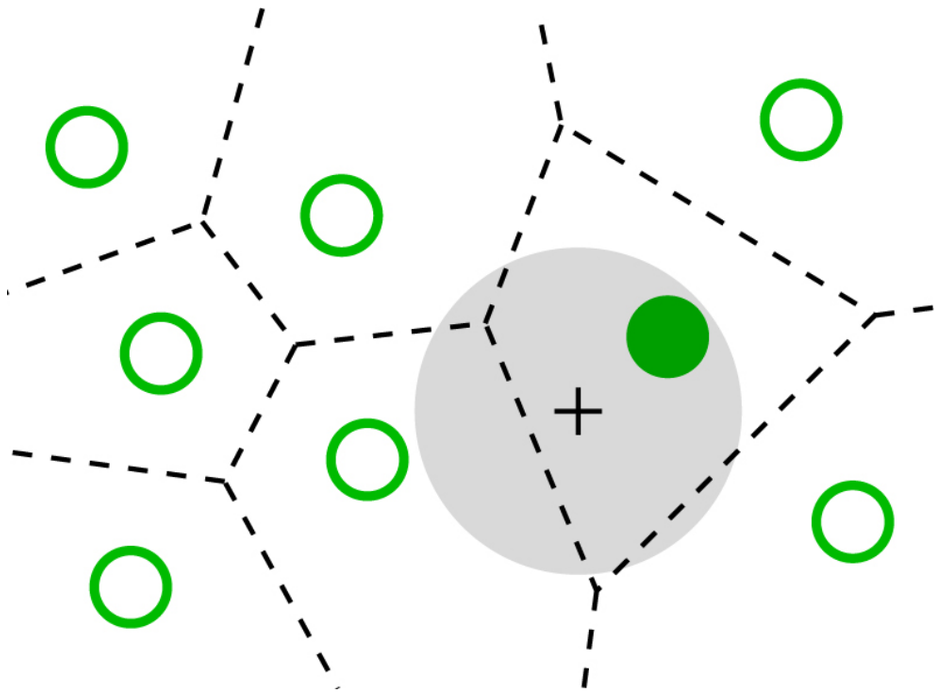
Application du modèle Entité-Composant-Système à la programmation d'interactions

Thibault Raffaillac, Stéphane Huot

Inria inventeurs du monde numérique

CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille

Programmation de nouvelles techniques d'interaction

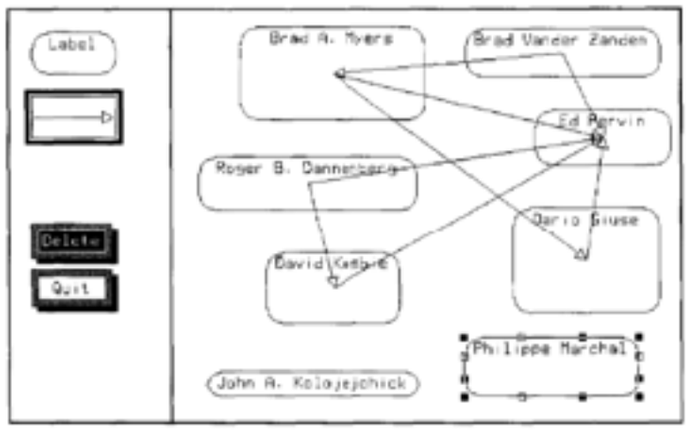


Bubble Cursor (Grossman et Balakrishnan)

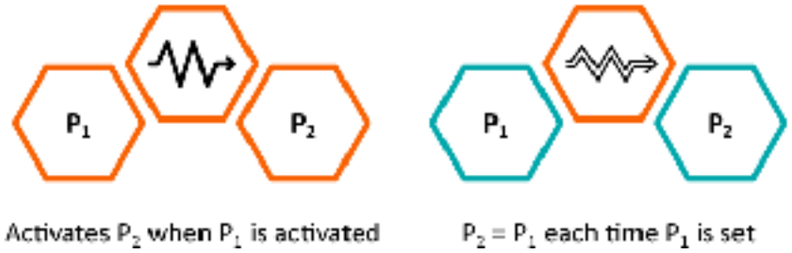
ExposeHK (Malacria et al.)

