# Approche Logique pour la Modélisation et l'Analyse des Systèmes (LOGIMAS)

Marc Aiguier

February 8, 2011

## Recherche au MAS

**Modélisation et analyse des systèmes complexes par des techniques issues des mathématiques appliquées**

- **Techniques mathématiques.** EDP, probabilités, statistiques, analyse numérique, logique mathématique, théorie des catégories, mathématiques discrètes (graphes, combinatoire, théorie de l'information, etc.)
- **Techniques informatiques.**lgorithmique (parallèle et distribuée), génie-logiciel formel (test et preuve, model-checking, structuration, raffinement, prototypage rapide), BD, apprentissage
- **Champs d'application.** Systèmes naturels et biologiques, systèmes financiers, traitements de l'image, systèmes industriels à logiciels prépondérants, traitement des données.

## Équipe LOGIMAS, Ingénerie formelle des systèmes complexes

**Thèmes.** Définition de méthodes et techniques rigoureuses (structuration, abstraction, preuve, test et prototypage) visant à la conception et la validation de systèmes complexes (industriels et naturels).

1. Dénotation formelle et générique de la complexité par émergence de propriétés logiques
2. Formalisation de la science des systèmes par une approche coalgébrique et des techniques d'analyse non-standard pour unifier le temps
3. Applications
   - Modélisation de systèmes industriels hétérogènes
   - Étude de l'émergence dans les systèmes de régulations génétiques

## Intêrét du formel

Les systèmes industriels modernes ne peuvent plus être appréhendés par un homme ou une équipe (trop gros, trop compliqués, trop complexes). Besoins de méthodes de conception.

Avantage des méthodes formelles :

1. Seul moyen d'aborder scientifiquement la correction des systèmes
2. Génération d'outils pour la conception (structuration, raffinement et prototypage rapide) et la vérification (Test et preuve)

## Vers une caractérisation des systèmes complexes par l'émergence de propriétés

### Qu'est-ce qu'un système complexe ?

Un système complexe est décrit récursivement comme une interconnexion de sous-systèmes (hétérogènes) tel que son comportement est holistique (i.e. il ne peut-être considéré que globalement).

### Conséquence

Apparition de propriétés globales non anticipées à partir de la connaissance des sous-systèmes.

## Adages populaires (sources Wikipedia)

- « un système composé d'un grand nombre d'entités en interaction »
- « possédant des propriétés émergentes non-déductibles de celles de ses éléments »
- « le Tout est plus que la somme de ses parties »

## Solution pour la formalisation : Émergence

### Idée générale de notre formalisation

Soit $XY$ composé de deux sous-systèmes $X$ and $Y$.
Soit $F$ une fonction mathématique qui donne la "richesse" potentielle de $XY$, $X$ et $Y$,
et soit $+$ l'operation qui permet de combiner les richesses potentielles des sous-systèmes entre eux.

### $XY$ **complexe** signifie alors

- Non-conformité : $\exists a \in F(X) \cup F(Y),\ a \notin F(XY)$, ou
- Vraie émergence : $\exists a \in F(XY),\ a \notin F(X) + F(Y)$

Sinon, $XY$ est dit **modulaire**, i.e. $F(XY) = F(X) + F(Y)$

## Notre cadre

1. $X$ et $Y$ sont des sous-systèmes dont les comportements sont spécifiés dans un langage formel donné (e.g. systèmes logiques, systèmes de transition, réseaux de Petri ou encore des réseaux de régulations),
2. le système $XY$ est construit à partir des sous-systèmes $X$ et $Y$ par un connecteur architectural (union, produit synchronisé, plongement)
3. $F(X)$: ensemble des propriétés exprimées dans un système logique donné (e.g. logique du 1er ordre ou modale) satisfaites par $\mathrm{X}$ (i.e. théorie de $X$ notée $\mathrm{X}^\bullet$),
4. et $F(X) + F(Y) = (\mathrm{X}^\bullet \cup \mathrm{Y}^\bullet)^\bullet$.

