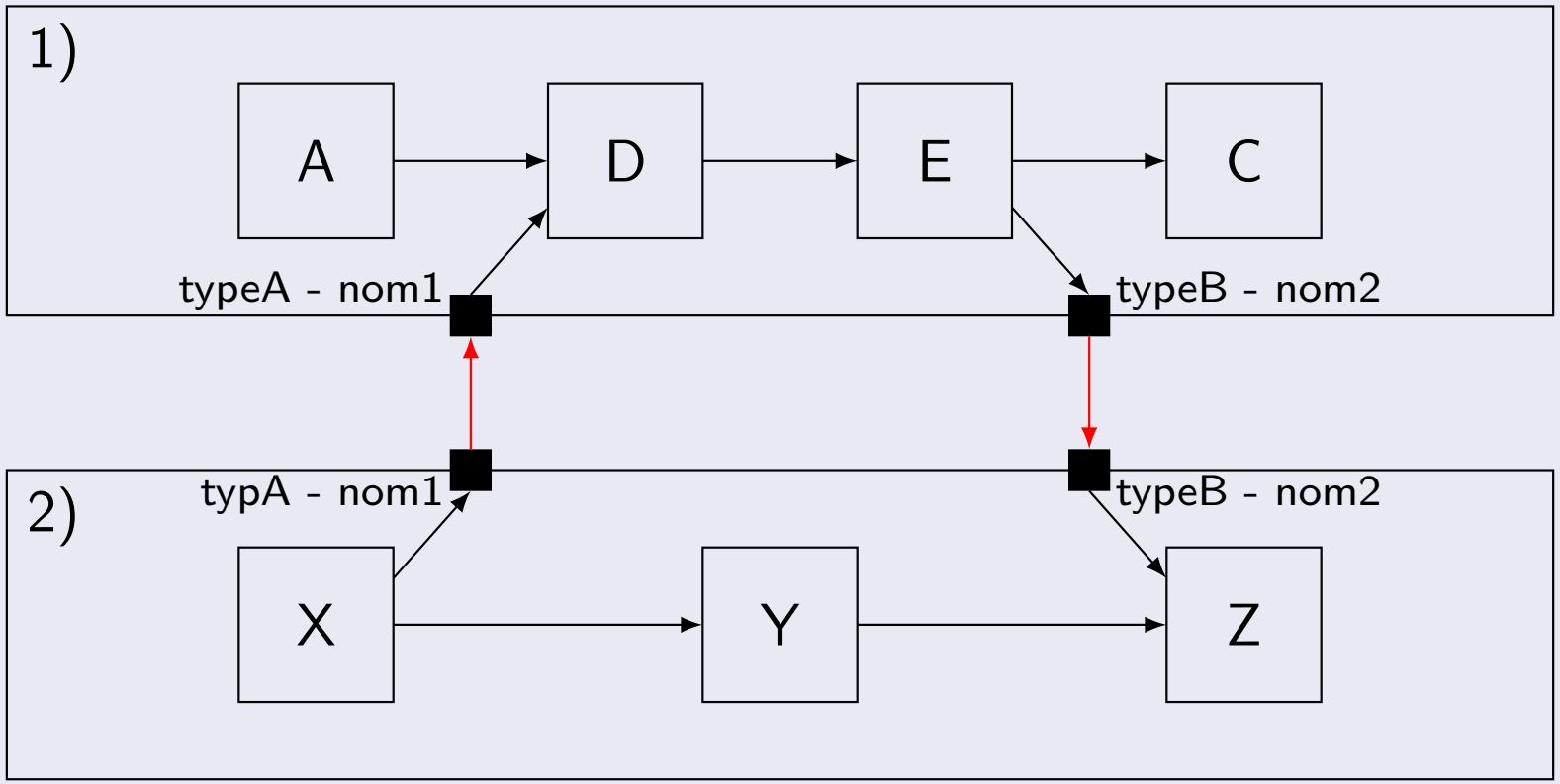
Commutation multischémas : Nouvelle approche



