Approche formelle et opérationnelle de la multimodélisation et de la simulation des systèmes complexes

DEVS / VLE

Gauthier Quesnel

Institut National de la Recherche Agronomique

Plan

Plan

Évolution des pratiques et des objectifs de modélisation :

- volonté de faire une synthèse des connaissances et des données
- les problématiques deviennent systémiques
- des formalismes différents pour des sous-systèmes différents
- des modèles pour l'aide à la décision

Quelques conséquences de cette évolution :

- nécessité de coupler des modèles hétérogènes
- problèmes formels et opérationnels difficiles
- explosion du nombre de paramètres
- besoins de méthodes et d'outils

Évolution des pratiques et des objectifs de modélisation :

- volonté de faire une synthèse des connaissances et des données
- les problématiques deviennent systémiques
- des formalismes différents pour des sous-systèmes différents
- des modèles pour l'aide à la décision

Quelques conséquences de cette évolution :

- nécessité de coupler des modèles hétérogènes
- problèmes formels et opérationnels difficiles
- explosion du nombre de paramètres
- besoins de méthodes et d'outils

Contexte

En informatique et automatique

La multi-modélisation est un champ de recherche en soi :

- Comment coupler les modèles ?
- Comment coupler les simulateurs ?
- Comment optimiser les calculs ?

La solution formelle et opérationnelle retenue :

- DEVS, *Discrete Event System Specification*
- Développement d'une plate-forme basée sur DEVS
- VLE : un laboratoire virtuel pour l'expérimentation et l'aide à la décision

Contexte

En informatique et automatique

La multi-modélisation est un champ de recherche en soi :

- Comment coupler les modèles ?
- Comment coupler les simulateurs ?
- Comment optimiser les calculs ?

La solution formelle et opérationnelle retenue :

- DEVS, *Discrete Event System Specification*
- Développement d'une plate-forme basée sur DEVS
- VLE : un laboratoire virtuel pour l'expérimentation et l'aide à la décision

Plan

DEVS, Discrete Event System Specification

Introduction

Un formalisme systémique de modélisation :

- événements discrets
- ensembles, états, fonctions de transition d'états
- approche modulaire et hiérarchique

DEVS à été initié par B. P. Zeigler en 1976 :

- +50 simulateurs DEVS (libres/payants, spécialisés etc.)
- Monde : États-Unis (Univ. Arizona), Canada (Univ. Mc Gill), Portugal (Univ. Calombra), Corée du Sud (Univ. Hang Kong), Allemagne (Univ. Rostock), ...
- France : Calais (Univ ulco), Clermont-Ferrand (Isima), en Corse (Univ. de Corse), Marseille (Univ Aix 3), Montpellier (Cirad), ...

DEVS, Discrete Event System Specification

Introduction

Un formalisme systémique de modélisation :

- événements discrets
- ensembles, états, fonctions de transition d'états
- approche modulaire et hiérarchique

DEVS à été initié par B. P. Zeigler en 1976 :

- +50 simulateurs DEVS (libres/payants, spécialisés etc.)
- Monde : États-Unis (Univ. Arizona), Canada (Univ. Mc Gill), Portugal (Univ. Calombra), Corée du Sud (Univ. Hang Kong), Allemagne (Univ. Rostock), ...
- France : Calais (Univ ulco), Clermont-Ferrand (Isima), en Corse (Univ. de Corse), Marseille (Univ Aix 3), Montpellier (Cirad), ...

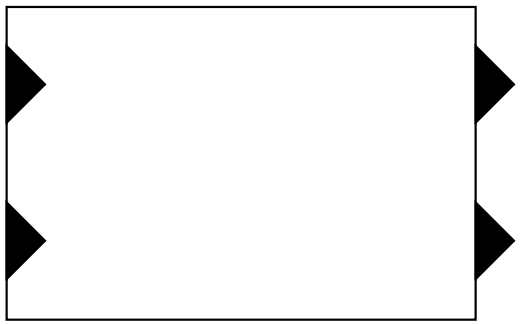
DEVS [Zeigler, 1976] :

- un **formalisme de haut niveau** (généralité)
  → DEVS « encapsule » les formalismes
- propose un **cadre formel** pour le couplage de modèles
  → Propriété de fermeture sous couplage
- utilise le paradigme des **événements discrets**
  → Les événements pilotent la simulation
- intègre un cadre opérationnel : les **simulateurs abstraits**
  → Un ensemble d'algorithmes

- Définition du modèle atomique :

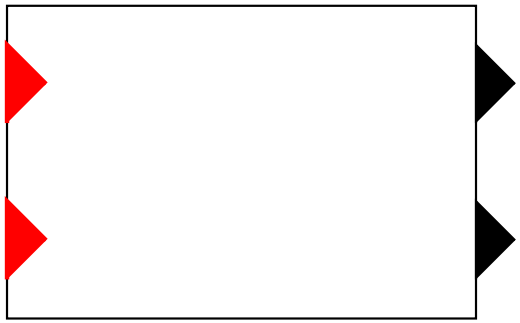
$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

  - Une structure mathématique composée de variables et de fonctions
  - Une représentation graphique :



DEVS, Discrete Event System Specification

Description du modèle atomique

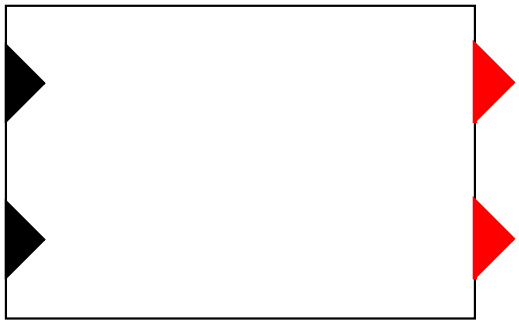


$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

X : l'ensemble des ports d'entrée et des valeurs attachées.

DEVS, Discrete Event System Specification

Description du modèle atomique

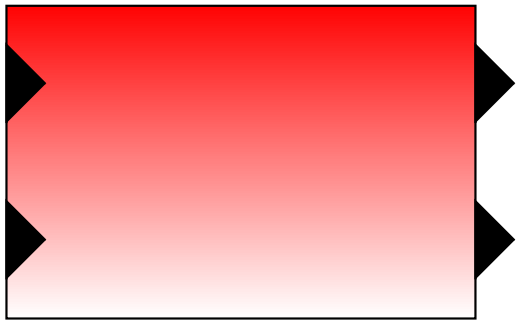


$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

Y : l'ensemble des ports de sortie et des valeurs attachées.

DEVS, Discrete Event System Specification

Description du modèle atomique

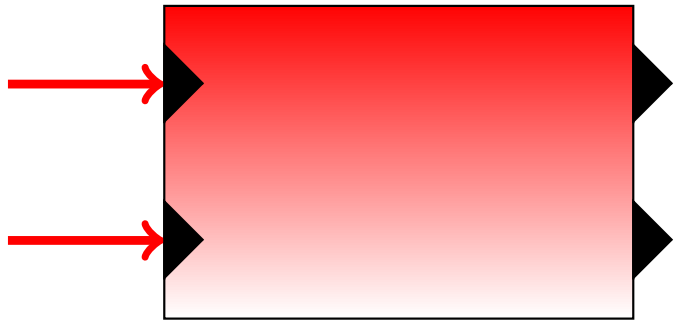


$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

S : l'ensemble des états du système.

DEVS, Discrete Event System Specification

Description du modèle atomique

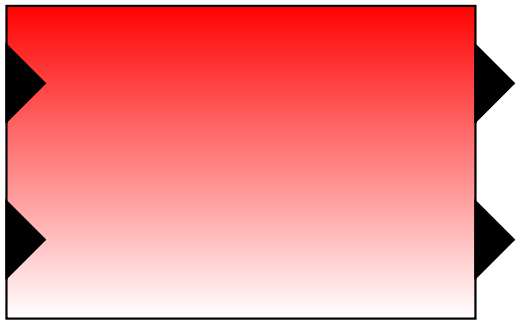


$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

δ_{ext} : fonction de transition externe : $\delta_{ext} : S \times X \rightarrow S$
représente les réponses du système aux événements externes.

DEVS, Discrete Event System Specification

Description du modèle atomique

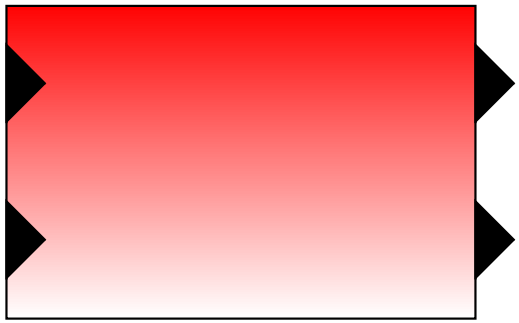


$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

δ_{int} : fonction de transition interne : $\delta_{int} : S \rightarrow S$
représente les évolutions autonomes.

DEVS, Discrete Event System Specification

Description du modèle atomique

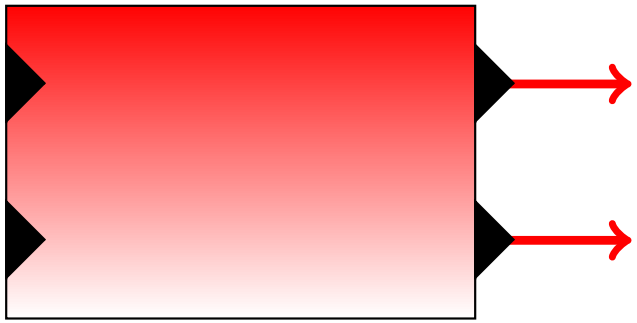


$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

$ta(s)$: le temps pendant lequel le modèle reste dans l'état S (hors événements externes)

DEVS, Discrete Event System Specification

Description du modèle atomique



$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda \rangle$$

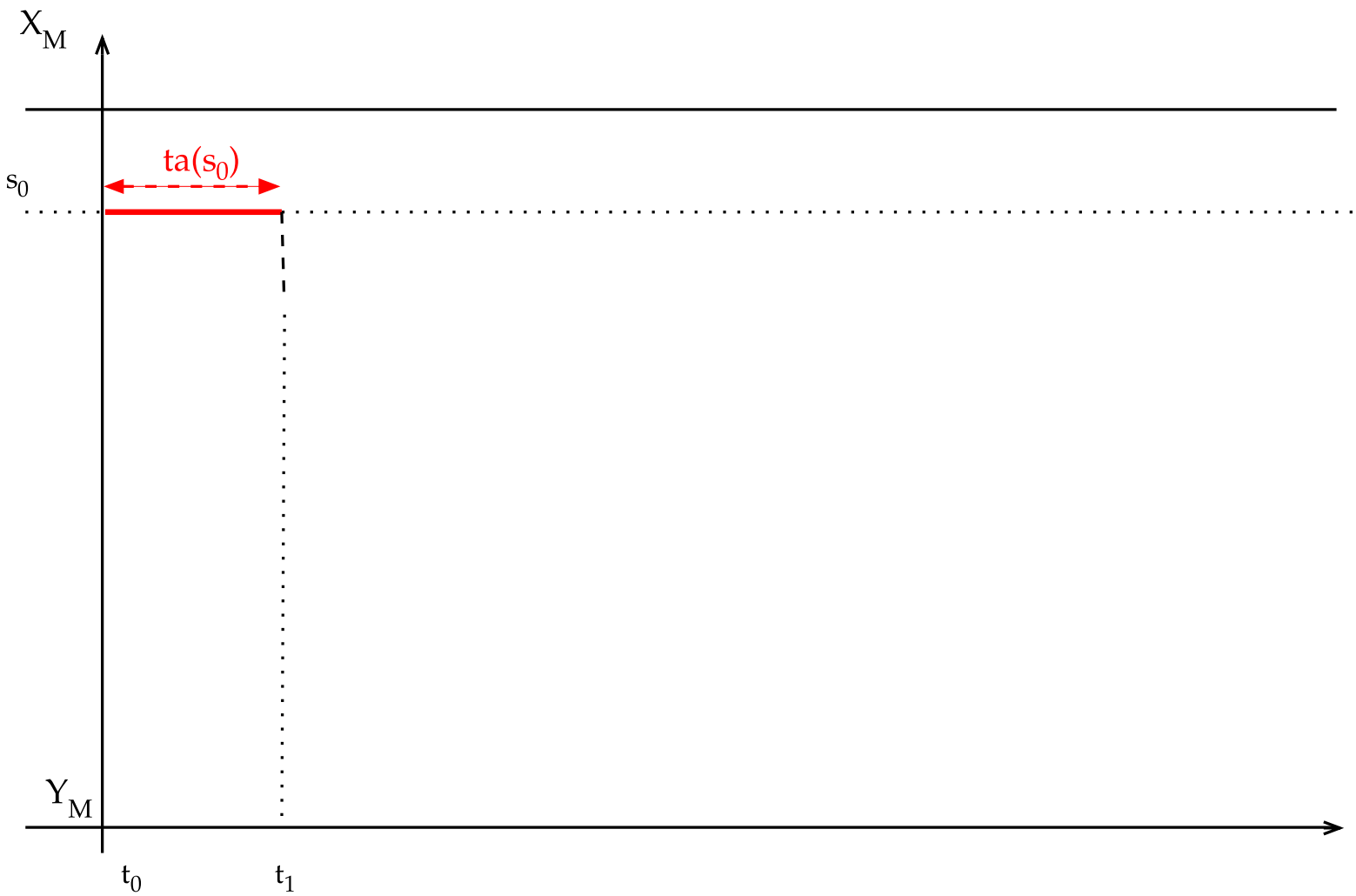
λ : la fonction de sortie : $\lambda : S \rightarrow Y$ représente les influences externes.

DEVS, Discrete Event System Specification

Exemple de graphe temporel

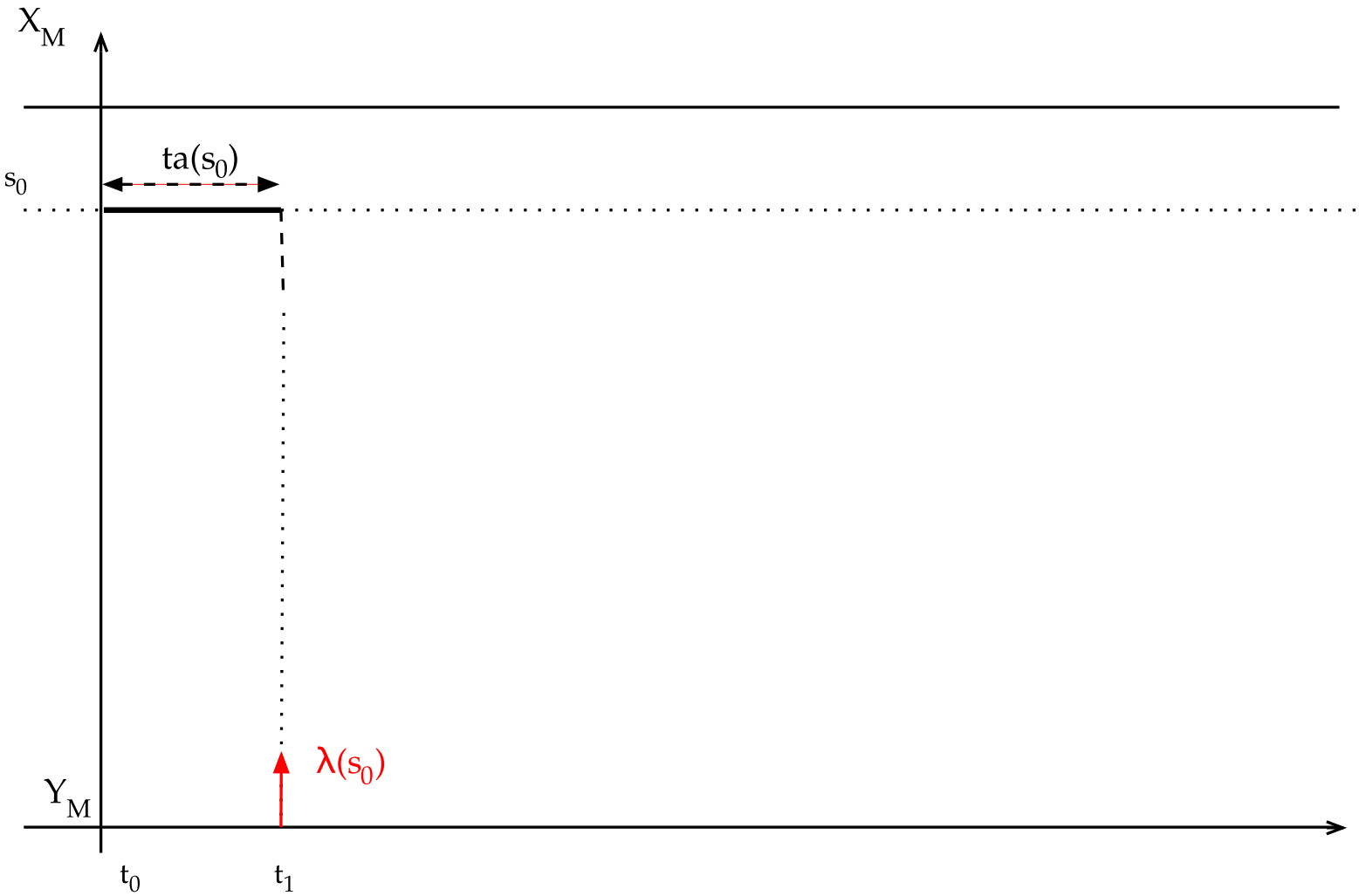
DEVS, Discrete Event System Specification