## Classique dans les autres sciences : les indécidables de Godel

### $X$ : arithmétique de Peano

- 0 n'est successeur de personne, tout entier autre que 0 est successeur de quelqu'un, la fonction successeur *succ* est injective, définition inductive de $+$ par rapport à 0 et *succ*, et
- schéma d'axiome pour l'induction.

### $Y$ : théorie des ensembles de ZFC

- axiome d'extentionalité, de la réunion, des parties, etc.
- et axiome du choix

$XY = X \cup Y$

### Propriété émergente

Consistence de l'arithmétique : pas prouvable dans $X$ (Godel) mais prouvable dans $XY$ (Gentzen).

## Moins standard dans les sciences de l'ingénieur

### Spécification élémentaire d'un ascenseur $X$

Chaque appel est enregistré dans une file et donc est retiré suivant une politique FiFo.

Pas de risque de famine dans F(X): tout appel sera à un moment donné satisfait (sauf panne)

### Spécification $Y$ d'un étage prioritaire dans un ascenseur

Les appels venant de cet étage sont mis en tête de la file.

## Un autre exemple (suite)

### Spécification $XY$ d'un ascenseur avec étage prioritaire

- Chaque appel provenant d'un étage non-prioritaire est enregistré dans la file suivant la politique FiFo.
- Chaque appel provenant de l'étage prioritaire est enregistré en tête de la file.

### Propriété émergente

Famine possible: il existe des scénarios où un appel non-prioritaire n'est jamais satisfait.

## Généricité

Cadre générique pour exprimer le langage formel, les spécifications et les connecteurs :

- **Le langage formel :** institutions [Goguen&Burstall92]
- **Spécifications :** objets abstraits au-dessus des institutions
- **Connecteur :** abstraction des primitives de structuration par des outils formels de la théorie des catégories (limite et co-limite, transformation naturelle et adjonction)

# Catégorie : définition

Une catégorie $C = (G, A)$ est la donnée :

- d'une collection d'objets $G$, et
- d'une collection de morphismes $A$.

- chaque morphisme a une source et une cible target dans $G$. $C(i,j)$ représente l'ensemble des morphismes de source $i$ et de cible $j$.
  Si $f \in C(i,j)$, nous écrivons alors $f : i \to j$ ou bien $i \xrightarrow{f} j$.
- Les morphismes peuvent être composés en accord avec les sources et les cibles :

$$\begin{array}{cccccc} \circ : & C(i,j) & \times & C(j,k) & \to & C(i,k) \\ & i \xrightarrow{f} j & & j \xrightarrow{g} k & \mapsto & i \xrightarrow{g \circ f} j \end{array}$$

- $\circ$ est associative ($(h \circ g) \circ f = h \circ (g \circ f)$)
- pour chaque objet $i$, il y a un morphisme identité $id_i \in C(i,i)$ neutre pour $\circ$.

Note : Les objets avec leurs morphismes peuvent être représentés par des graphes où les chemins correspondent aux compositions des morphismes par la loi $\circ$.

## Exemples

- La catégorie $\mathcal{E}ns$ (Set), dont les objets sont les ensembles, et les morphismes les applications, avec la composition usuelle des applications.
- La catégorie $\mathcal{O}rd$, dont les objets sont les ensembles ordonnés et les morphismes les applications croissantes.
- La catégorie $\mathcal{T}op$, dont les objets sont les espaces topologiques, et les morphismes les applications continues.
- La catégorie $\mathcal{G}rp$, dont les objets sont les groupes et les morphismes les homomorphismes.
- La catégorie $\mathcal{E}v$, dont les objets sont les espaces vectoriels et les morphismes les applications linéaires.
- $\ldots$

# Catégorie duale et foncteur

## Catégorie duale

La catégorie duale ou catégorie opposée $C^{op}$ d'une catégorie $C$ est la catégorie définie en "inversant" les morphismes de $C$, i.e. $|C^{op}| = |C|$ et $f^{op} \in C^{op}(i,j) \Leftrightarrow f \in C(j,i)$.