Utilisation de frameworks d'interaction **monolithiques**

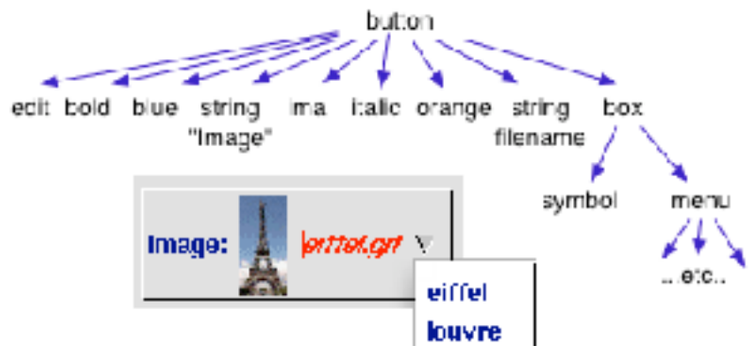
Solutions existantes



Garnet (Myers et. al)



Djnn (Chatty et. al)



UBit (Lecolinet)

Notre solution

Adaptation du modèle **Entité-Composant-Système** à la programmation d'IHM

Issu du domaine du Jeu Vidéo

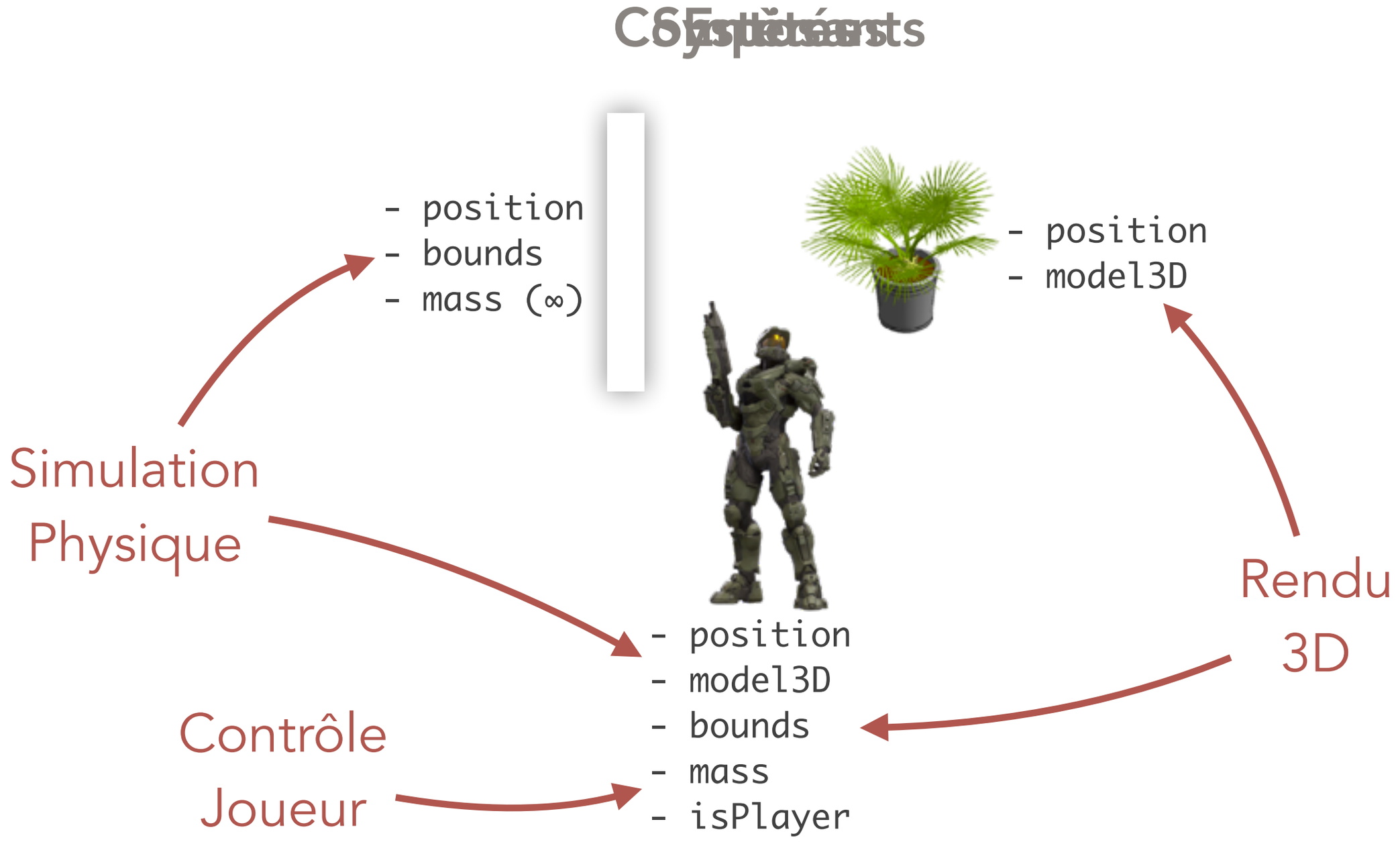
Défini par Adam Martin (2007)