Exemple de graphe temporel



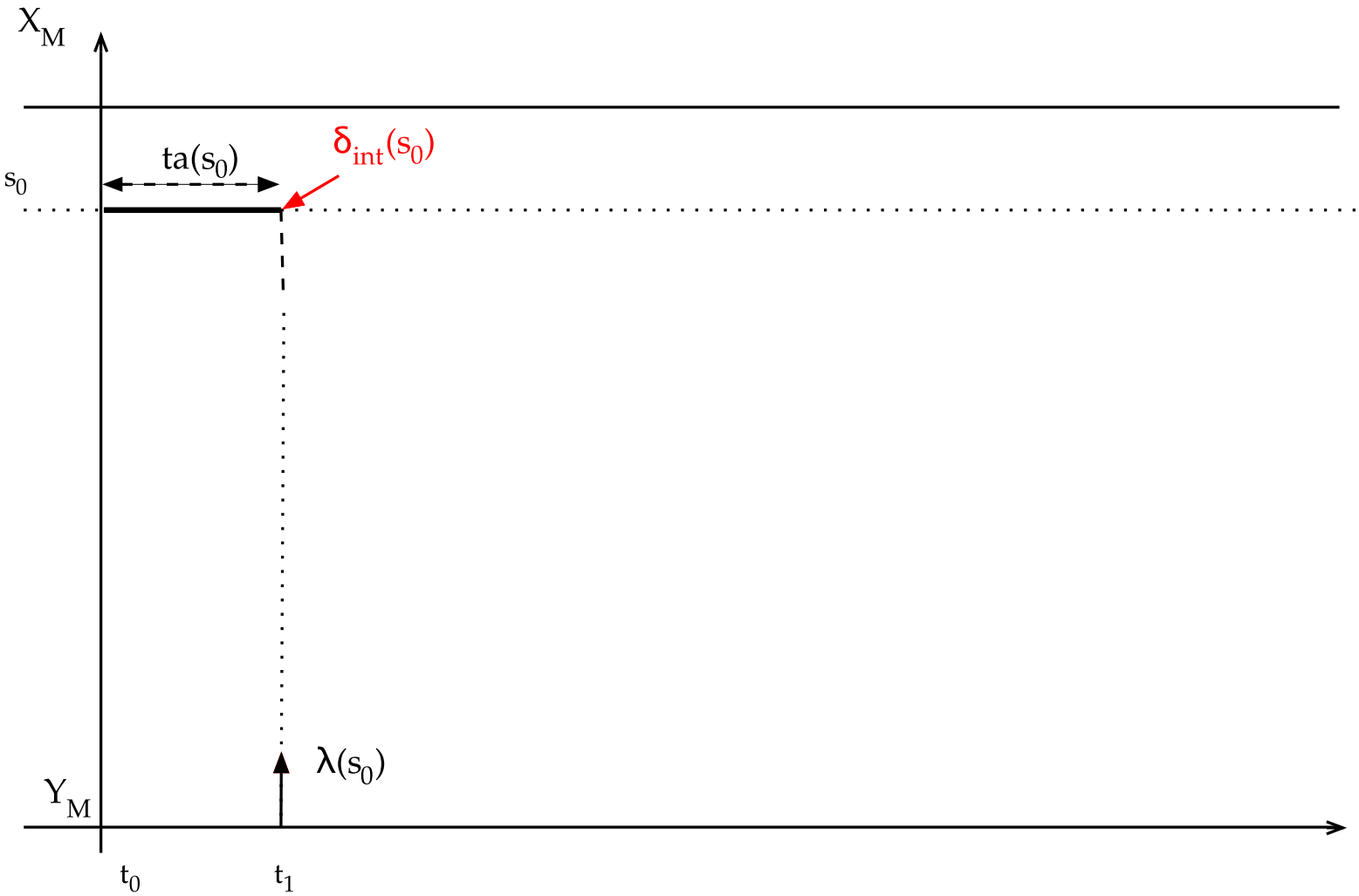
DEVS, Discrete Event System Specification

Exemple de graphe temporel



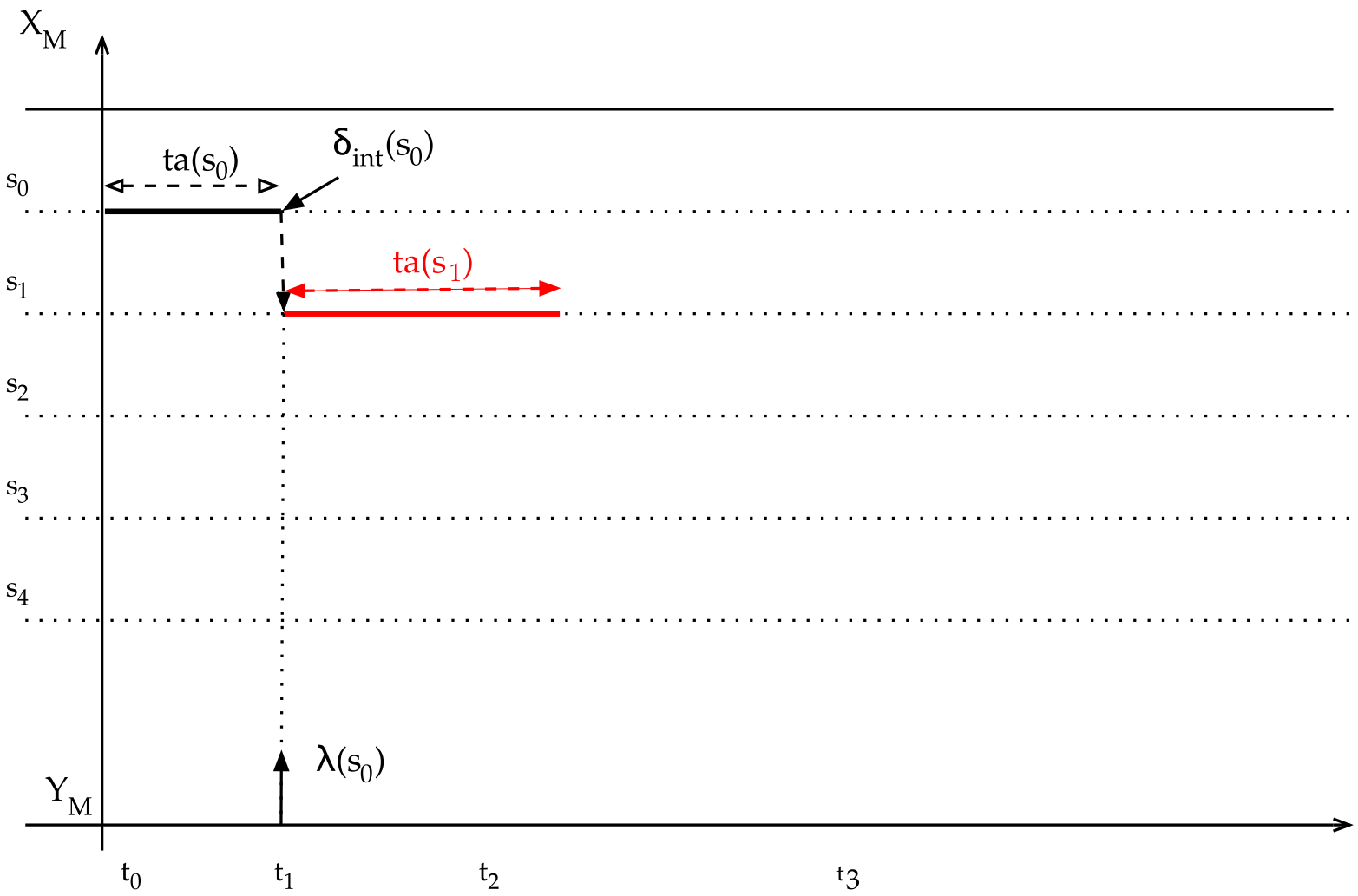
DEVS, Discrete Event System Specification

Exemple de graphe temporel



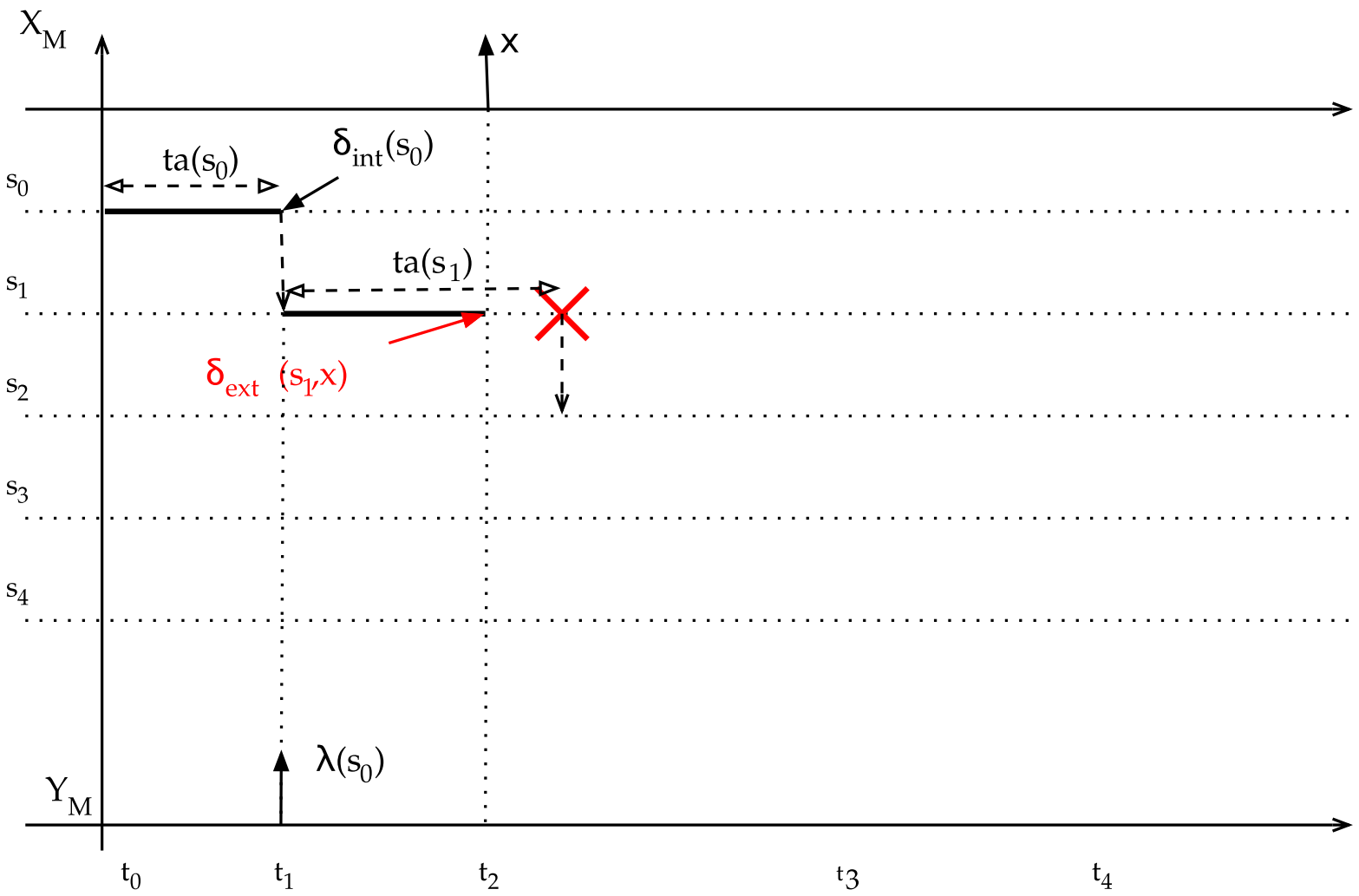
DEVS, Discrete Event System Specification

Exemple de graphe temporel



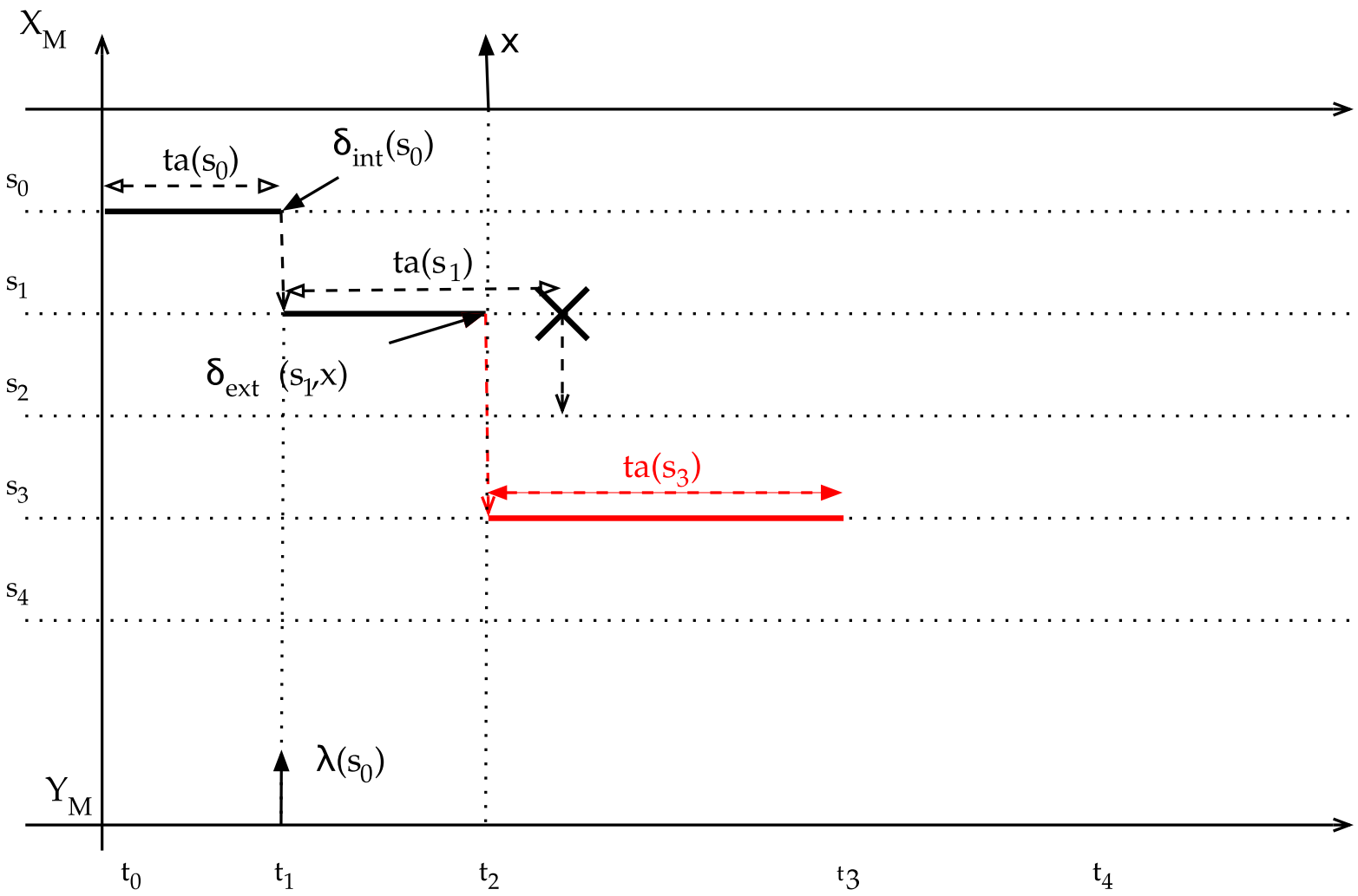
DEVS, Discrete Event System Specification

Exemple de graphe temporel



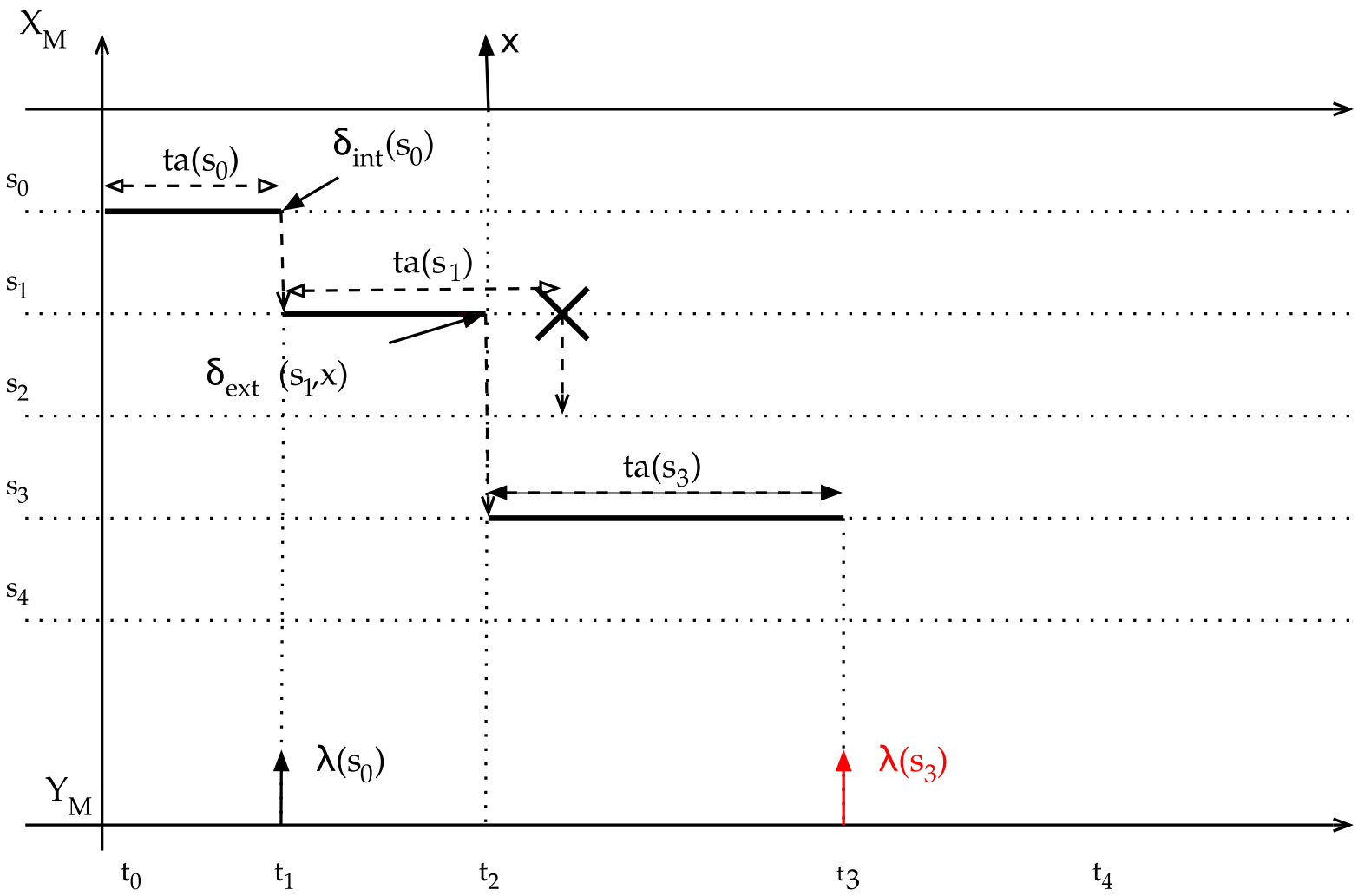
DEVS, Discrete Event System Specification