## Foncteur

Soient $C$ and $D$ deux catégories. Un foncteur $F$ de $C$ dans $D$ est défini par une application de $|C| \to |D|$ et une famille d'applications $F_{i,j} : C(i,j) \to D(F(i), F(j))$ tel que :

- $F_{i,i}(Id_i) = Id_{F(i)}$
- $F(g \circ f) = F(g) \circ F(f)$

Exemple : $\mathcal{P} : Set^{op} \to Set$ défini par :

- $S \mapsto \mathcal{P}(S)$
- $(f^{op} : S' \to S) \mapsto (f^{-1} : \mathcal{P}(S') \to \mathcal{P}(S))$ où $f^{-1} : (A \subseteq S') \mapsto \{s \in S | f(s) \in A\}$

Cat : La catégorie des catégories dont les morphismes sont les foncteurs.

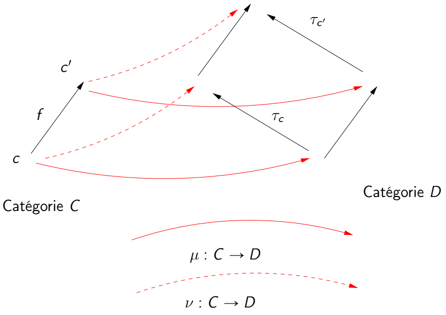
# Transformations naturelles

Étant donnés deux foncteurs $\mu : C \to D$ et $\nu : C \to D$, une transformation naturelle $\tau : \mu \to \nu$ est une application $c \in |C| \mapsto \tau_c \in D(\mu(c), \nu(c))$ de $D$ telle que pour tout $f : c \to c'$ dans $C$ nous avons:

$$\nu(f) \circ \tau_c = \mu(f) \circ \tau_{c'}$$

Fonc(C,D) : catégorie des foncteurs de $C$ dans $D$ dont les morphismes sont les transformations naturelles.

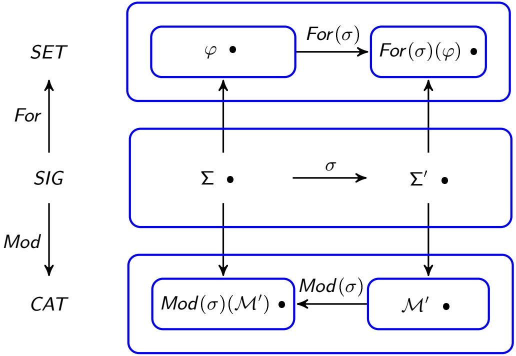
# Représentation graphique




Transformation naturelle $\tau : \mu \to \nu$

Le diagramme commute

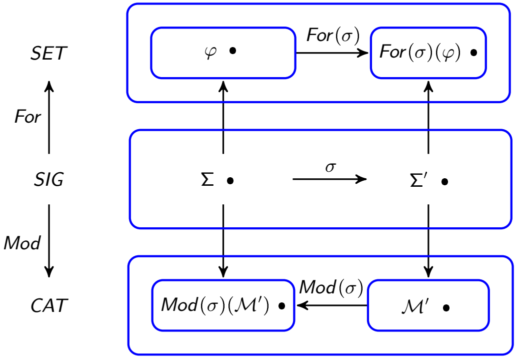
# Institution




relation de satisfaction : $\mathcal{M} \models_{\Sigma} \varphi$

condition de satisfaction :
$\mathcal{M}' \models_{\Sigma'} For(\sigma)(\varphi)$ ssi $Mod(\sigma)(\mathcal{M}') \models_{\Sigma} \varphi$

# Institution




relation de satisfaction : $\mathcal{M} \models_\Sigma \varphi$

condition de satisfaction :
$\mathcal{M}' \models_{\Sigma'} For(\sigma)(\varphi)$ ssi $Mod(\sigma)(\mathcal{M}') \models_\Sigma \varphi$

- $Mod(\Phi), \forall \mathcal{M} \in | Mod(\Phi) |, \forall\ \varphi \in \Phi,$

  $$\mathcal{M} \models_\Sigma \varphi$$

- Conséquence sémantiques de $\Phi$ :

  $\Phi^\bullet = \{\varphi \mid \forall \mathcal{M} \in Mod(\Phi), \mathcal{M} \models_\Sigma \varphi$