Thief: The Dark Project (1998)

Dungeon Siege (2002)

unity

Entité-Composant-Système



Du Jeu Vidéo à l'IHM

Jeu Vidéo	IHM
MÀJ à tickrate fixe (ex. 60Hz)	MÀJ réactive à des événements
Relations faibles entre éléments	Relations de contenance et proximité
Un couple clavier/souris	Multiples périphériques
Pipeline d'exécution prédéterminé	Orchestration dynamique de comportements

Adaptation à l'IHM

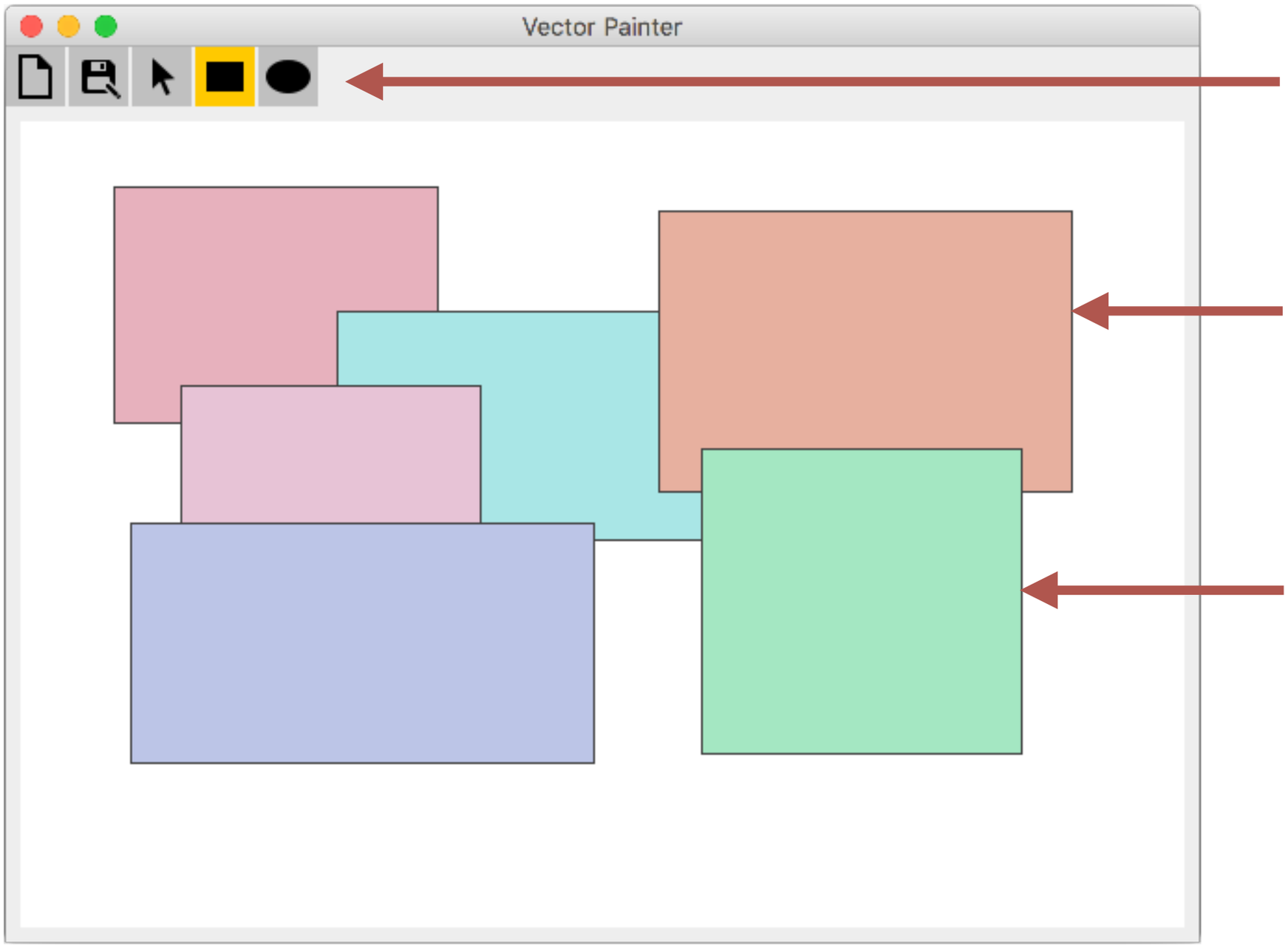
Tout est Entité !

⇒ widgets (boutons, menus, ...)

⇒ périphériques (souris, clavier, ...)