Exemple de graphe temporel



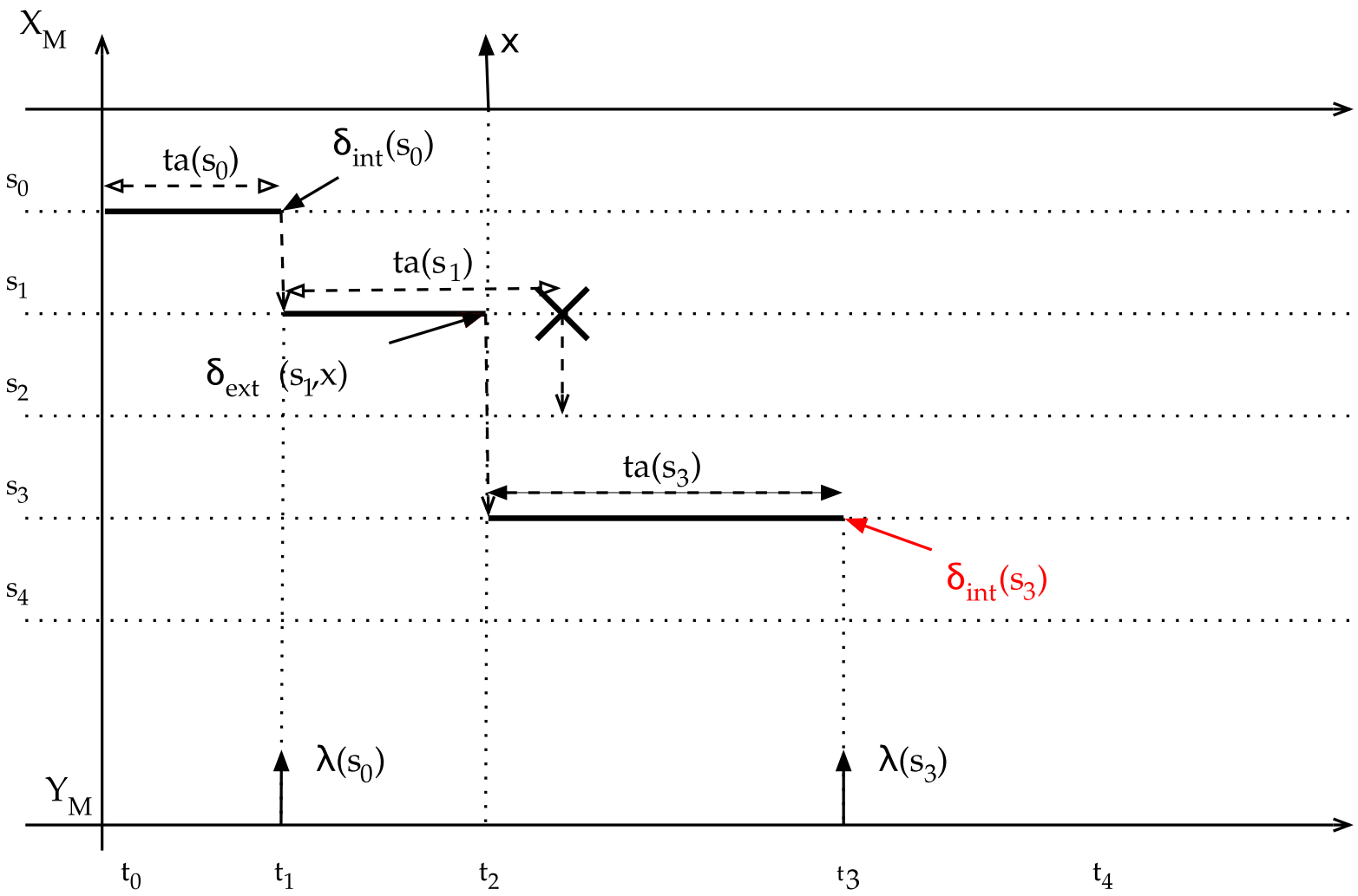
DEVS, Discrete Event System Specification

Exemple de graphe temporel



DEVS, Discrete Event System Specification

Exemple de graphe temporel



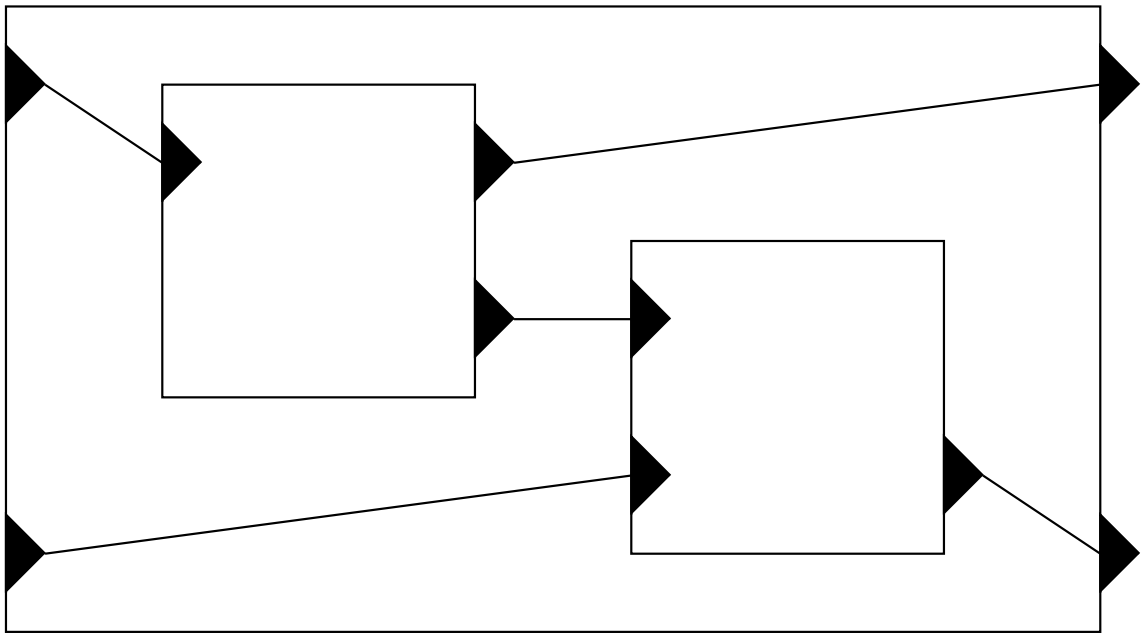
DEVS, Discrete Event System Specification

Description du modèle couplé

- Définition du modèle couplé (ou réseau de modèles) :

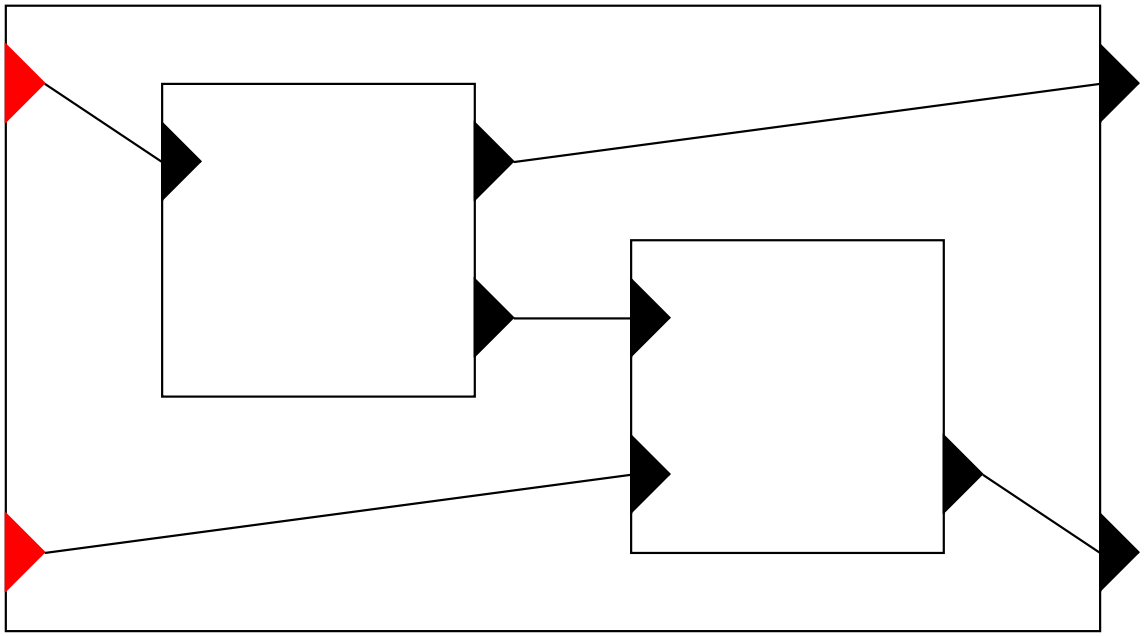
$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

  - Une structure mathématique composée uniquement de variables
  - Une représentation graphique :



DEVS, Discrete Event System Specification

Description du modèle couplé

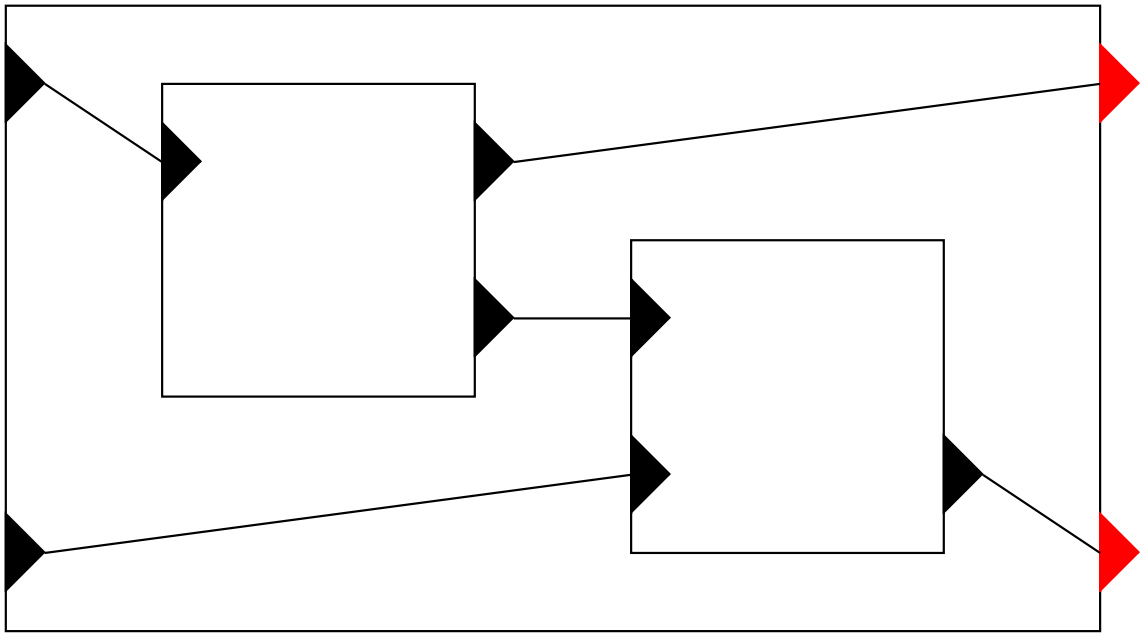


$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

X l'ensemble des ports d'entrée et des valeurs associées.

DEVS, Discrete Event System Specification

Description du modèle couplé

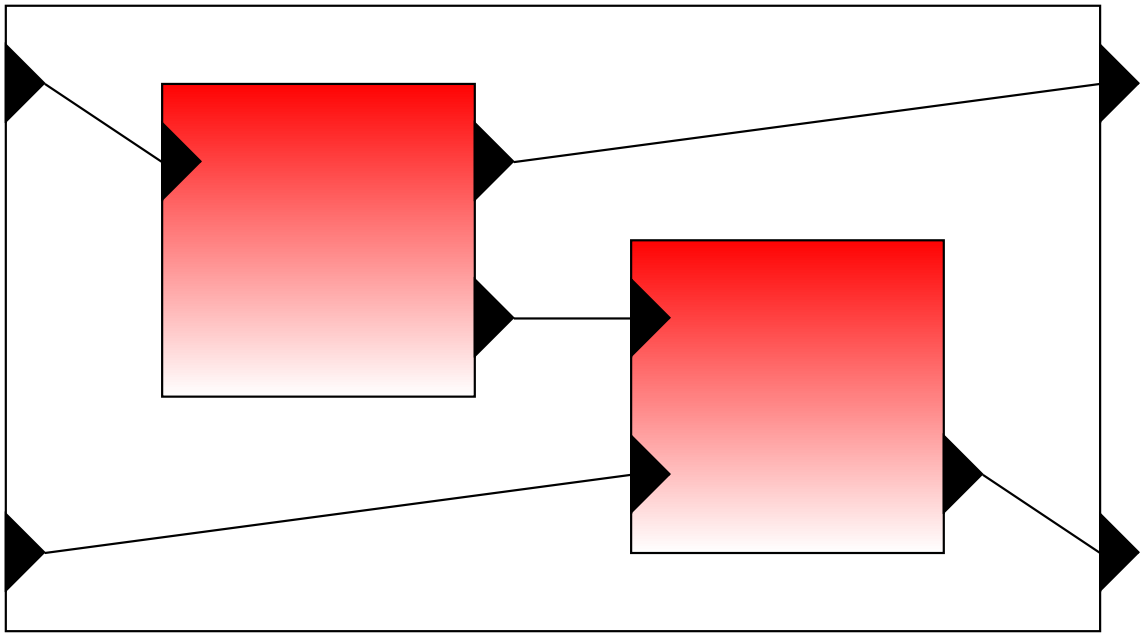


$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

Y l'ensemble des ports de sortie et des valeurs associées.

DEVS, Discrete Event System Specification

Description du modèle couplé

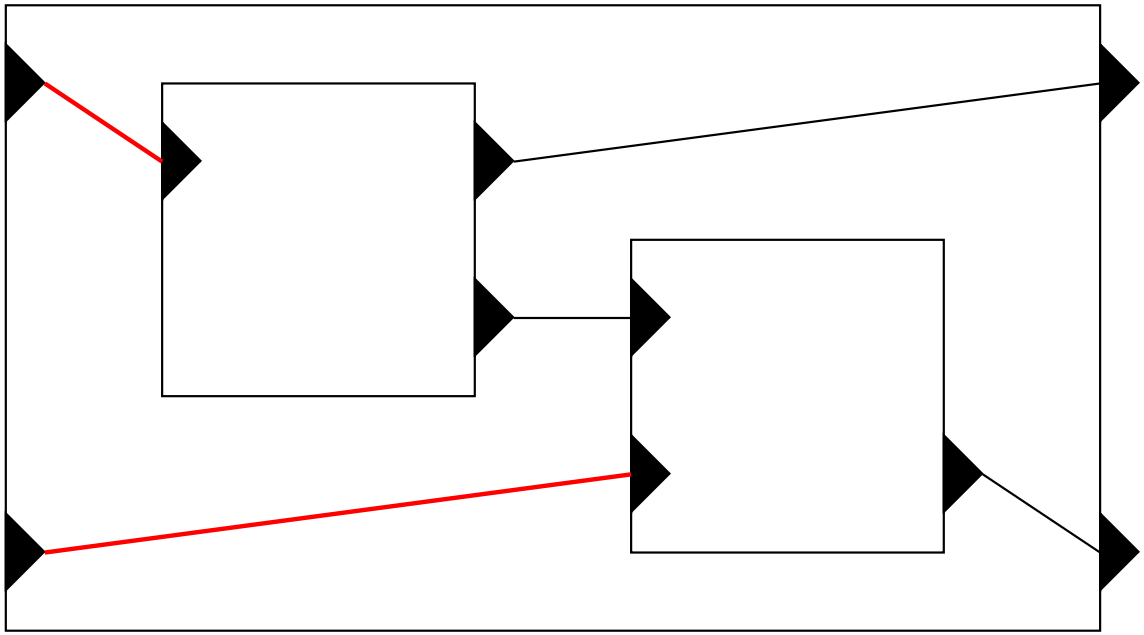


$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

D l'ensemble des identifiants des sous-modèles avec : $\{M_d | d \in D\}$.

DEVS, Discrete Event System Specification

Description du modèle couplé

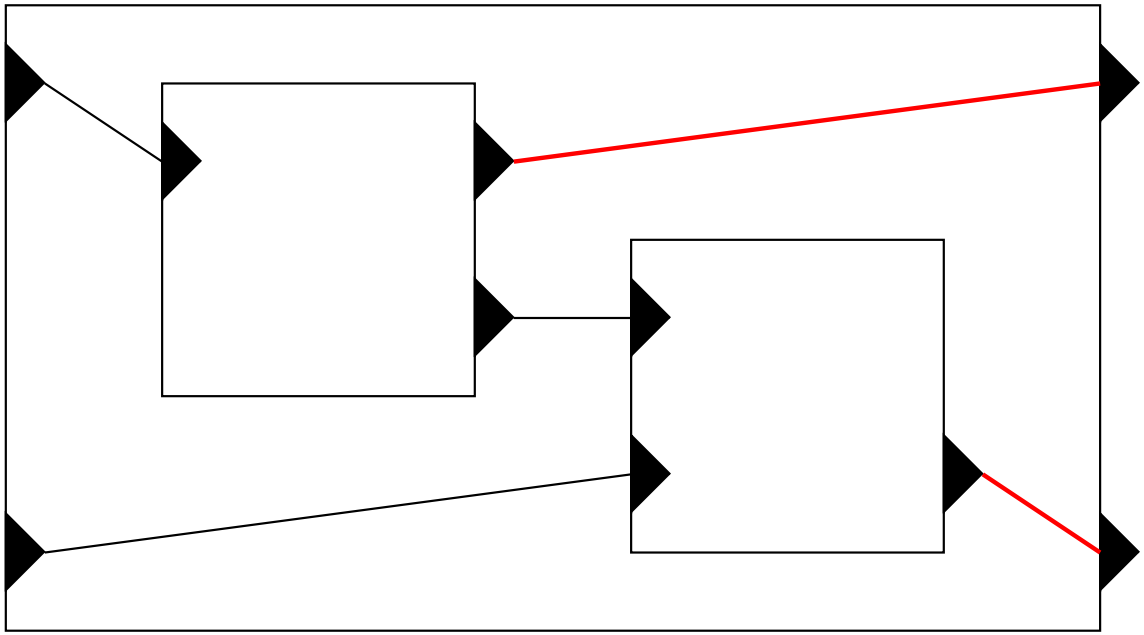


$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

EIC l'ensemble des connexions d'entrée.

DEVS, Discrete Event System Specification

Description du modèle couplé

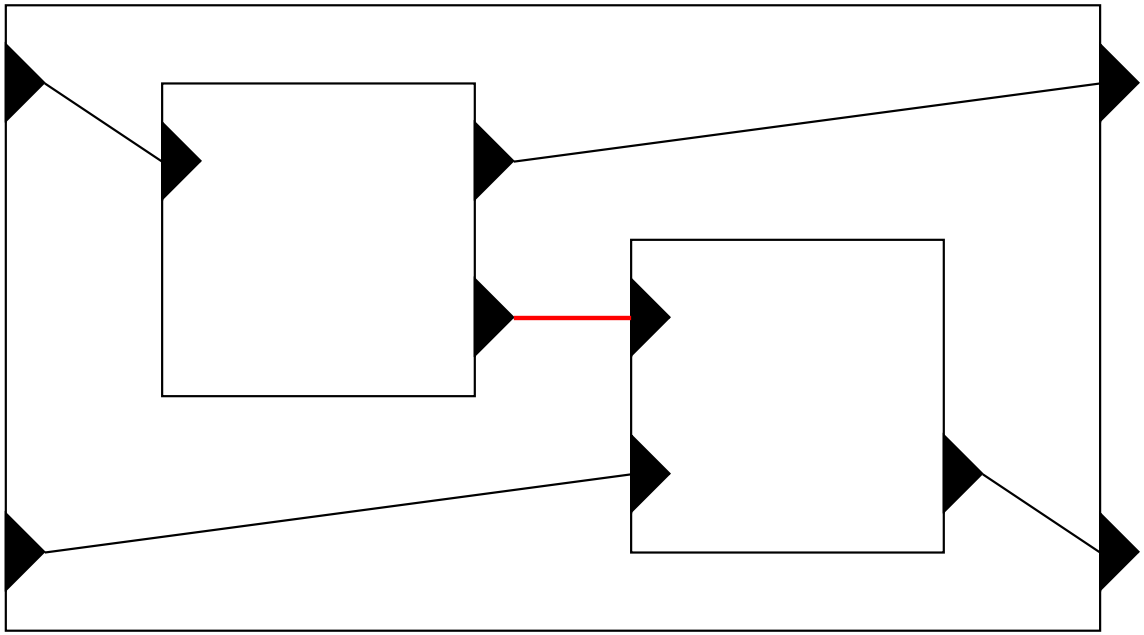


$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

EOC l'ensemble des connexions de sortie.