Une institution $\mathcal{I}$ : $(SIG, For, Mod, (\models_\Sigma)_{\Sigma \in |SIG|})$ qui vérifie la condition de satisfaction

# Exemple : logique équationnelle (1)

- Signature :

$$\boxed{\Sigma = (S, F)}$$

  - $S$ : ensemble de types
  - $F = \{f : s_1 \times \cdots \times s_n \to s\}$

- Morphisme de signatures :

$$\boxed{\sigma : (S, F) \to (S', F')}$$

  - $\sigma = (\sigma_S, \sigma_F)$ préserve les types

- Algèbre :

$$\boxed{\mathcal{A} = (A, (f_{\mathcal{A}})_{f \in F})}$$

  - $A = (A_s)_{s \in S}$
  - $f_{\mathcal{A}} : A_{s_1} \times \cdots \times A_{s_n} \to A_s$ pour $f : s_1 \times \cdots \times s_n \to s \in F$

- Le réduit d'un modèle par $\sigma$ :

$$\boxed{Mod(\sigma)((A', (f_{\mathcal{A}'})_{f \in F})) = (A, (f_{\mathcal{A}})_{f \in F})}$$

  - pour tout $s \in S, A_s = A'_{\sigma_S(s)}$
  - $f_{\mathcal{A}} = \sigma_F(f)_{\mathcal{A}'}$

## Exemple : logique équationnelle (2)

- Formules :

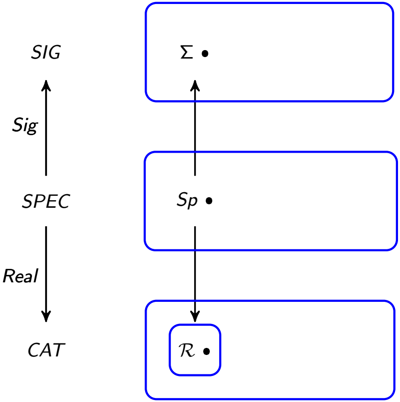
$$\boxed{\varphi ::= t = t' \mid \neg\varphi \mid \varphi \vee \varphi}$$

  $t$ et $t'$ deux termes sur $\Sigma = (S, F)$

- Relation de satisfaction : $\mathcal{A}$ algèbre de $\Sigma$
  - $\nu : V \to A$ une interprétation des variables, étendues aux termes
  - $\mathcal{A} \models^{\nu}_{\Sigma} t = t'$ ssi $\nu(t) = \nu(t')$
  - $\mathcal{A} \models_{\Sigma} \varphi$ ssi pour tout $\nu$, $\mathcal{A} \models^{\nu}_{\Sigma} \varphi$

- Condition de satisfaction : vérifiée

# Cadre de spécifications au dessus de $\mathcal{I}$




- $Sp$ : objet abstrait
- $\Sigma$ : Syntaxe de la spécification
- $Real(Sp)$ : capture la sémantique de la spécification $Sp$

avec $Real(Sp)$ dans $Mod(Sig(Sp))$

# Cadre de spécifications au dessus de $\mathcal{I}$

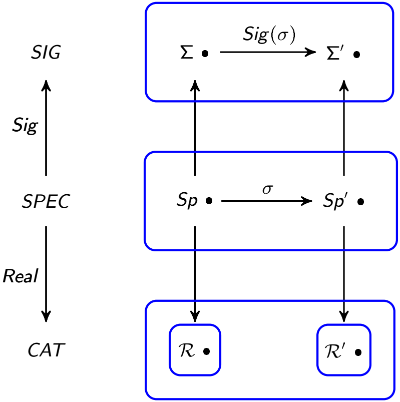
$$
\begin{tikzcd}
\mathit{SIG} & \Sigma \arrow[r, "\mathit{Sig}(\sigma)"] & \Sigma' \\
\mathit{SPEC} \arrow[u, "\mathit{Sig}"] \arrow[d, "\mathit{Real}"'] & \mathit{Sp} \arrow[u] \arrow[r, "\sigma"] \arrow[d] & \mathit{Sp}' \arrow[u] \arrow[d] \\
\mathit{CAT} & \mathcal{R} & \mathcal{R}'
\end{tikzcd}
$$