⇒ Systèmes

Exemple d'application

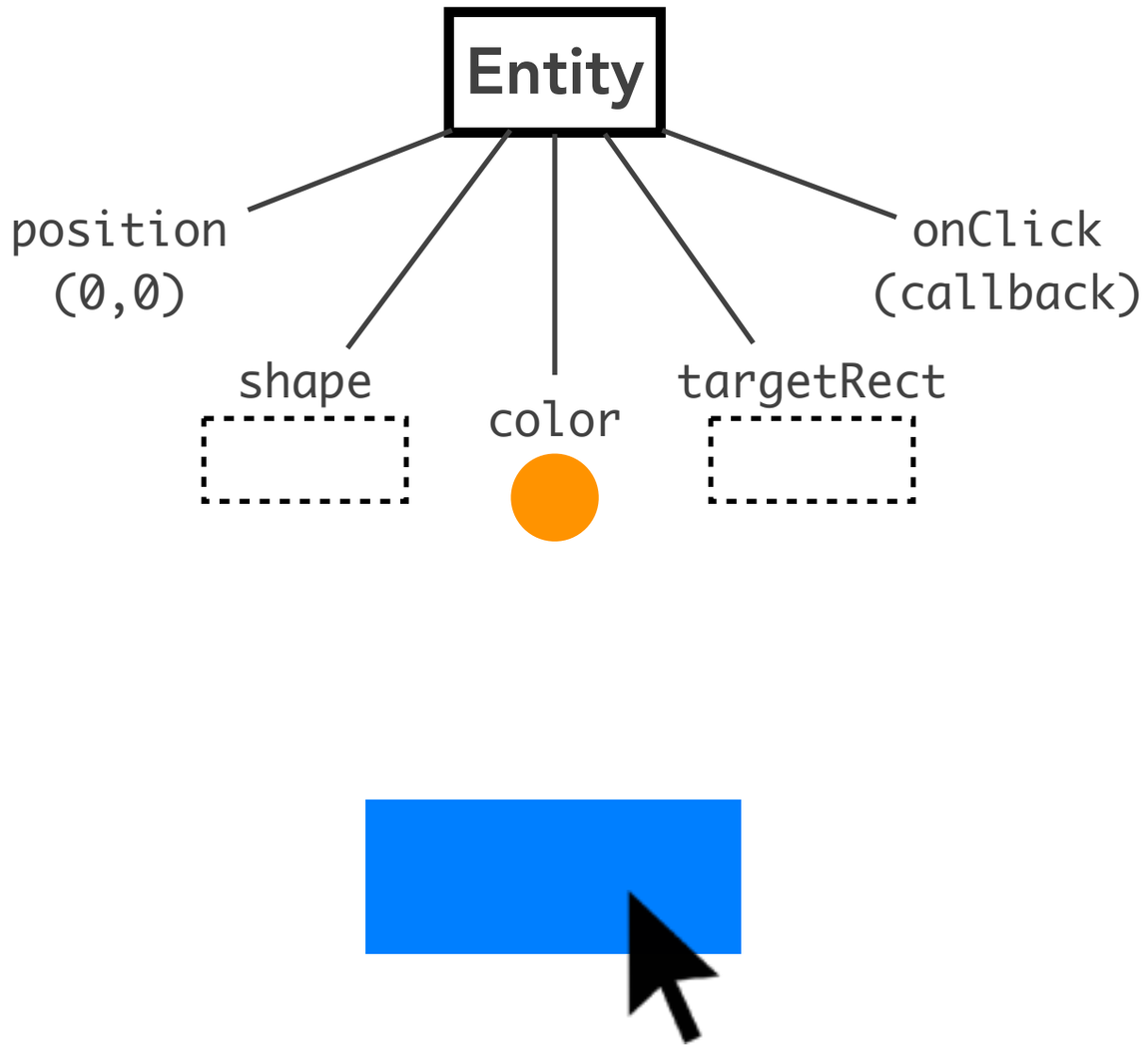


gestion de divers modes d'interaction

tracé de formes par appui-déplacement

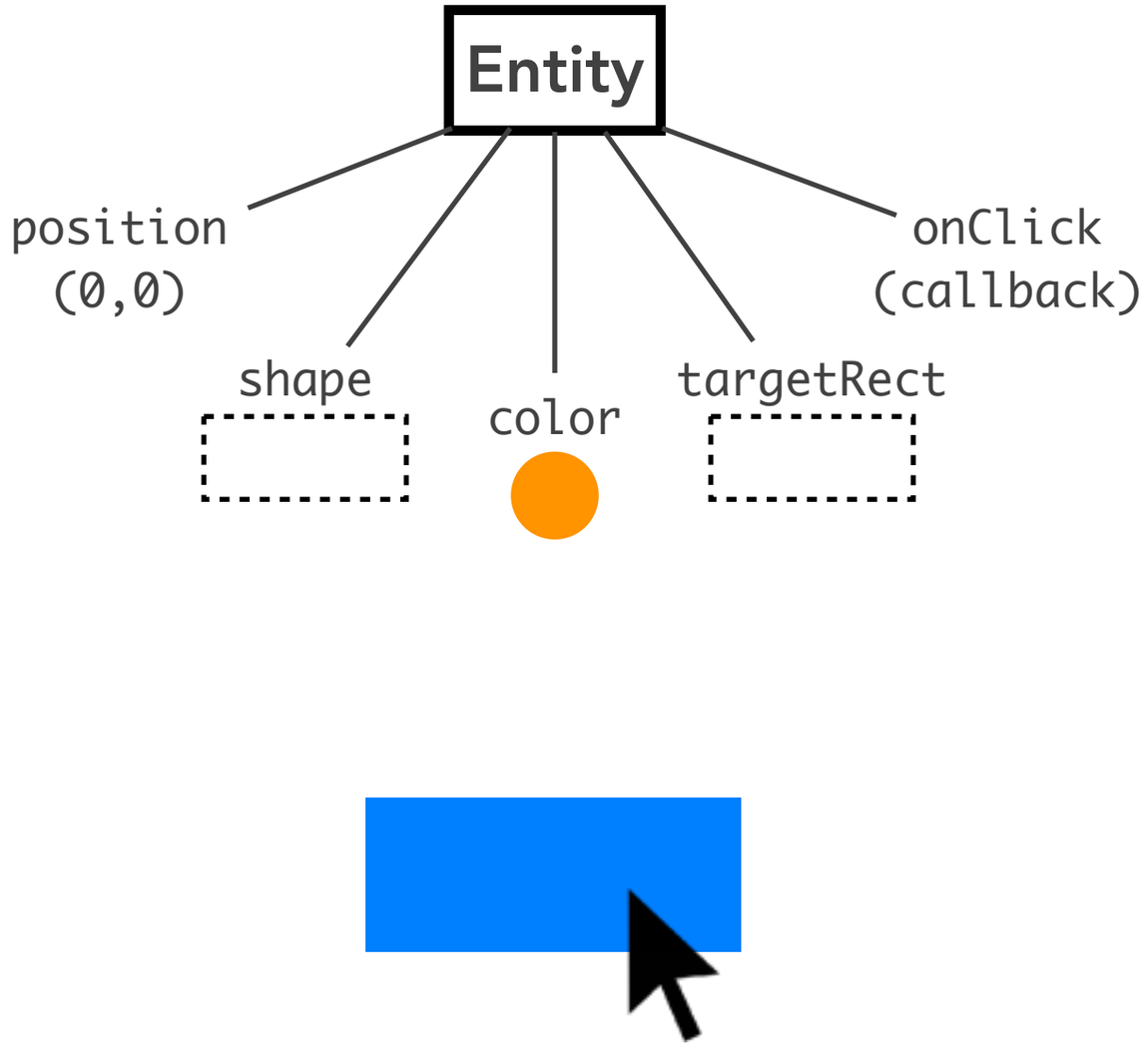
modification des formes par glisser-déposer

Création d'Entité




```
e = new Entity()

e.add("position",
   new Point(0,0))

e.add("shape",
   new Rectangle(0,0,50,20))

e.add("color",
   new Color(255,127,0))

e.add("targetRect",
   e.get("shape"))

e.add("onClick",
   () => {e.color.set(0,127,255)})
```