DEVS, Discrete Event System Specification

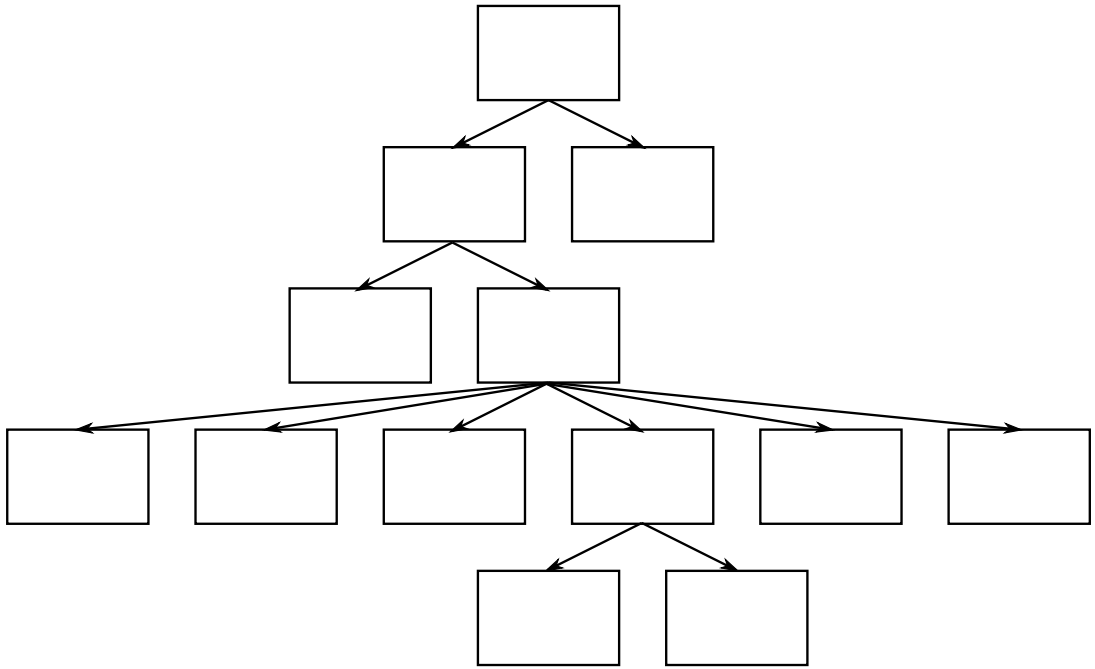
Description du modèle couplé



$$N = \langle X, Y, D, EIC, EOC, IC \rangle$$

IC l'ensemble des connexions internes.

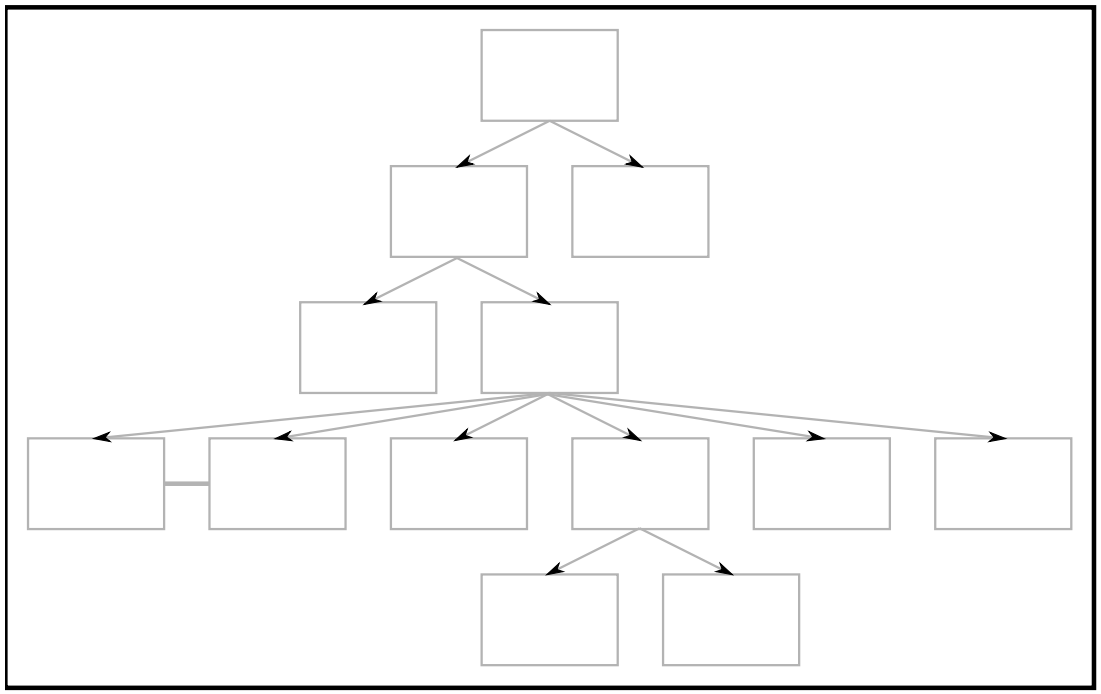
Composition hiérarchique de modèles



Propriété de fermeture sur couplage

Garantie qu'un modèle DEVS couplé est rigoureusement équivalent à un modèle DEVS atomique.

Composition hiérarchique de modèles



Propriété de fermeture sur couplage

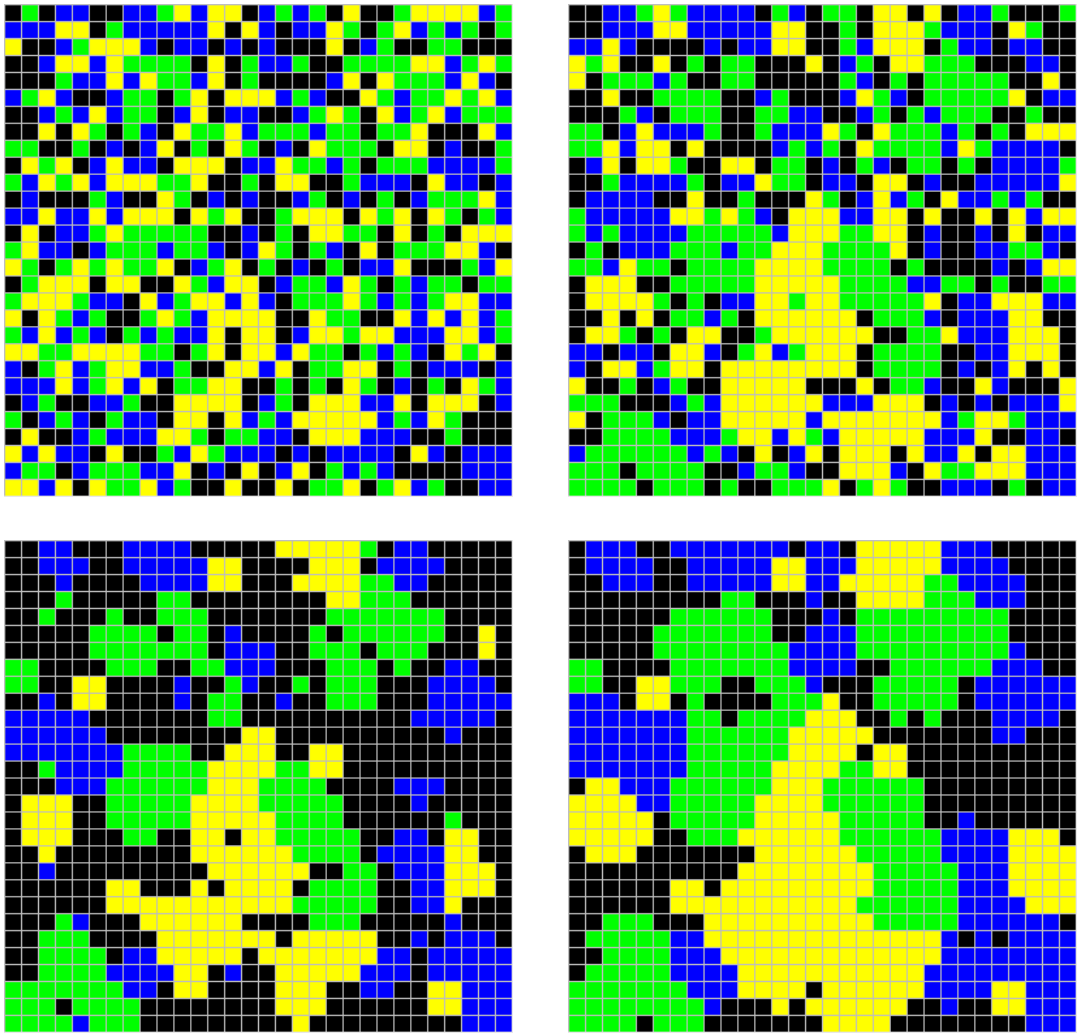
Garantie qu'un modèle DEVS couplé est rigoureusement équivalent à un modèle DEVS atomique.

DEVS « classique » possède un problème dans sa gestion des événements simultanés [Zeigler et al., 2000] (deux événements internes à la même date) :

→ **problème** : pas de priorité entre les événements stockés à la même date.

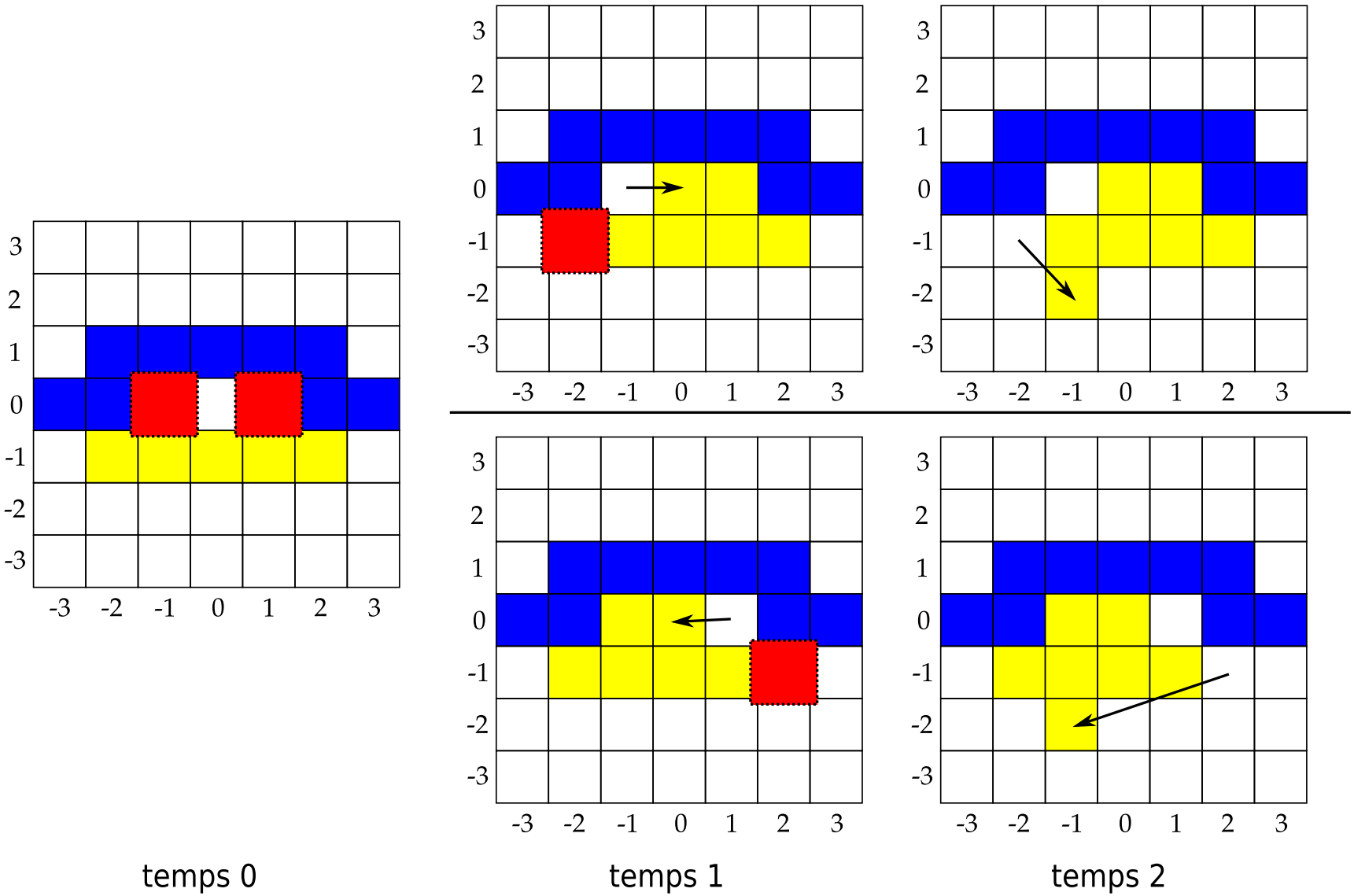
Les simulateurs DEVS

DEVS parallèle



Les simulateurs DEVS

DEVS Parallèle



DEVS Parallèle [Zeigler et al., 2000] :

- propose une spécification de parallélisation « simple »
  - → parallélise l'exécution des fonctions de transitions des modèles atomiques.
  - → utilise des sous-ensembles, ou sacs, d'événements levés à partir de la même date et du même sac

$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda, \delta_{con} \rangle$$

Où :

- $\delta_{ext} : Q \times X^b \rightarrow S$: fonction de transition interne avec X^b les événements.
- $\delta_{con} : S \times X^b \rightarrow S$: fonction de conflit entre δ_{int} et δ_{ext} entre : $\delta_{con}(s, x) = \delta_{ext}(\delta_{int}(s), 0, x))$ ou $\delta_{int}(\delta_{ext}(s, ta(s), x))$

DEVS Parallèle [Zeigler et al., 2000] :

- propose une spécification de parallélisation « simple »
  - $\rightarrow$ parallélise l'exécution des fonctions de transitions des modèles atomiques.
  - $\rightarrow$ utilise des sous-ensembles, ou sacs, d'événements levés à partir de la même date et du même sac

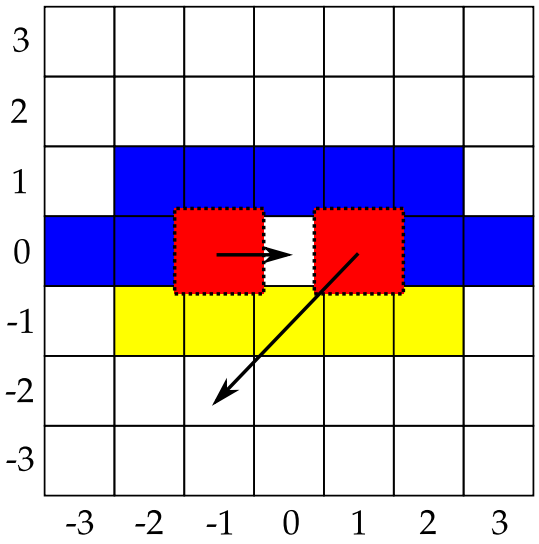
$$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, ta, \lambda, \delta_{con} \rangle$$

Où :

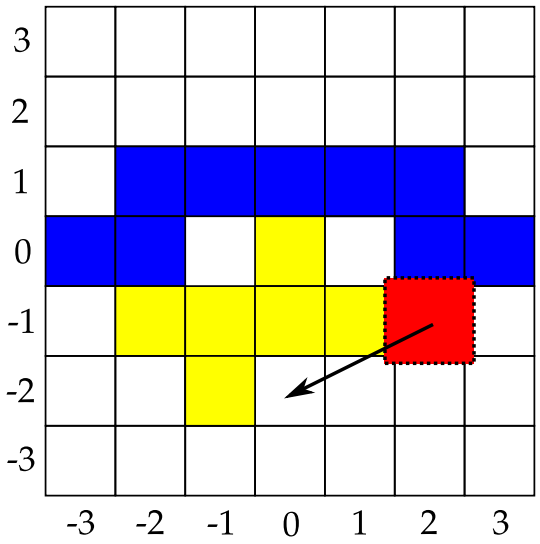
- $\delta_{ext} : Q \times X^b \rightarrow S$: fonction de transition interne avec X^b les événements.
- $\delta_{con} : S \times X^b \rightarrow S$: fonction de conflit entre δ_{int} et δ_{ext} entre : $\delta_{con}(s, x) = \delta_{ext}(\delta_{int}(s), 0, x))$ ou $\delta_{int}(\delta_{ext}(s, ta(s), x))$

Les simulateurs DEVS

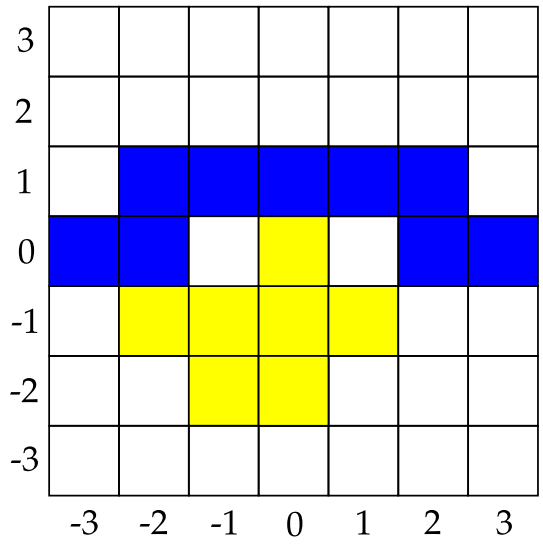
DEVS Parallèle



temps 0



temps 1



temps 2

- DEVS possède une structure statique : modèles couplés/atomiques
  - → offrir une structure dynamique permet de répondre à la diversité des modèles
- La spécification choisie : *Dynamic Structure DEVS* [Barros, 1995].
  - → ajoute un modèle atomique au réseau de modèles : **exécutif**
  - → le modèle exécutif possède le contrôle complet de la structure du modèle couplé :
    - ajouts et suppressions de modèles, de connexions et/ou de ports.