- $\mathit{Sp}$ : objet abstrait
- $\Sigma$ : Syntaxe de la spécification
- $\mathit{Real}(\mathit{Sp})$ : capture la sémantique de la spécification $\mathit{Sp}$

- $\mathit{Real}$ : simple application (non foncteur)
- Capture de nombreuses classes de spécifications

avec $\mathit{Real}(\mathit{Sp})$ dans $\mathit{Mod}(\mathit{Sig}(\mathit{Sp}))$

# Cadre de spécifications au dessus de $\mathcal{I}$




- $Sp$ : objet abstrait
- $\Sigma$ : Syntaxe de la spécification
- $Real(Sp)$ : capture la sémantique de la spécification $Sp$

- $Real$ : simple application (non foncteur)
- Capture de nombreuses classes de spécifications

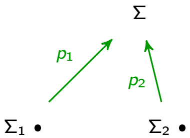
avec $Real(Sp)$ dans $Mod(Sig(Sp))$

- Un cadre de spécifications au dessus de $\mathcal{I}$ : $(SPEC, Sig, Real)$
- $Sp^\diamond = \{\varphi \in For(Sig(Sp)) \mid \forall \mathcal{R} \in |Real(Sp)|, \mathcal{R} \models_{Sig(Sp)} \varphi\}$

## Intuition : Union des spécifications algébriques

- Union de deux spécification algébriques : $Sp_1 = (\Sigma_1, Ax_1)$ et $Sp_2 = (\Sigma_2, Ax_2)$

$$\boxed{Sp = Union(Sp_1, Sp_2)}$$




- Inclusion

## Intuition : Union des spécifications algébriques

- Union de deux spécification algébriques : $Sp_1 = (\Sigma_1, Ax_1)$ et $Sp_2 = (\Sigma_2, Ax_2)$

$$\boxed{Sp = Union(Sp_1, Sp_2)}$$

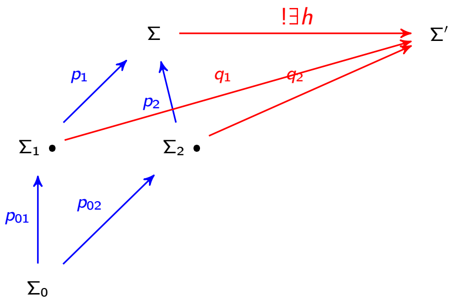
$$\begin{tikzcd}
 & \Sigma & \\
\Sigma_1 \bullet \arrow[ur, "p_1"] & & \Sigma_2 \bullet \arrow[ul, "p_2"'] \\
\Sigma_0 \arrow[u, "p_{01}"] \arrow[urr, "p_{02}"] & &
\end{tikzcd}$$

- Inclusion
- Partage

# Intuition : Union des spécifications algébriques

- Union de deux spécification algébriques : $Sp_1 = (\Sigma_1, Ax_1)$ et $Sp_2 = (\Sigma_2, Ax_2)$

$$\boxed{Sp = Union(Sp_1, Sp_2)}$$



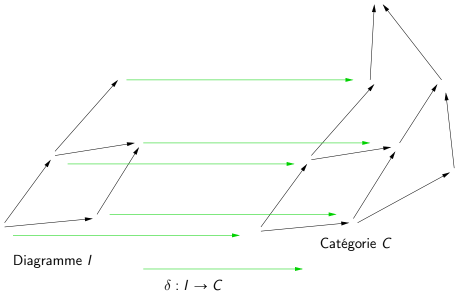

- Inclusion
- Partage
- $\Sigma$ minimal

# Diagramme

$I$, $C$ : deux catégories.

## Diagramme

Un diagramme dans $C$ de structure $I$ est un foncteur $\delta : I \to C$.




$\Delta_{(I,C)}$ : catégorie des diagrammes dans $C$ de structure $I$ avec pour morphisme les transformations naturelles entre tout $\delta, \delta' : I \to C$.