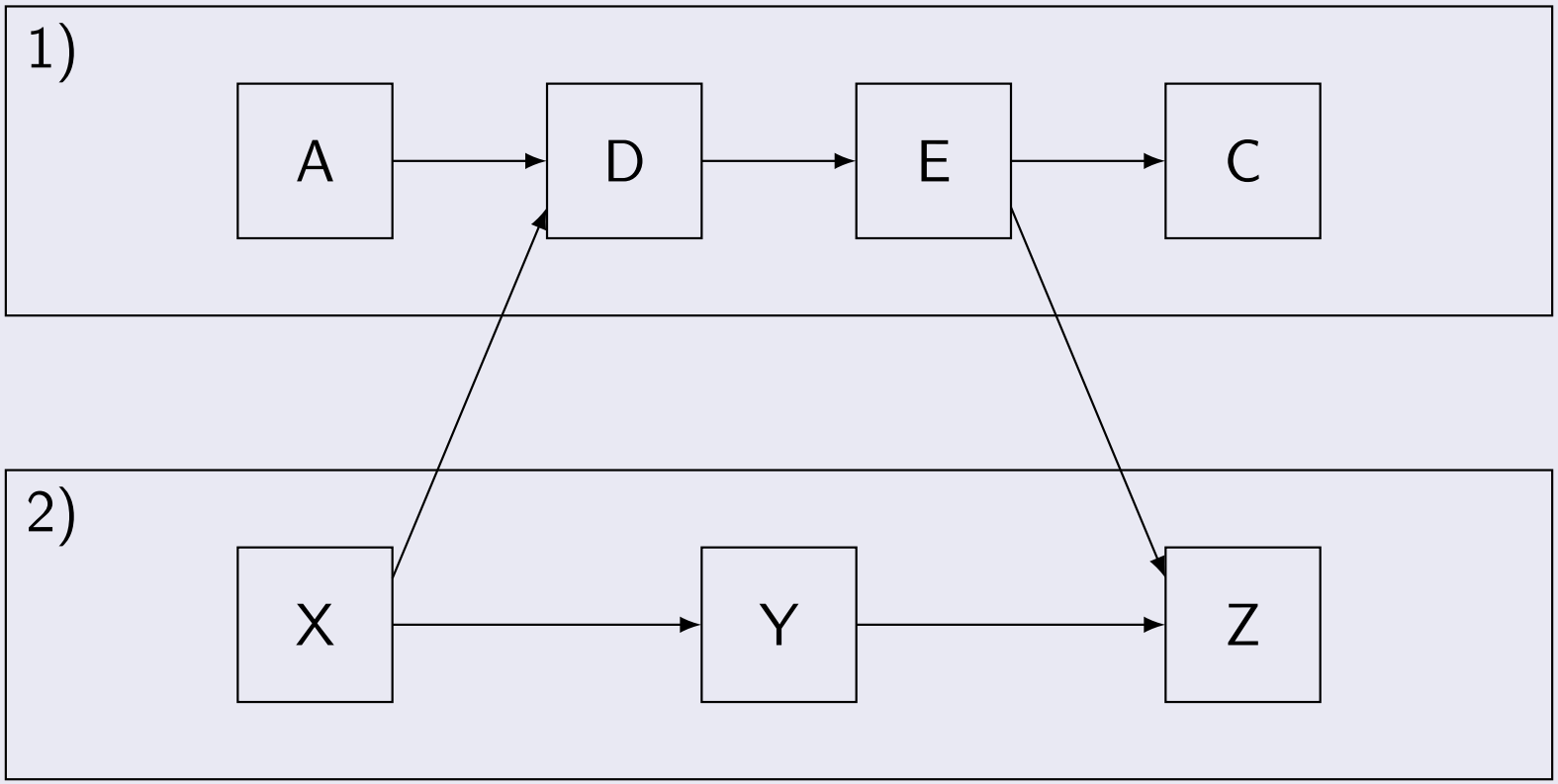
Commutation multischémas : Nouvelle approche



Commutation multischémas : Nouvelle approche



Commutation multischémas : Nouvelle approche



Plan

1. ContrACT : état des lieux
2. Propositions d'amélioration de l'ordonnanceur
3. Proposition d'amélioration des schémas
4. Conclusion

Conclusion

- Ajout à l'architecture de nouvelle caractéristique :
  - Gestion des évènements temporels ;
  - Adaptation des durées d'exécutions ;
  - Priorité entre schémas ;
  - ...
- Simplification du développement grâce à la réification des schémas ;
- Devraient aider à la mise en place d'architecture tolérante aux fautes.

Merci de votre attention.

Des questions ?