Les simulateurs DEVS

DEVS structures dynamiques

Réseau de modèles : déportent les connexions dans un modèle dit *exécutif* :

$$DSDEVN_N = \langle X_N, Y_N, \chi, M_\chi \rangle$$

où :

- X_N et Y_N sont les ports d'entrée et de sortie du réseau
- χ le nom du modèle *exécutif*
- M_χ le modèle *exécutif*

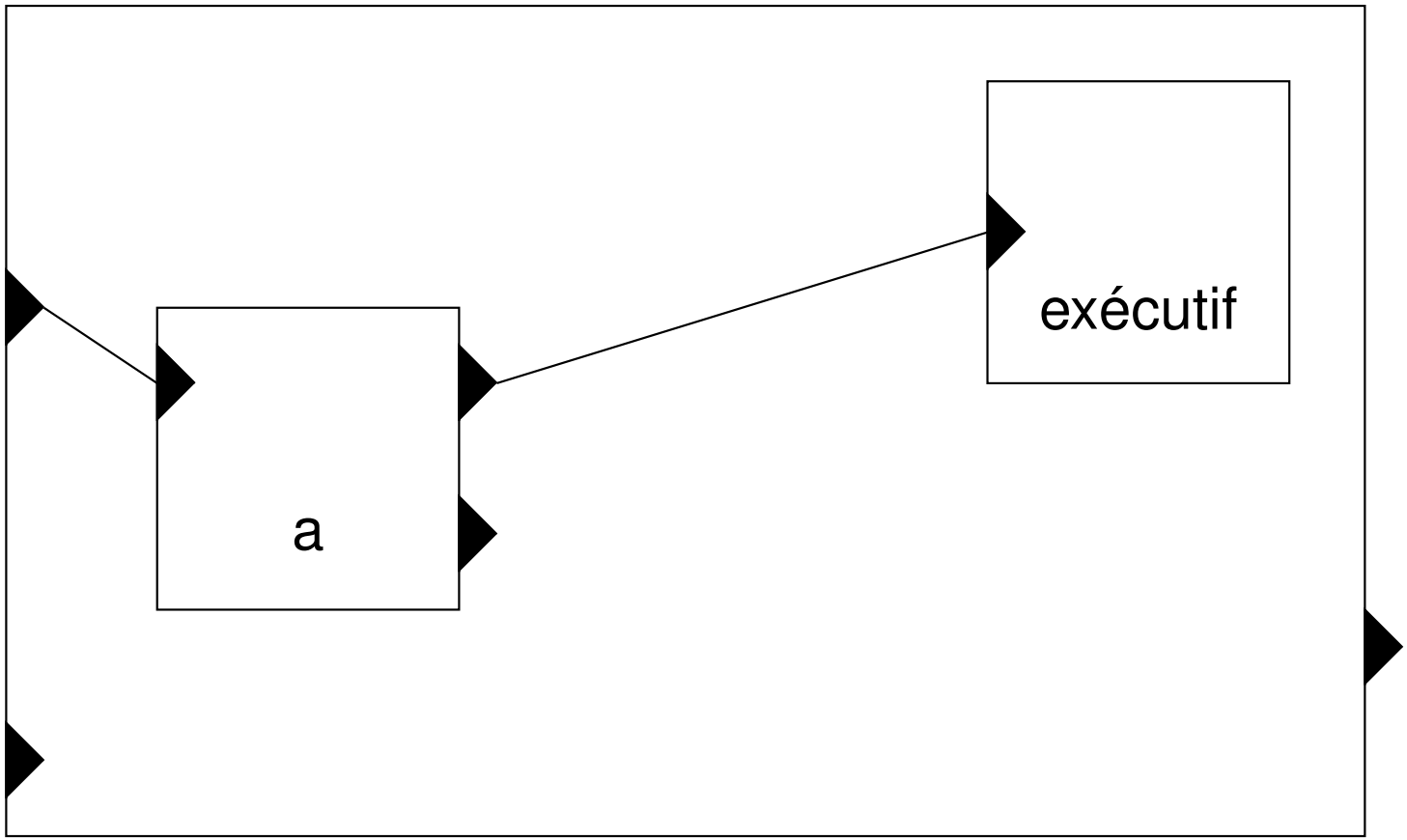
$$M_\chi = \langle X_\chi, Y_\chi, S_\chi, \delta^\chi_{int}, \delta^\chi_{ext}, ta_\chi, \lambda_\chi, \Sigma^*, \gamma \rangle$$

Avec :

$$\begin{cases} \gamma : S_\chi \rightarrow \Sigma^* \text{ la fonction de structure} \\ \Sigma* : \text{ l'ensemble des structures} \\ \Sigma_\alpha \in \Sigma^* : \\ \qquad \Sigma_\alpha = \langle D, EIC, EOC, IC \rangle \end{cases}$$

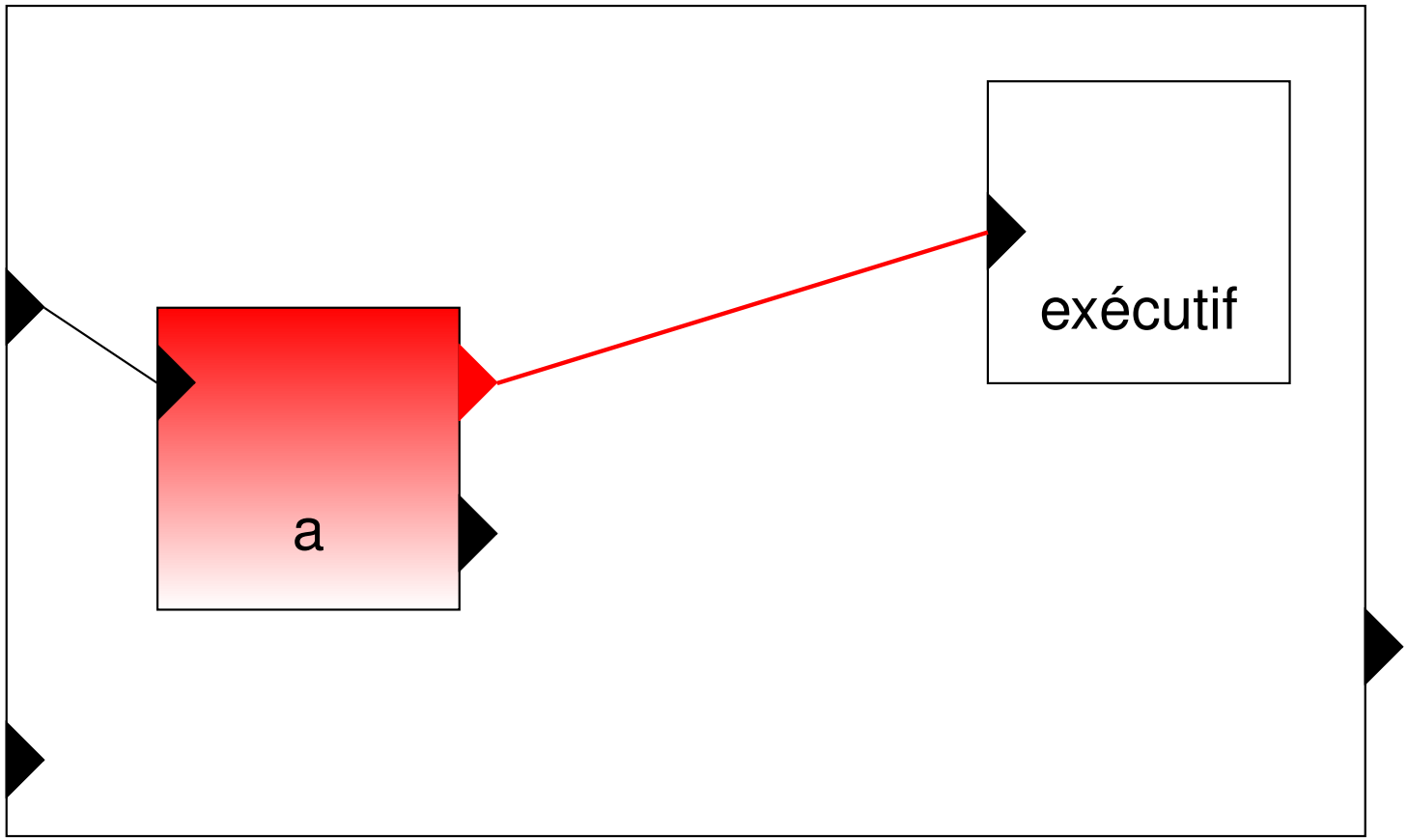
Les simulateurs DEVS

DEVS structures dynamiques



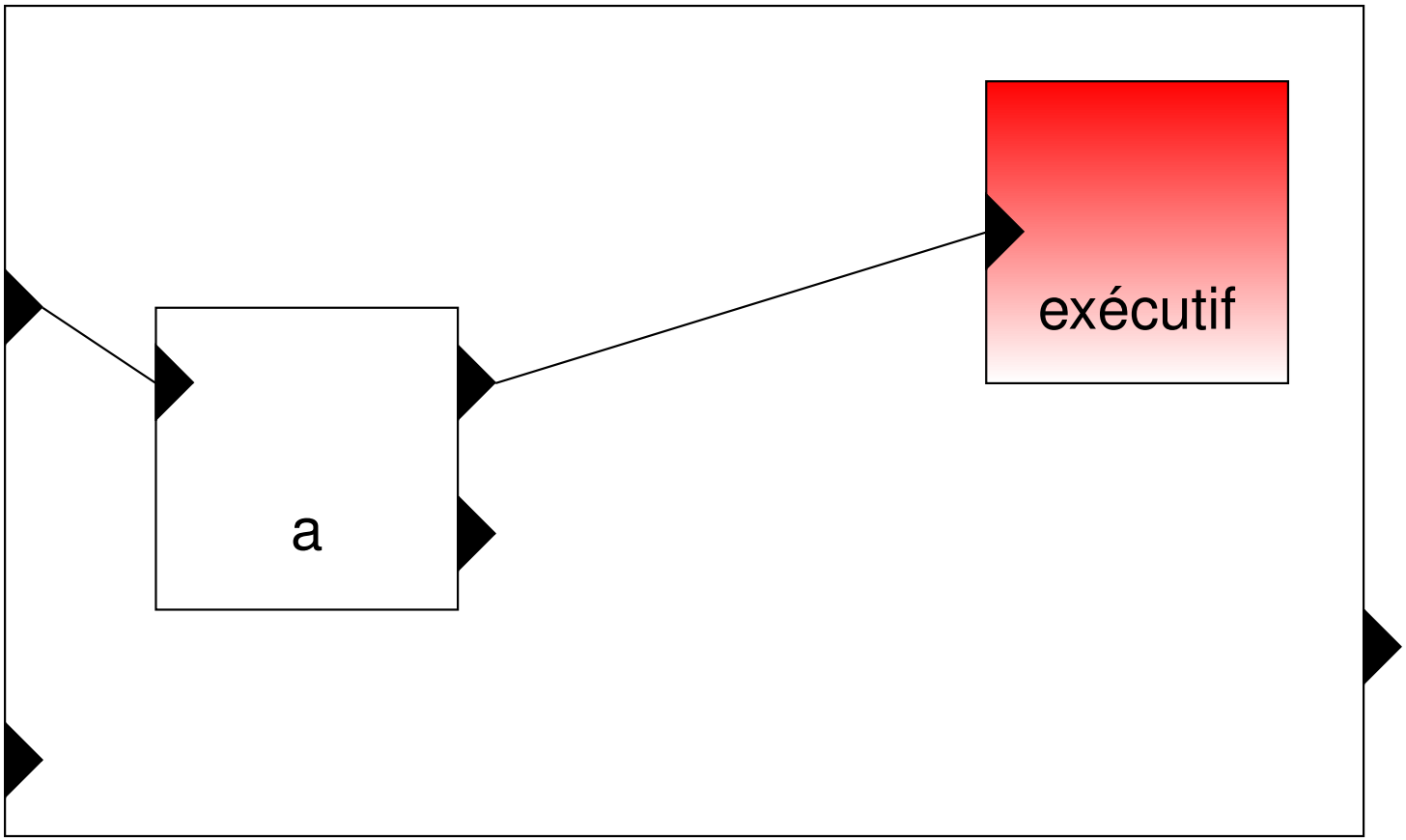
Les simulateurs DEVS

DEVS structures dynamiques



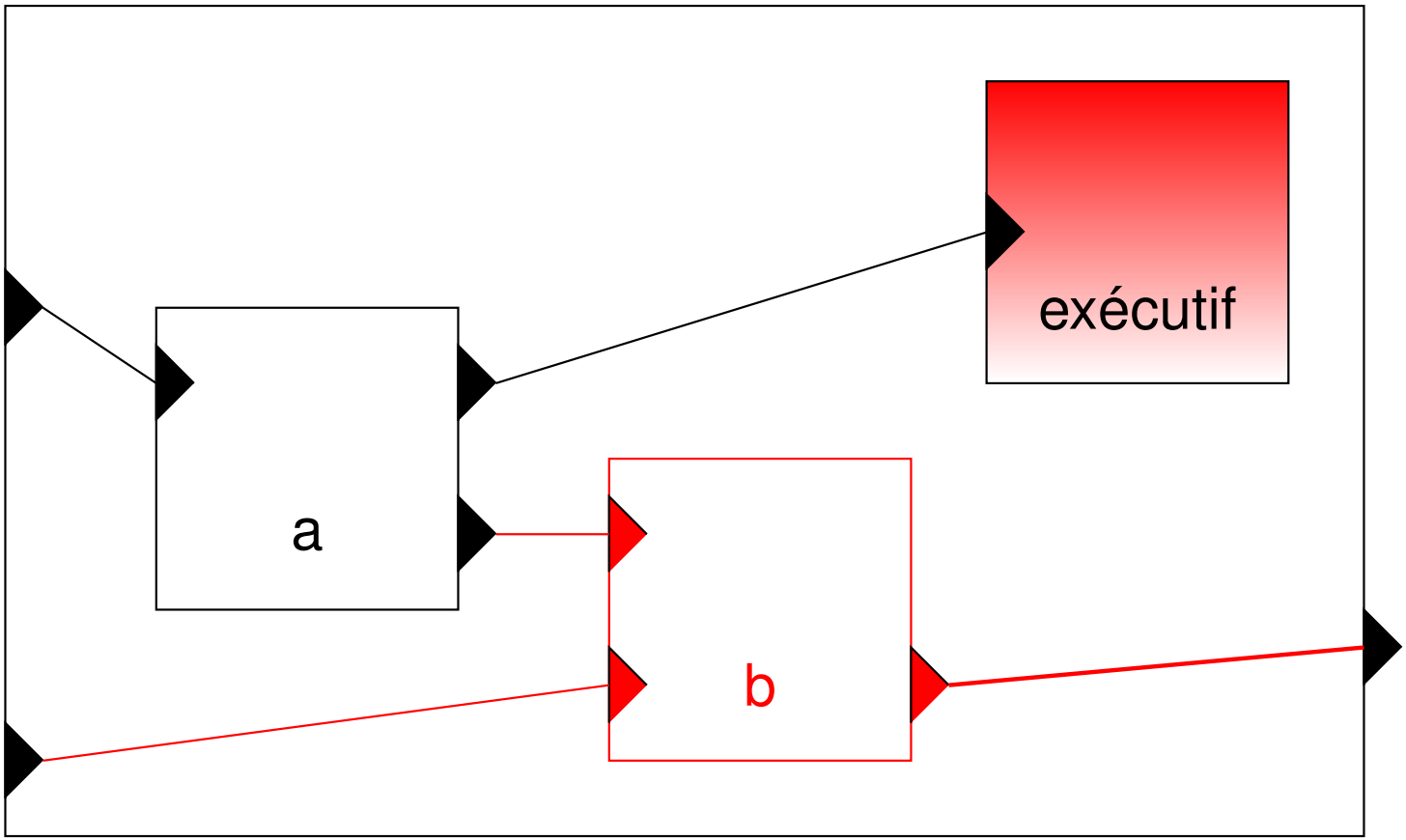
Les simulateurs DEVS

DEVS structures dynamiques



Les simulateurs DEVS

DEVS structures dynamiques



Les simulateurs DEVS

Et les autres...

DEVS Parallèle Parallélisation des simulateurs (méthode pessimiste)

DEVS TimeWarp Parallélisation des simulateurs (méthode optimiste)

DEVS temps réel Couplage DEVS avec des modèles temps réel

DS-DEVS Utilisation de graphes dynamiques

Fuzzy DEVS Intégrer des modèles en logique floue

etc.

Les extensions de DEVS

Les extensions de DEVS permettent d'étendre formellement la spécification DEVS : nouveaux types modèles.

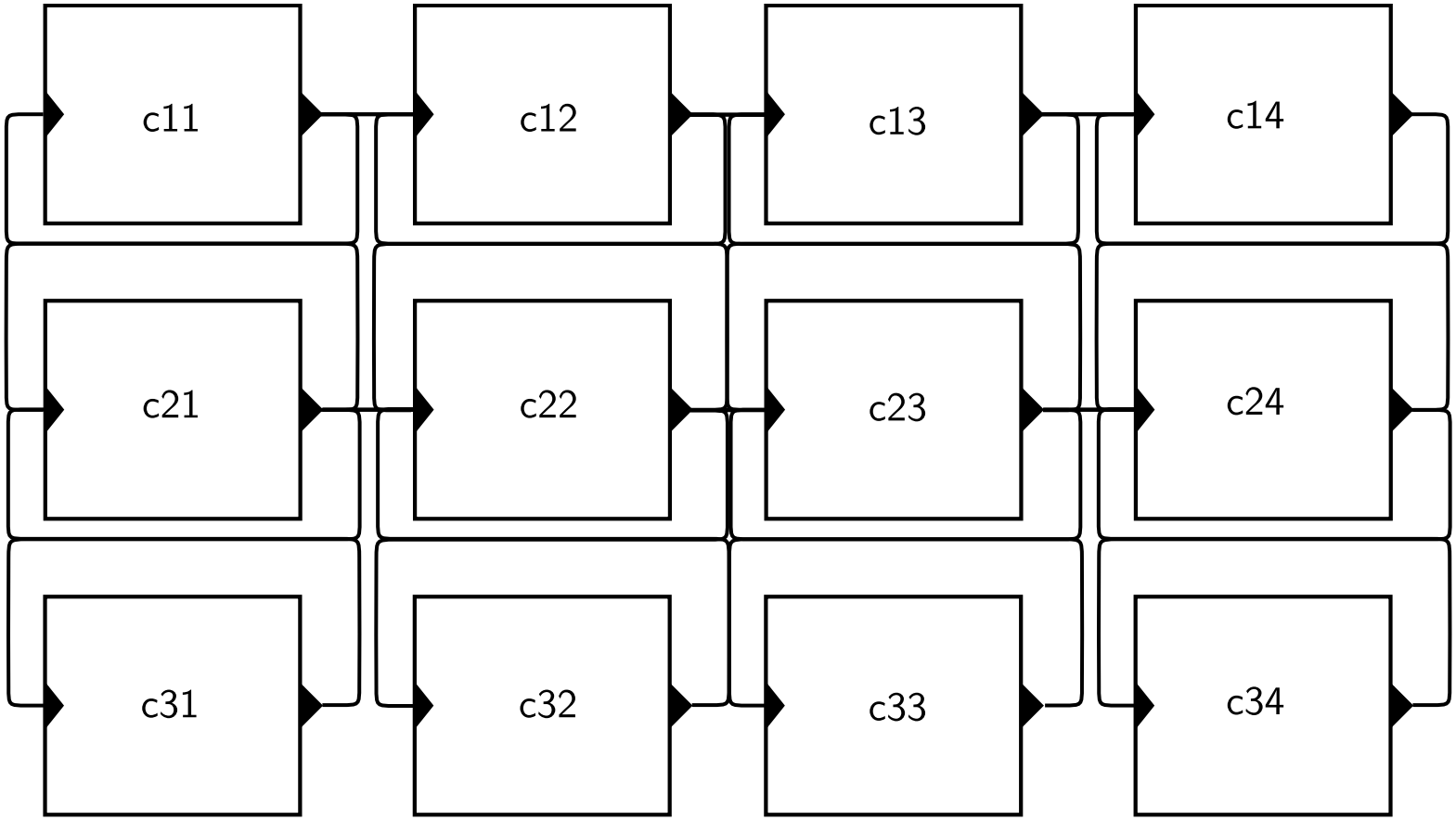
Quelques exemples pour les modèles :

Cell-DEVS

- l'état du modèle S contient son état et celui de ces voisins
- une diffusion instantanée λ ou avec délais δ_{int} des modifications ou perturbations aux voisins δ_{ext}
- la cellule est dupliquée pour former un automate cellulaire.