# Co-cone

$\delta : I = (G_I, A_I) \to C$ un diagramme.

## Co-cone

Un co-cone de base $\delta$ est un objet $z$ de $C$ avec une famille $\{p_i : \delta(i) \to z\}_{i \in I}$ de morphismes dans $C$, noté simplement $p : \delta \to z$, tel que pour tout $f : i \to j$ in $I$, $p_i = p_j \circ \delta(f)$.

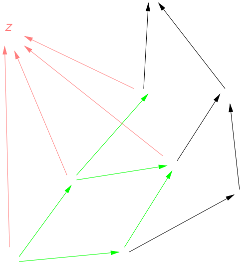



Diagramme $I$

Catégorie $C$

Cocone de base $\delta : I \to C$

# Co-limite

Co-cone initial parmi les co-cones de base $\delta$.

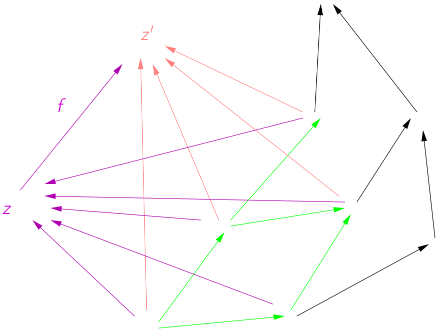


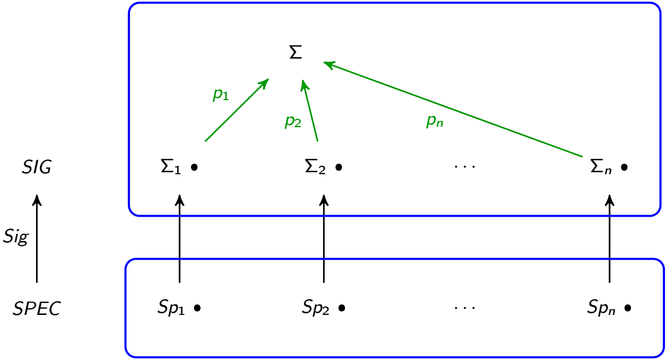

Diagramme $I$

Catégorie $C$

Colimite de base $\delta : I \to C$

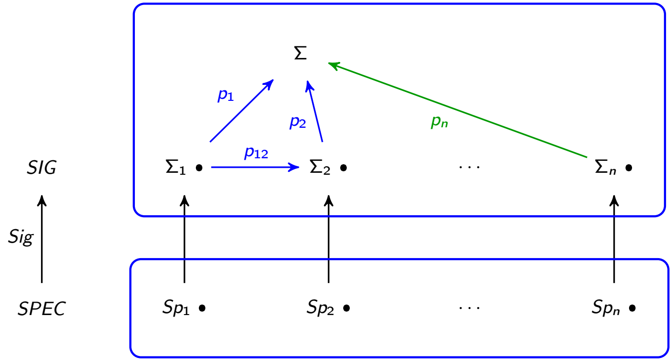
(unicité de $f : z \to z'$)

## Connecteur de spécifications



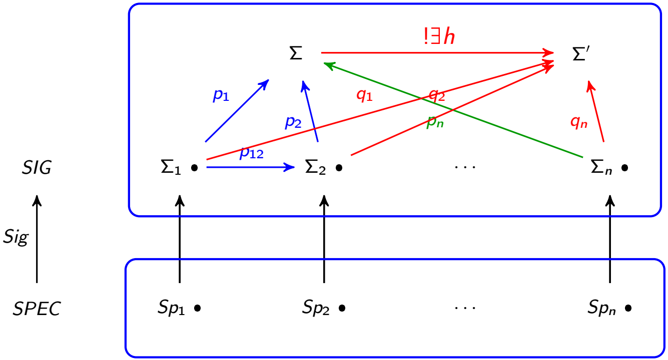

- Inclusion

# Connecteur de spécifications




- Inclusion
- Partage

# Connecteur de spécifications



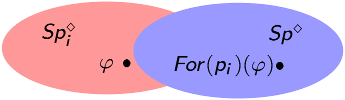

- Inclusion
- Partage
- $\Sigma$ minimal

- Un connecteur $c = (\Sigma, (p_i : \Sigma_i \to \Sigma)_{i \in \{1,\ldots,n\}})$
- $Sp$ tq. $Sig(Sp) = \Sigma$ : spécification composée
- $Sp_i$ : spécification composante