Les extensions de DEVS

Graphe de couplage d'un CellDevs

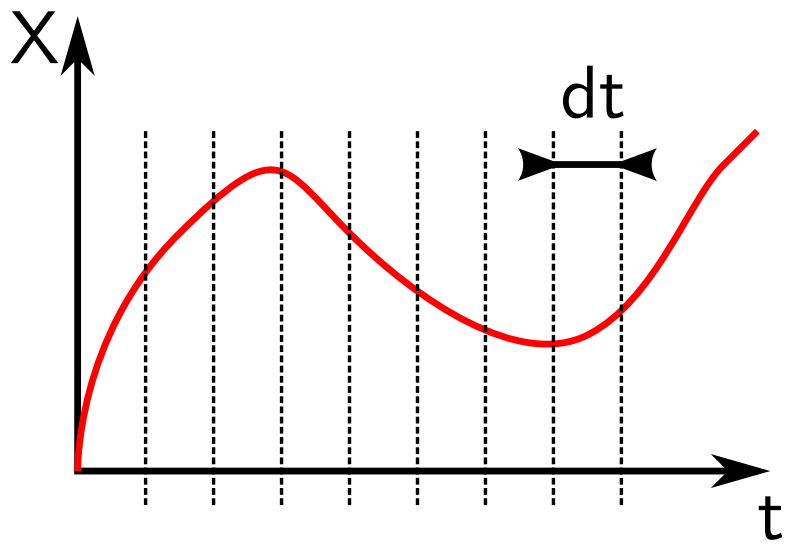


Les extensions de DEVS

QSS : moteur d'intégration d'équations différentielles

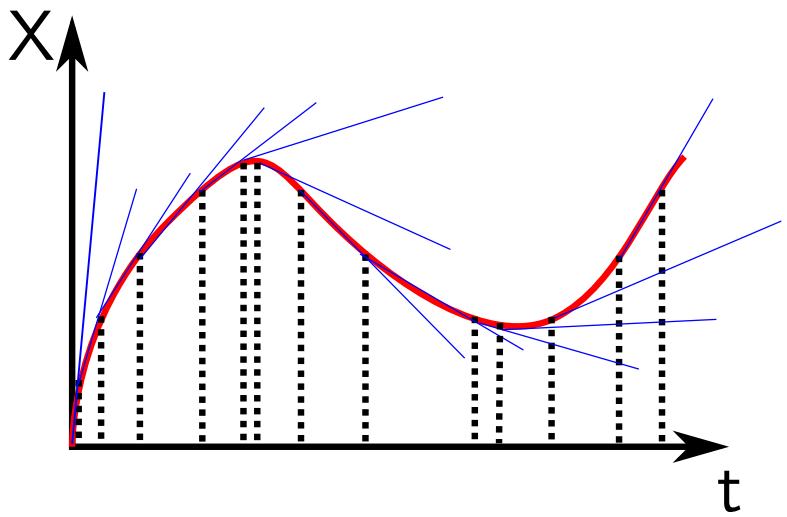
Comment résoudre les équations différentielles ordinaires :

- DESS [Zeigler et al., 2000] : discrétisation de l'espace du temps :
  - *ta* la constante du pas d'intégration
  - δ_{int} le moteur d'intégration (Euler, Runge Kutta, etc.).



Les extensions de DEVS

- QSS [Kofman and Junco, 2001] : discrétisation de l'espace des valeurs :
  - δ_{int} calcul de la pente (pour QSS1)
  - *ta* calcul de la date de dépassement du seuil δ_q.



Développement d'un système d'équations

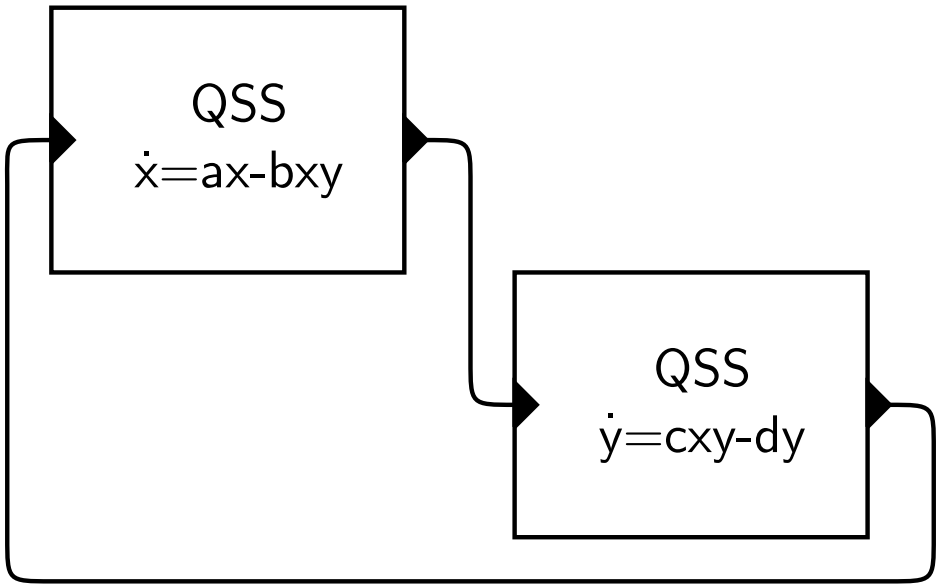
Exemple avec le modèle de Lotka-Voltera

DEVS s'appuie, entre autre, sur deux propriétés très importantes :

- le couplage de modèles : un modèle atomique peut être combiné à un autre modèle pour former le couplage de modèle.
- la fermeture sous couplage : un modèle couplé possède les mêmes propriétés qu'un modèle atomique.

Développement d'un système d'équation

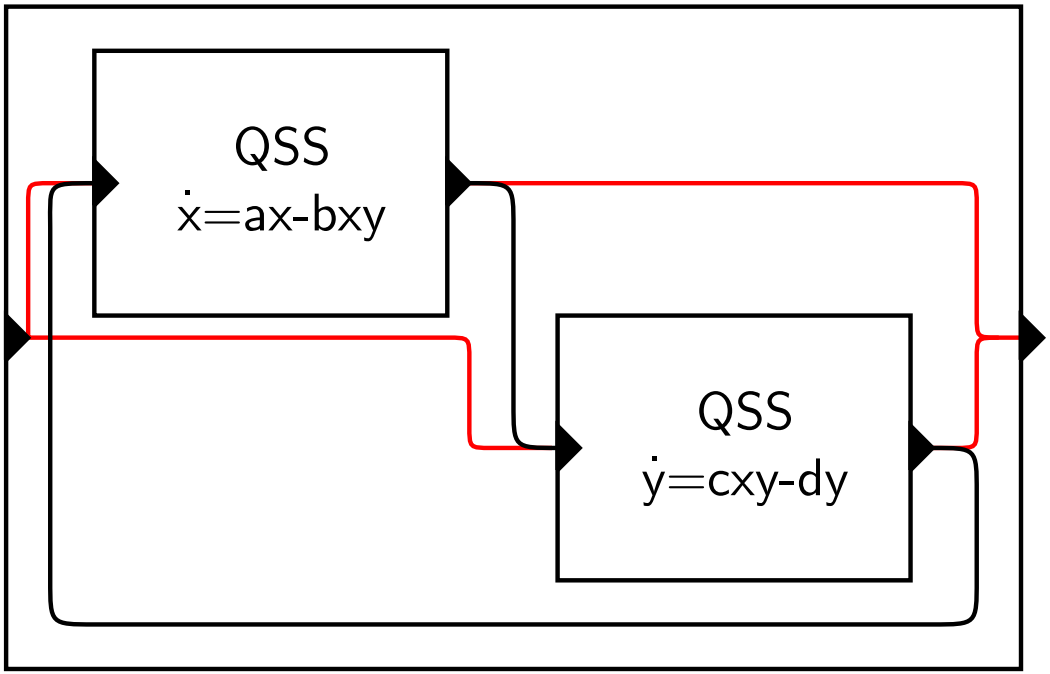
Exemple avec le modèle de Lotka-Voltera couplé à un modèle de décision



Deux modèles QSS avec une équation chacun : construction d'un système d'équations par le couplage.

Développement d'un système d'équation

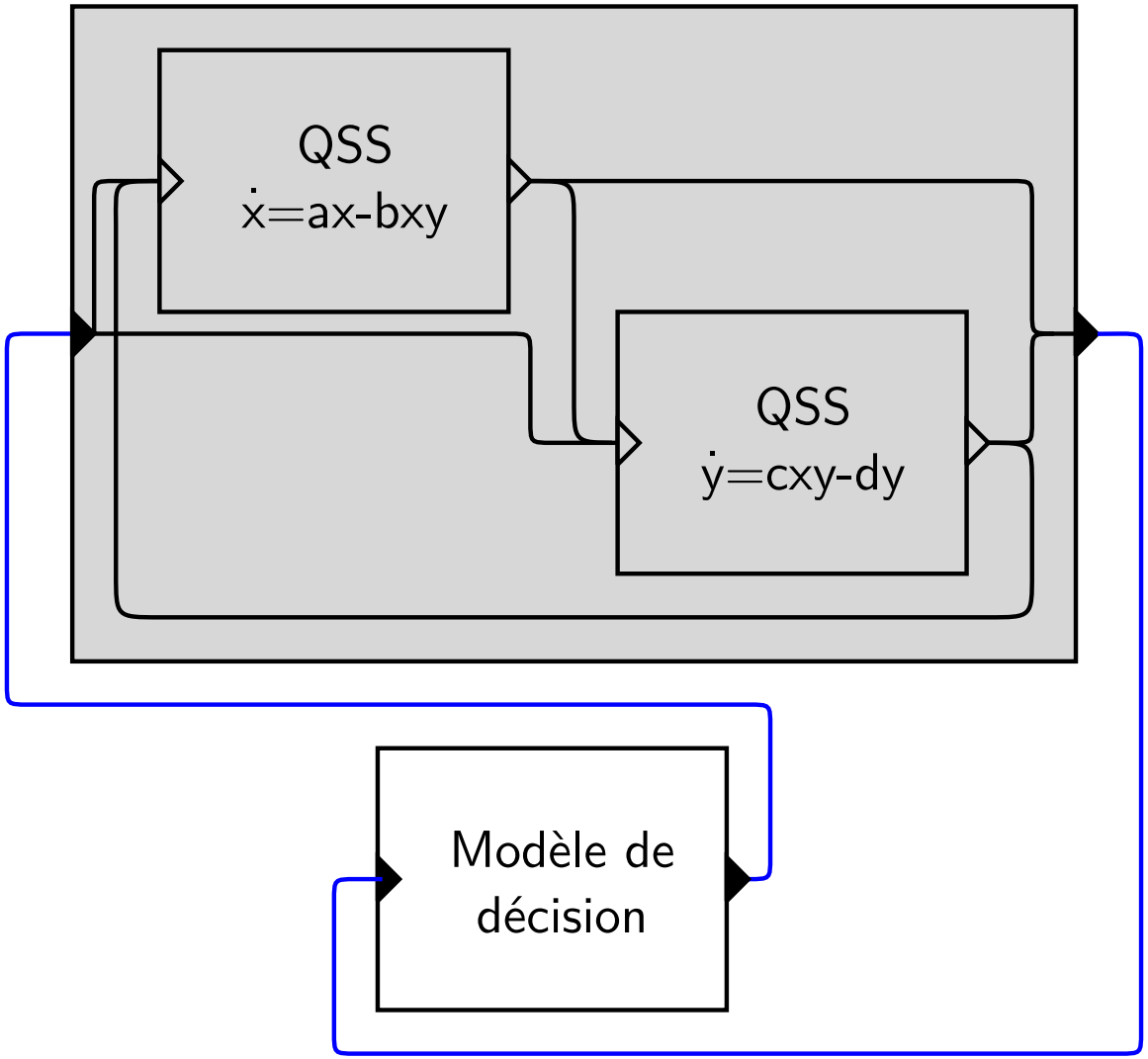
Exemple avec le modèle de Lotka-Voltera couplé à un modèle de décision



Le couplage peut former un nouveau modèle dont seuls les ports d'entrée et de sortie permettent la communication.

Développement d'un système d'équation

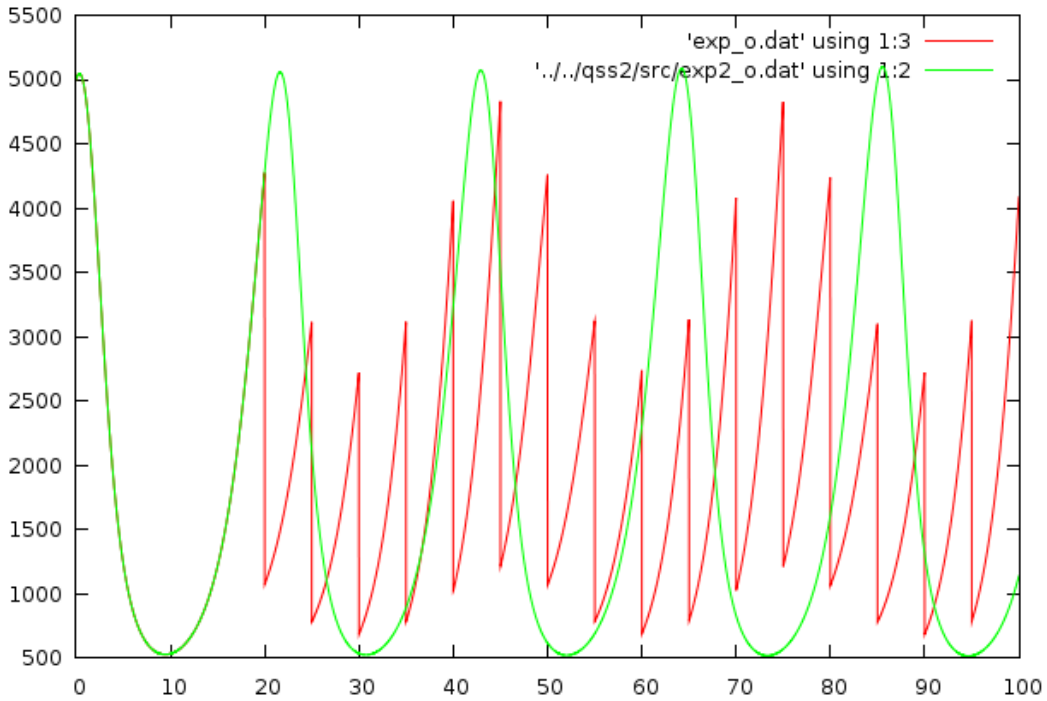
Exemple avec le modèle de Lotka-Voltera couplé à un modèle de décision



Un modèle de décision peut venir se coupler sur ce modèle.

Développement d'un système d'équation

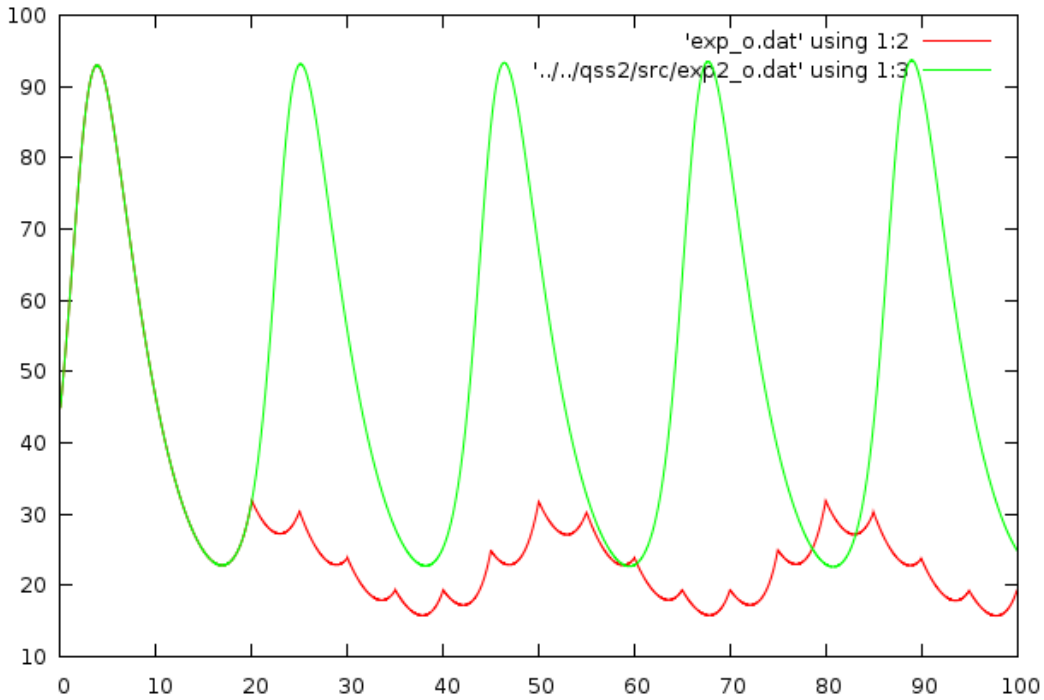
Exemple avec le modèle de Lotka-Volterra couplé à un modèle de décision



1. Vert, variable X (proies) sans perturbation.
2. Rouge, variable X avec une perturbation commençant quand les proies dépassent le seuil de 2000 puis toutes les 5 ut. (-75% de X).

Développement d'un système d'équation

Exemple avec le modèle de Lotka-Volterra couplé à un modèle de décision



1. Vert, variable Y sans perturbation.
2. Rouge, variable Y lorsque la variable X répercute ses modifications.

Les extensions de DEVS

Qss1, Qss2 et Qss3 Moteur d'intégration d'équations différentielles

Cell-Qss Moteur d'intégration d'équations différentielles spatialisées

Réseau de Petri Traduction (*mapping*) des RdP en DEVS

Automate à état finis Traduction des FSA en DEVS

etc.

GSMP et GSMDP Processus de Markov généralisé, en cours par Emmanuel Rachelson

Plan

Description globale

Simulateur

- une implémentation des simulateurs abstraits de B. P. Zeigler et F. J. Barros : DSDE : DEVS parallèle et DS-DEVS
- des extensions DEVS : Cell-DEVS, QSS1, QSS2, Cell-Qss, Réseau de Petri, Automate à états finis, DESS, ...
- une licence libre : GPL

Technique

- fonctionnement en bus logiciel : les modèles se connectent sur le bus pour communiquer
- développé en C++ via une approche par composants $\rightarrow$ développement aisée de couche au dessus de VLE (R-VLE)
- l'ensemble de la plate-forme : 100 000 lignes
- noyau du simulateur : quelques centaines de lignes

Nos apports à DEVS

L'ensemble de nos apports à DEVS :

→ Séparer du comportement des modèles, les fonctionnalités communes :

1. initialisation des modèles
2. observations des modèles
3. les interrogations entre les modèles.

→ Simplifier le comportement et le développement des modèles.

Nos apports à DEVS

Initialisation des modèles atomiques

- Initialisation
  - Initialiser est la première partie de la simulation
  - Consiste à appliquer une valeur à un état du système
- Plans d'expériences
  - Attribuer plusieurs valeurs à un état du système afin d'effectuer plusieurs simulations
- Problème
  - pas de formalisation du paramétrage des modèles

$\rightarrow$ Développement d'un fonction d'initialisation δ_{init} dans la spécification DEVS.

Nos apports à DEVS

Initialisation des modèles atomiques

- Initialisation
  - Initialiser est la première partie de la simulation
  - Consiste à appliquer une valeur à un état du système
- Plans d'expériences
  - Attribuer plusieurs valeurs à un état du système afin d'effectuer plusieurs simulations
- Problème
  - pas de formalisation du paramétrage des modèles

$\longrightarrow$ Développement d'un fonction d'initialisation δ_{init} dans la spécification DEVS.

DEVS et les initialisations :

Solution : $M = \langle X, Y, X_{init}, S, \delta_{ext}, \delta_{init}, \delta_{int}, \lambda, ta \rangle$

- fonction d'initialisation $\delta_{init} : S \times X_{init} \rightarrow S'$
- X_{init} l'ensemble des ports d'initialisation et des valeurs associées
- initialise une ou plusieurs variables d'états du modèle atomique
- permet de paramétrer les modèles en cours de construction par DS-DEVS.

Nos apports à DEVS

Observations des modèles atomiques

Observer ?

- Étudier un système implique son observation
- Observer un modèle est l'action de récupérer la valeur d'un état du modèle
- Observation du système entraîne la modification de l'état du modèle : Principe d'incertitude de Karl Werner Heisenberg, 1927.

→ Développement d'une fonction d'observation δ_{obs} dans la spécification DEVS.

- Garantir la **non modification** de l'état du modèle observé afin de réaliser des simulations reproductibles et non dépendantes des observations
- Découper la notion d'observation du comportement du modèle.

Nos apports à DEVS

Observations des modèles atomiques

Observer ?

- Étudier un système implique son observation
- Observer un modèle est l'action de récupérer la valeur d'un état du modèle
- Observation du système entraîne la modification de l'état du modèle : Principe d'incertitude de Karl Werner Heisenberg, 1927.

→ Développement d'une fonction d'observation δ_{obs} dans la spécification DEVS.

- Garantir la **non modification** de l'état du modèle observé afin de réaliser des simulations reproductibles et non dépendantes des observations
- Découper la notion d'observation du comportement du modèle.

DEVS et les observations :

Problème :

- pas de fonction d'observation incluse dans DEVS :
  - laissée au modélisateur : sortie directe dans le code des modèles ou utilisation de λ

Solution : $M = \langle X, Y, X_{init}, Y_{obs}, S, \delta_{ext}, \delta_{init}, \delta_{int}, \delta_{obs}, \lambda, \lambda_{obs}, ta \rangle$

- $\delta_{obs} : S \rightarrow S$ et $\lambda_{obs} : S \rightarrow Y_{obs}$
- deux modes de captures formalisés :
  - événementiel : à chaque changement d'état du modèle un événement d'état est créé par λ_{obs}
  - planifié : chaque pas de temps génère un événement d'état λ_{obs}.

DEVS et la complexité des modèles :

Problème :

- les relations de type questions–réponses sont difficiles à modéliser :
  - force un changement d'état à la réception d'un événement externe
  - change d'état pour réaliser un appel à λ
  - gérer les questions entre ces changements.

Entraîne la création d'une multitude d'états temporaires.

→ Développement d'une fonction δ_{inst} pour la gestion des interrogations instantanées de modèles.

DEVS et la complexité des modèles :

Problème :

- les relations de type questions–réponses sont difficiles à modéliser :
  - force un changement d'état à la réception d'un événement externe
  - change d'état pour réaliser un appel à λ
  - gérer les questions entre ces changements.

Entraîne la création d'une multitude d'états temporaires.

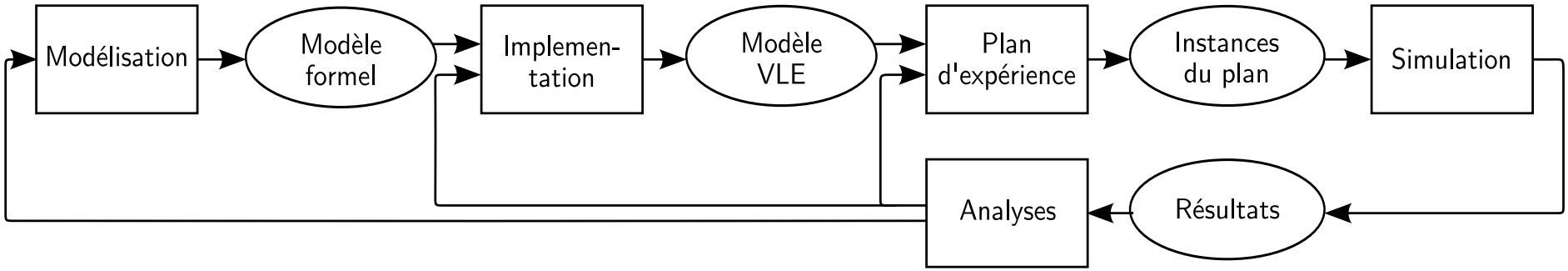
$\rightarrow$ Développement d'une fonction δ_{inst} pour la gestion des interrogations instantanées de modèles.

Solution : les événements instantanés

$$M = \langle X, Y, X_{init}, Y_{obs}, S, \delta_{ext}, \delta_{init}, \delta_{int}, \delta_{obs}, \delta_{inst}, \lambda, \lambda_{obs}, ta \rangle$$

- $\delta_{inst} : Q \times X \rightarrow Y$ où $Q = \{(s, e)\}$
- Ne modifie pas l'état du modèle interrogé

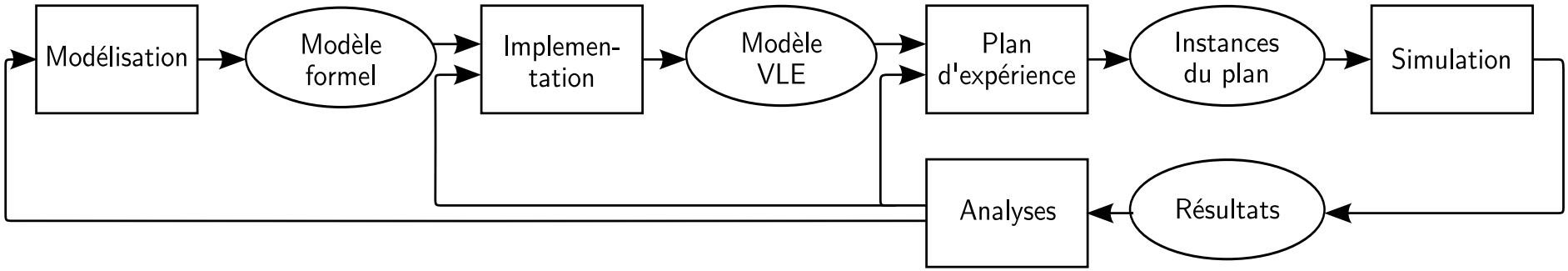
Description environnement VLE



Comment partager ?

- modèle formel : DEVS (δ_{int}, δ_{ext}, ta, λ, ...), voir dans ses extensions : équations différentielles, équations aux différences, automates, réseaux de Petri etc.
- modèle VLE : implémentation du modèle formel dans les API de VLE.

Description environnement VLE



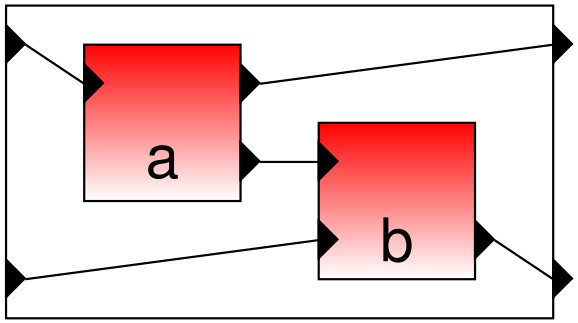
Comment partager ?

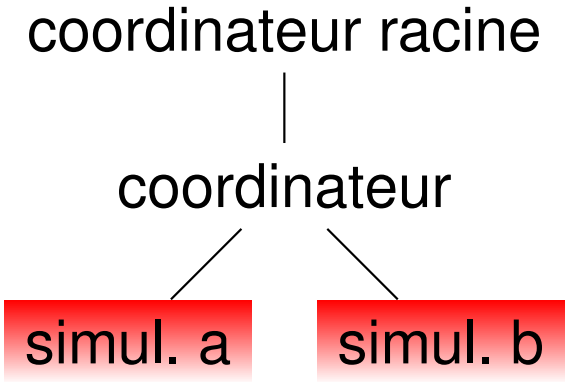
- modèle formel : DEVS (δ_{int}, δ_{ext}, *ta*, λ, ...), voir dans ses extensions : équations différentielles, équations aux différences, automates, réseaux de Petri etc.
- modèle VLE : implémentation du modèle formel dans les API de VLE.

B. P. Zeigler propose des algorithmes pour le formalisme DEVS [Zeigler et al., 2000] les simulateurs abstraits :

- à chaque modèle atomique est attaché un simulateur
  - → **simulateur** pilote l'appel des fonctions du modèle atomique.
- à chaque modèle couplé est attaché un coordinateur
  - → **coordinateur** achemine les événements entre modèles.
- un coordinateur racine gère la simulation.
  - → **coordinateur racine** traite les événements internes et l'échéancier.

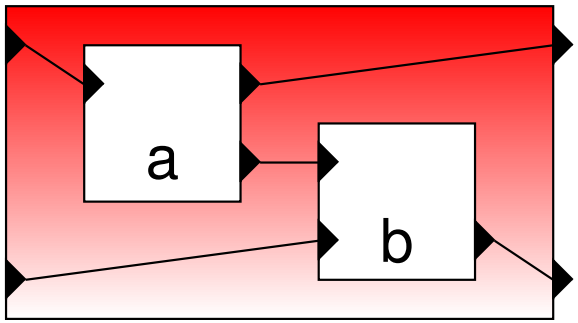
Représentation de la correspondance graphe – simulateur :

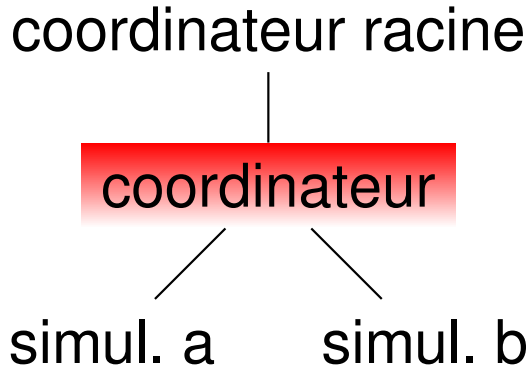




- les événements internes sont envoyés au coordinateur racine
- les événements externes sont envoyés aux coordinateurs

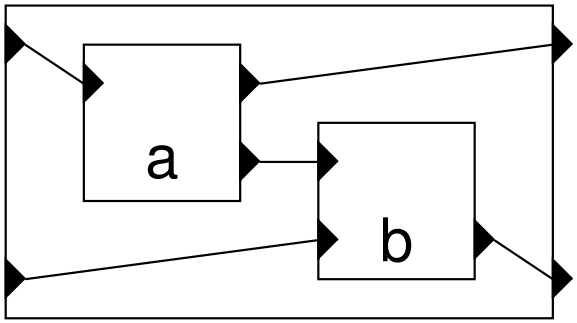
Représentation de la correspondance graphe – simulateur :

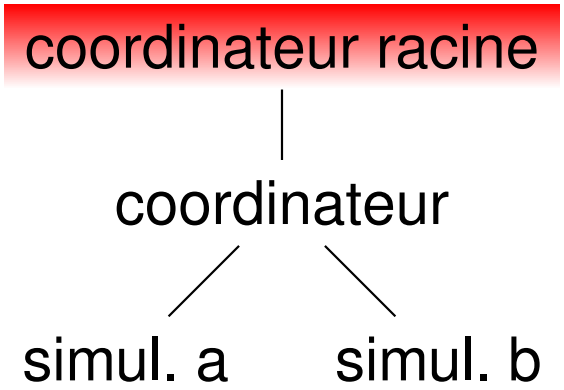




- le coordinateur gère les événements externes, si ceux-ci sortent du modèle couplé, ils sont envoyés au coordinateur parent.

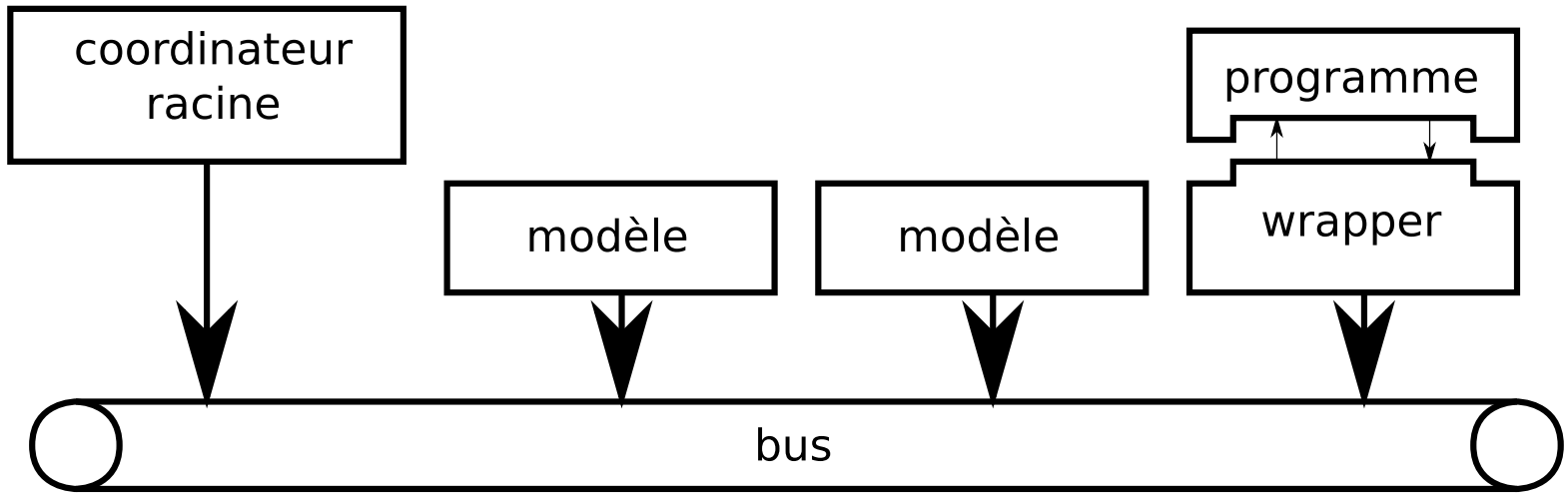
Représentation de la correspondance graphe – simulateur :





- gère tous les événements internes des simulateurs pour déterminer le simulateur actif.

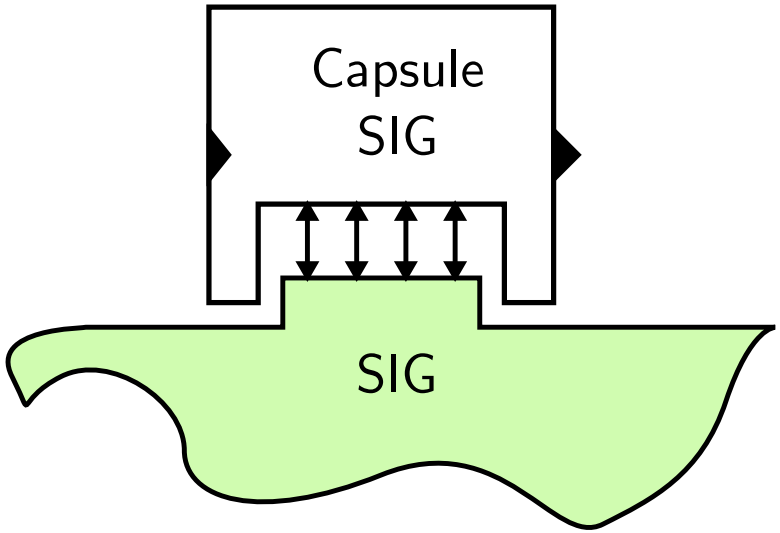
VLE : implémentation du DEVS-Bus (issus des travaux de Kim [Kim and Kim, 1996]) :



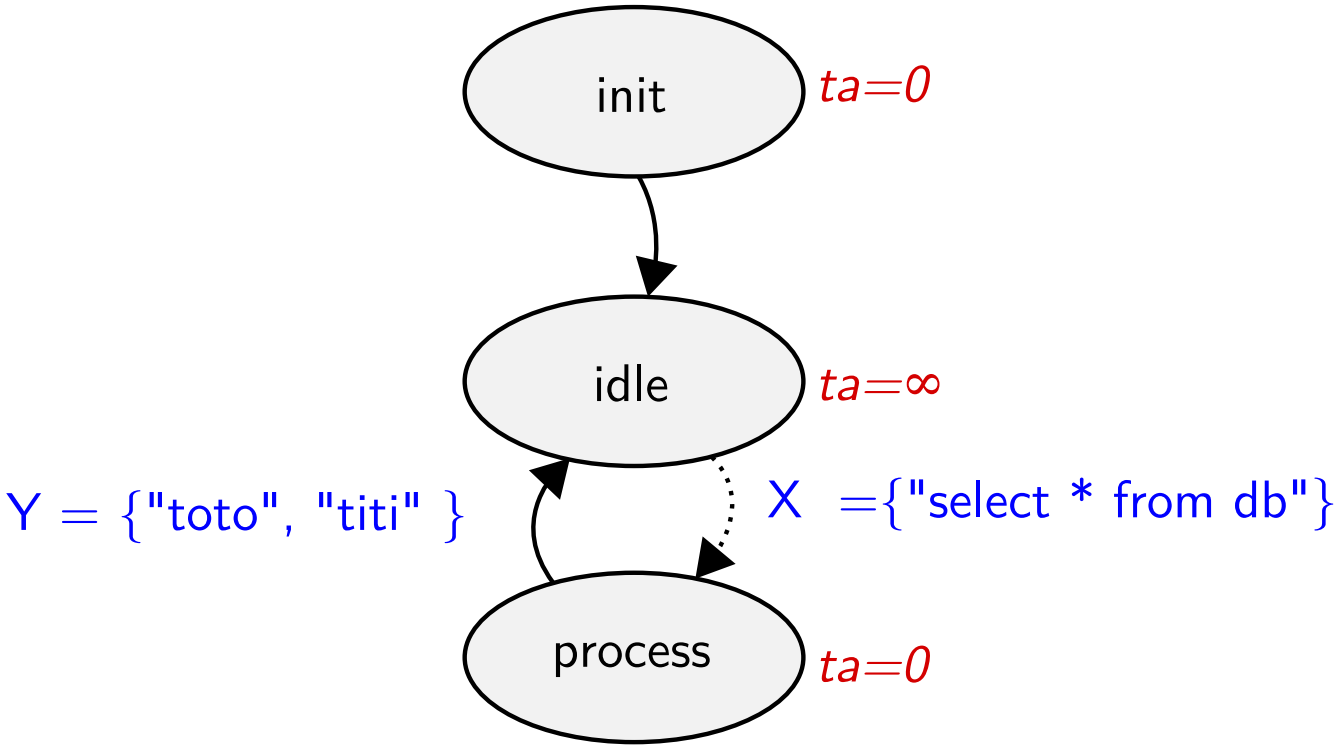
Réutilisation de modèles ou programmes existants

Les capsules ou *wrappers* proviennent des travaux sur le DEVS-Bus [Kim and Kim, 1998] dont les propriétés sont :

- d'être des modèles DEVS classiques
- de faire le lien entre un formalisme ou un programme existant et DEVS (Problème complexe, traduction des notions de temps, d'états, de transitions [Quesnel et al., 2004]).
- l'état d'un *wrapper* est l'union de son état et du composant encapsulé



Réutilisation de modèles ou programmes existants



Graphe d'états d'un *wrapper* autour d'une base de données SQL.