# Spécification modulaire/complexe

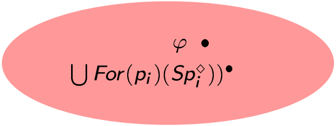
- *Sp* est modulaire si et seulement si les deux propriétés suivantes sont satisfaites:
  - Conformité:

$$\forall p_i, \forall \varphi \in For(Sig(Sp_i))$$



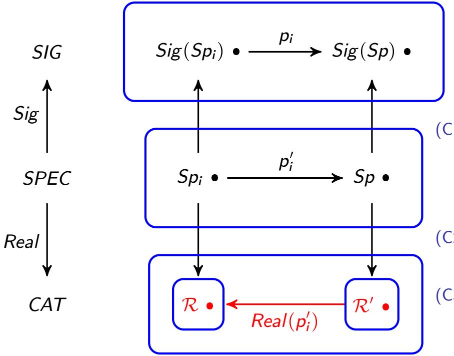

  - Non vraie émergence :

$$\forall \varphi \in Sp^\diamond$$




- Si *Sp* n'est pas modulaire, alors elle est complexe

# Conditions de modularité




(C1) *Real* un foncteur contravariant

$$\mathcal{R}' \in |Real(Sp)| \Rightarrow Real(p'_i)(\mathcal{R}') \in |Real(Sp_i)|$$

(C2) $\forall \mathcal{R} \in |Real(Sp_i)|, \exists \mathcal{R}' \in |Real(Sp)|,$
$Real(p'_i)(\mathcal{R}') = \mathcal{R}$

(C3) $\forall \varphi \in For(Sig(Sp_i)), \forall \mathcal{R}' \in |Real(Sp)|,$
$\mathcal{R}' \models_{Sig(Sp)} For(p_i)(\varphi) \Rightarrow Real(p'_i)(\mathcal{R}') \models_{Sig(Sp_i)} \varphi$

$(C1) + (C2) + (C3)$ implique *Sp* n'admet pas de propriétés de non-conformité

## Ingénierie systèmes et science des systèmes

- Science et technique pluridisciplinaire pour traiter :
  - d'objets (existants ou à créer) appelés **systèmes**
  - pouvoir supporter des raisonements sur les propriétés structurelles de ces objets

**L'ingénierie système est une réponse aux difficultés pratiques et conceptuelles de la description et de la conception de systèmes**

- 3 acceptations : Un modèle, une méthode, une science
- Générique : peut s'appliquer aux systèms industriels, IT, organisationnels, etc.

## Ingénierie systèmes ?

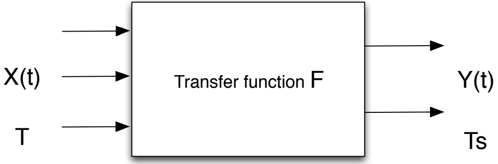
### Consensus autour de concepts clés

- **Système.** Fonctions de transfert causale implantées par des systèmes de transition
- **Besoins fonctionnels (requirements).** Propriétés temporelles
- **Intégration**
  - Opérations de composition
  - Opération d'abstraction

## Modélisation des systèmes

**Système.** objet mathématique :

- qui transforme les entrées X (reçues à $\neq$ moments sur une échelle de temps TX) en des sorties Y (reçues à des moments dans TY)
- caracterisé par ses états internes (évoluants dans une autre échelle de temps $TQ$) tels que $TX, TY \subseteq TQ$




$$F : X^{TX} Y^{TY}$$
$$X : TX \rightarrow Input$$
$$Y : TY \rightarrow Output$$

$$\forall t \in TQ, \forall X, X', X :: t = X' :: t \Longrightarrow F(X) :: t = F(X') :: t$$

# Implantation des systèmes

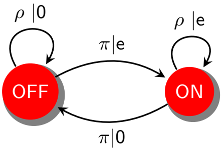
## Définition

Une co-algèbre $(Q, \alpha : Q \to H(Q))$ sur la signature/foncteur $H = (Out \times \_)^{In}$ munie :

- d'un état distingué $q_0$ dénotant l'**état initial**
- de 2 échelles de temps $TX$ et $TY$

Par des résultats de terminalité dans les co-algèbres, on peut calculer la fonction de transfert à partir de son implantation.