Traducteur

Le concept de traducteur provient des travaux de transformation de modèles :

- difficulté de gérer de très grands modèles type automates cellulaires
- besoin de définir le graphe de modèles et/ou les modèles par des expressions plus simples
- monter dans les **niveaux sémantiques** et proposer une **ontologie** proche des modélisateurs.

  → But : cacher DEVS, graphe de modèles et codes des modèles via la génération de code.

Traducteur

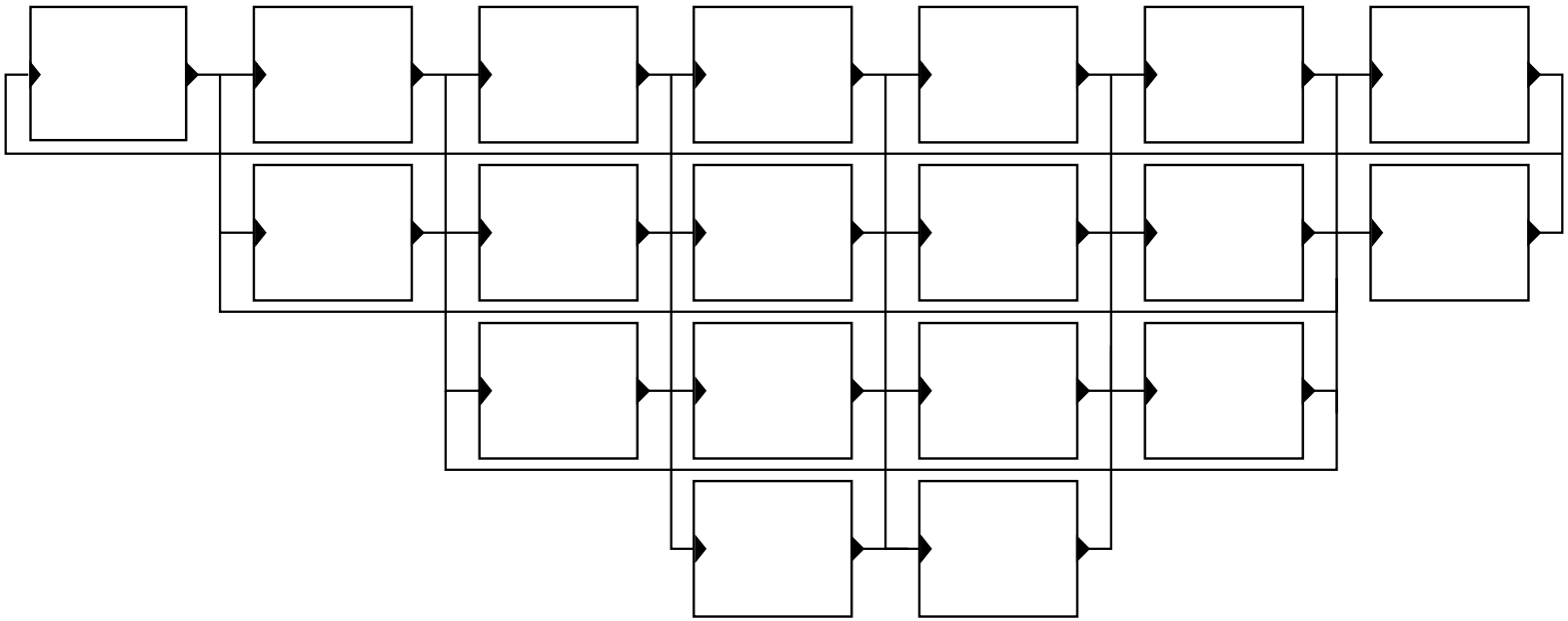
Ex. Générateur de matrices de modèles DEVS

Traducteur de grille

```
<celldevs>
<grid>
<dim axe="0" number="11" />
<dim axe="1" number="7" />
</grid>
<cells connectivity="neuman|moore" library="libcellule" >
<prefix>cell</prefix>
<init>
0 0 0 1 1 1 1 1 1 1 0
0 0 0 0 1 1 1 1 1 1 0
0 0 0 0 0 1 1 1 1 0 0
0 0 0 0 0 0 1 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0
</init>
</cells>
</celldevs>
```

Traducteur

Ex. Générateur de matrices de modèles DEVS



Génération d'un graphe de modèles à partir d'une ontologie différentes de DEVS.

Les utilisations de VLE

ANR CHALOUPE CHangement gLObal, dynamiqUe de la biodiversité marine exploitée et viabilité des PEcheries – Tâche 4, modélisation. Couplage de modèles de résolution d'équations récurrentes.

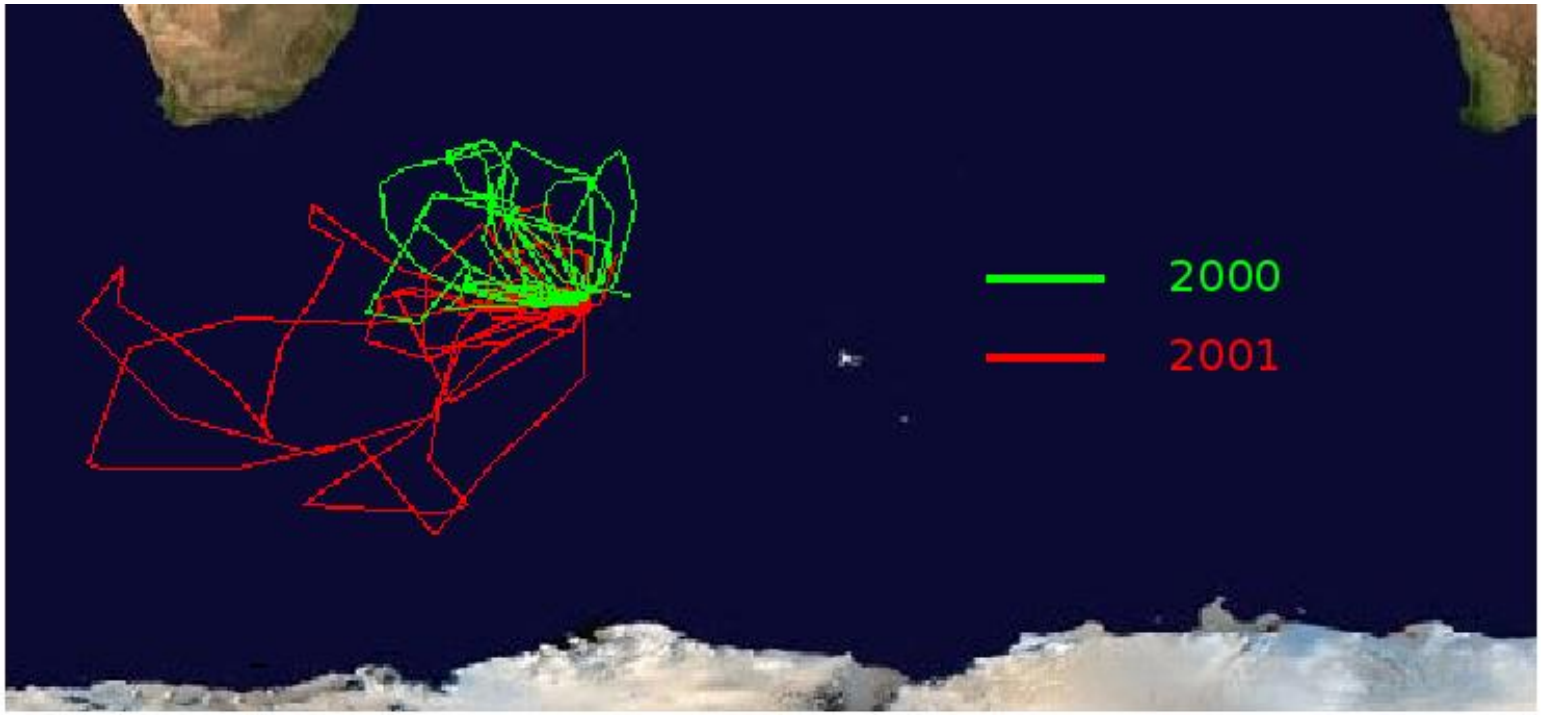
ANR REMIGE Réponses comportementales et démographiques des prédateurs marins de l'Océan Indien aux changements globaux – R. Duboz. Agent-DEVS.

IRD Sète Modélisation de la dynamique des communautés de poissons qui vivent dans un estuaire du Sénégal – R. Duboz. Couplage Agent-DEVS, QSS.

PNEC Plan National d'Études Côtières, étude des interactions entre zooplanctons et phytoplanctons – É. Ramat. Agent-DEVS et QSS.

Remige

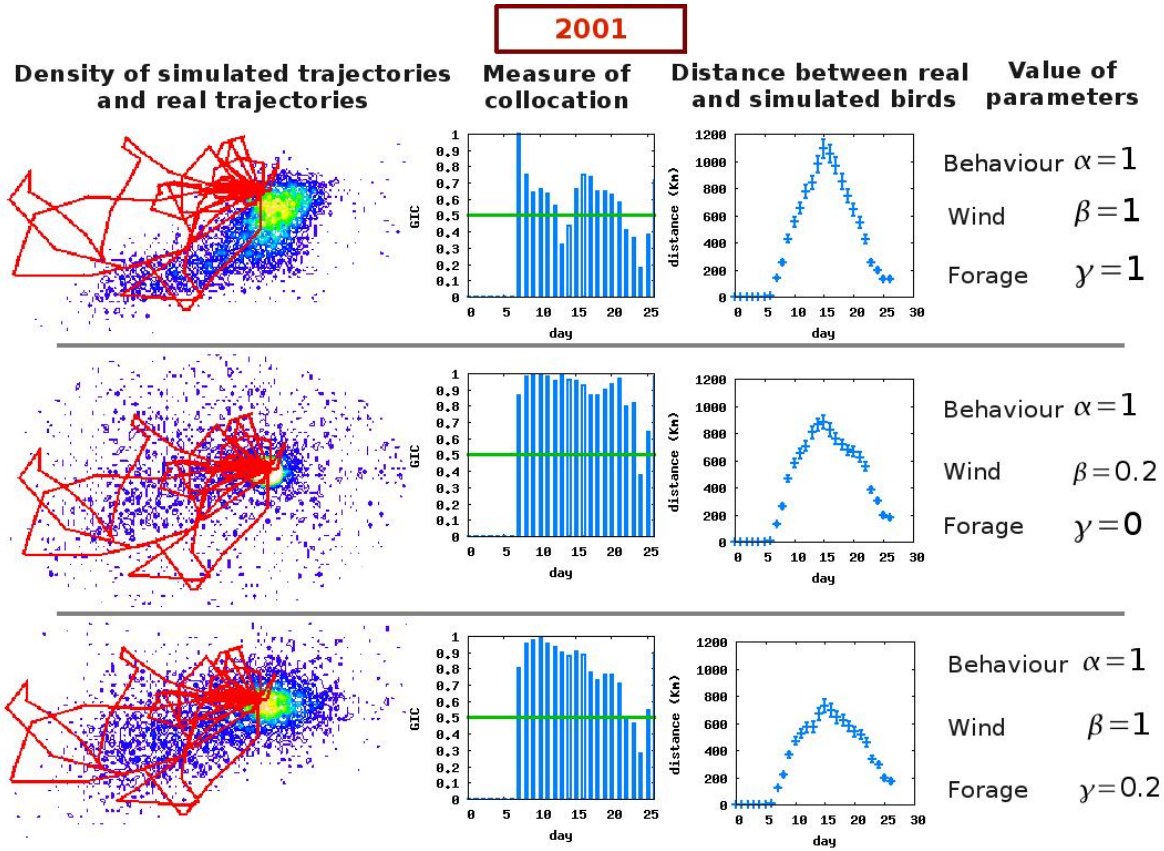
Facteurs qui influencent le déplacement des albatros des îles du Crozet



- Les trajectoires de 12 albatros en 2000 et 2001
- Question : quels sont les paramètres de direction des albatros ?

Remige

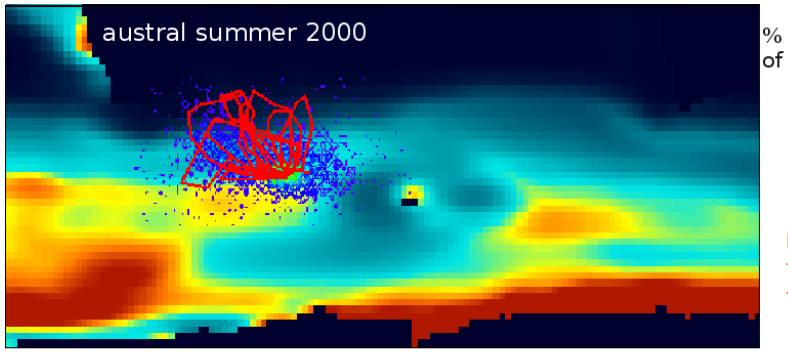
Facteurs qui influencent le déplacement des albatros des îles du Crozet



- Le comportement est gérée par : le comportement α, le vent β et la nourriture γ (des influences dans le calcul du comportement).

Remige

Facteurs qui influencent le déplacement des albatros des îles du Crozet



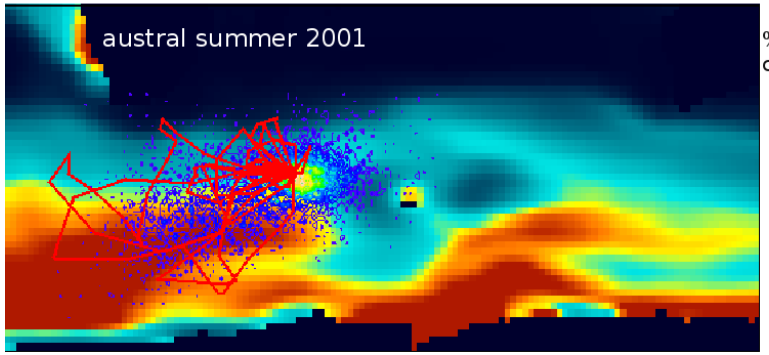
% weights in the computation of birds direction

food 4.8%

wind 47.6%

behavior 47.6%

mismatch between forage maximum and foraging range



% weights in the computation of birds direction

food 9.1%

wind 45.45%

behavior 45.45%

better match between forage maximum and foraging range

- Une estimation des paramètres pour trouver α, β et γ
- VLE en multi-simulation, plan d'expériences etc.
- Le vent et la nourriture, une base de données sous forme de SIG.

Plan

Perspectives à court terme

VLE

- Développement de nouvelles extensions
- Travaux sur la transformation de modèles : traducteur (Ex. MDA)
- Attacher de la sémantique aux ports : vérifications des messages échangés (Ex. types et unités)

Record

- Calibrage automatique et analyses de systèmes par méthodes statistiques : paquet rvle pour R
- Développement de couplage avec des méthodes d'optimisations : paquets pour Matlab/Scilab
- Construction d'extensions pour des modèles standards en agronomie.

→ **Volonté de simplifier les efforts de développement de modèles et d'outils.**

Perspectives à court terme

VLE

- Développement de nouvelles extensions
- Travaux sur la transformation de modèles : traducteur (Ex. MDA)
- Attacher de la sémantique aux ports : vérifications des messages échangés (Ex. types et unités)

Record

- Calibrage automatique et analyses de systèmes par méthodes statistiques : paquet rvle pour R
- Développement de couplage avec des méthodes d'optimisations : paquets pour Matlab/Scilab
- Construction d'extensions pour des modèles standards en agronomie.

→ Volonté de simplifier les efforts de développement de modèles et d'outils.

Perspectives à court terme

VLE

- Développement de nouvelles extensions
- Travaux sur la transformation de modèles : traducteur (Ex. MDA)
- Attacher de la sémantique aux ports : vérifications des messages échangés (Ex. types et unités)

Record

- Calibrage automatique et analyses de systèmes par méthodes statistiques : paquet rvle pour R
- Développement de couplage avec des méthodes d'optimisations : paquets pour Matlab/Scilab
- Construction d'extensions pour des modèles standards en agronomie.

→ **Volonté de simplifier les efforts de développement de modèles et d'outils.**

Approche formelle et opérationnelle de la multimodélisation et de la simulation des systèmes complexes

DEVS / VLE

Gauthier Quesnel

Institut National de la Recherche Agronomique


Barros, F. J. (1995).
Dynamic structure discrete event system specification: A new modeling and simulation formalism for dynamic structure systems.
In *Proceedings of the 1995 Winter Simulation Conference*, pages 781–785.


Kim, J. Y. and Kim, T. G. (1998).
A heterogeneous simulation framework based on the DEVS bus and the High Level Architecture.
In *Winter Simulation Conference*, Washington, DC.


Kim, Y. J. and Kim, T. G. (1996).
A heterogeneous distributed simulation framework based on DEVS formalism.
In *Sixth Annual Conference On Artificial Intelligence, Simulation and Planning in High Autonomy Systems*, pages 116–121, La Jolla, California, USA.


Kofman, E. and Junco, S. (2001).

Quantized State Systems. a DEVS Approach for Continuous Systems Simulation.
In *Transactions of SCS.*, volume 18, pages 123–132.


Quesnel, G., Duboz, R., and Ramat, É. (2004).
DEVS wrapping: A study case.
In *Proceedings of Conference on Conceptual Modeling and Simulation 2004*, pages 374–382, Genoa, Italy.


Zeigler, B. P. (1976).
Theory of Modeling and Simulation.
Wiley Interscience.


Zeigler, B. P., Kim, D., and Praehofer, H. (2000).
Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems.
Academic Press.