## Lampes en temps discrets




- $T_i = T_o = \mathbb{N}$,
- $In = \{\rho, \pi\}$ ($\rho$ bouton relaché, $\pi$ pressé)
- $Out = \{0, e\} \subseteq \mathbb{R}^+$ (niveau d'énergie produit par la lampe)

## Lampes en temps continus

Observation d'un signal continu $y(t)$ donnant le niveau d'énergie produit par la lampe à $t$.

$$y'(t) = e \times x(t) - k \times y(t)$$
$$y(0) = 0$$

- $x(t)$ 1 où 0 (bouton pressé ou relevé)
- $k > 0$ paramètre réel pour exprimer la rapidité de la lumière par rapport à l'état du bouton

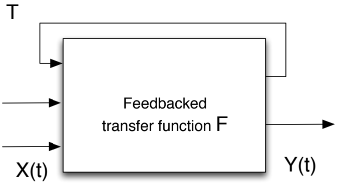
## Intégration

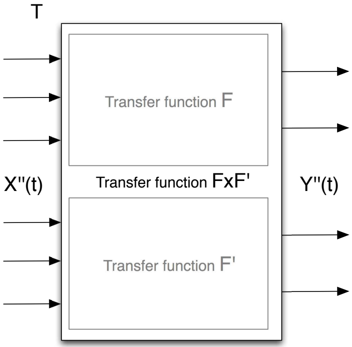
Moyen de construire de plus grands systèmes à partir de systèmes existants par des opérations de composition ...

Le mur comme résultat de l'intégration d'un tas de briques ...

# Opérations de composition

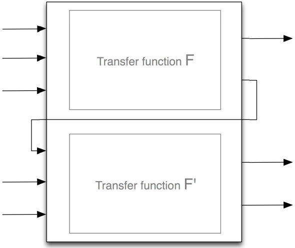
2 opérations de base : Feedback et produit algébrique

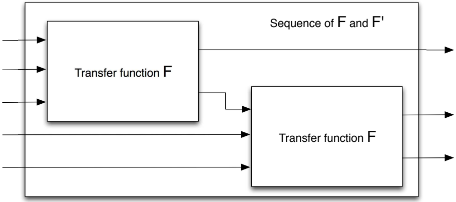


Suffisant pour représenter les autres opérateurs

**Exemple :** Séquentialité



## Abstraction

- **Abstraction de flôts de données :** Fonction de transfert $A : D_c^{T_c} \rightarrow D_a^{T_a}$ surjective
- **Concrétisation de flôts de données :** Fonction inverse $C : D_a^{T_a} \mathcal{P}(D_c^{T_c})$
- **Abstraction de fonction de transfert :** $F : In^{T_i} \rightarrow Out^{T_o}$, $A_i : In^{T_i} \rightarrow In'^{T_i'}$ et $A_i' : Out^{T_o} \rightarrow Out'^{T_o'}$. Fonction de transfert $F_a : In'^{T_i'} \rightarrow Out'^{T_o'}$ telle que :

$$
\begin{tikzcd}
In^{T_i} \arrow[r, "F"] \arrow[d, "A_i"'] & Out^{T_o} \arrow[d, "A_o"] \\
In'^{T_i'} \arrow[r, "F_a"'] & Out^{T_o'}
\end{tikzcd}
$$

## Pour conclure

**Vision architecturale.** Opérationnelle, Fonctionnelle, Organique

**Question posée.** Pourquoi ?, Quoi ?, Comment ?

**Mots-clés.** Contexte/mission, Service/fonction, Composant/configuration

Notre formalisme permet de répondre au trois (une sémantique pour Sysml).