Fabrique d'Entité




```
e = new Button(...)
```

Caractéristiques d'une Entité

Ajout/retrait de champs (*Composants*) dynamique
↳ **dictionnaire**

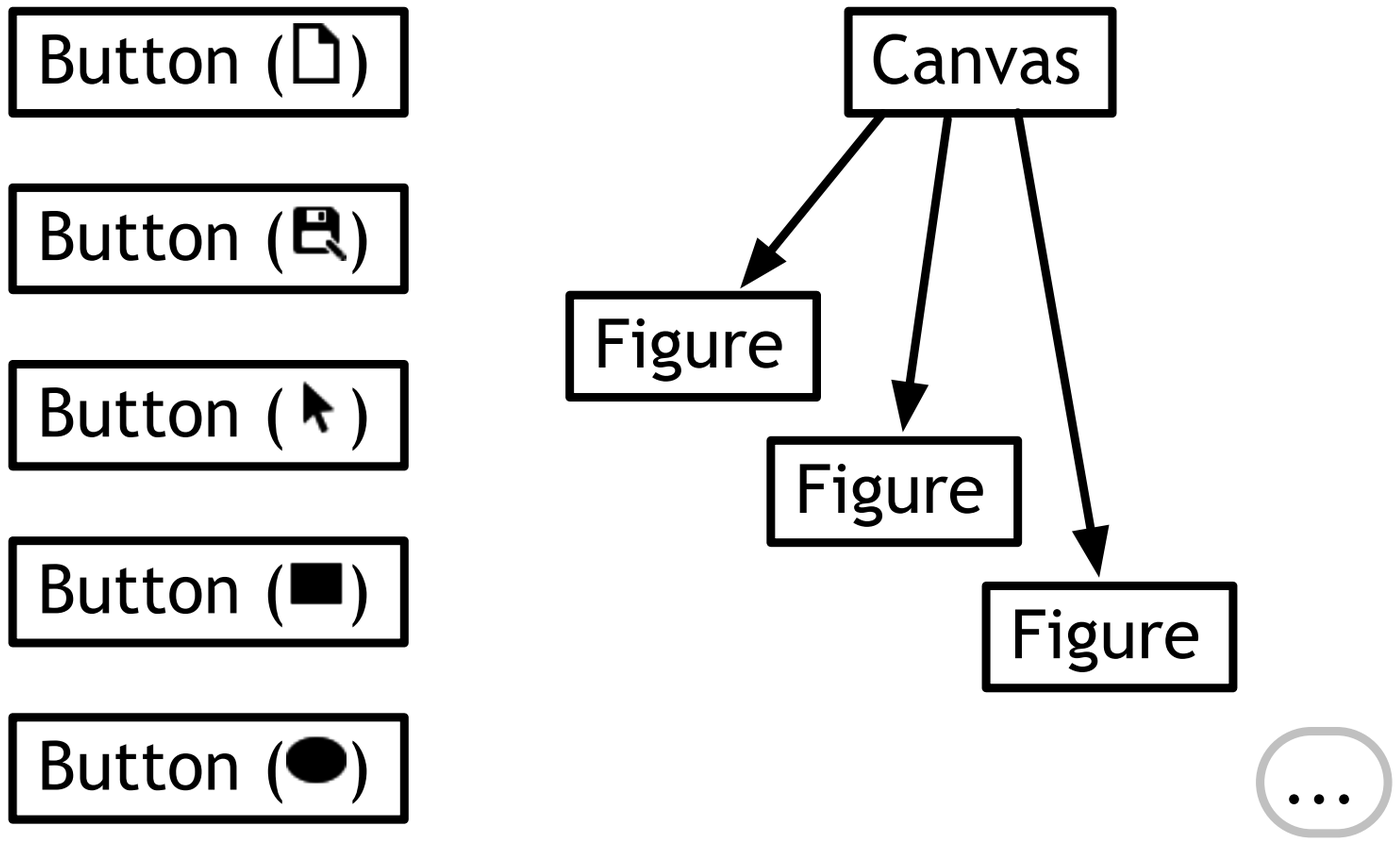
Suppression manuelle (pas de ramasse-miettes)

Visible de tous (à partir des Composants qu'elle possède)
↳ **existe** dès sa création, sans arbre de scène

Aucun héritage (classe ou prototype)

Liste des Entités actives

Interface



Sélecteurs

Obtenir une liste des Entités possédant certains Composants

↳`selectAll(composantA, composantB, ...)`

Analogue aux sélecteurs CSS et tags de SwingStates

Utilisation systématique de caches pour optimiser

Implémentation d'un Système

```
s = new Entity()
s.add("execute", () => {
  targets = selectAll("position",
                      "targetShape")
  for (p : selectAll("isPointer")) {
    t = findClosest(p, targets)
    p.add("target", t)
    t.add("pointedBy", p)
  }
})
s.add("readsMouseMotion", true)
```

Un Système est caractérisé par le Composant execute

Liste des cibles

Liste des pointeurs

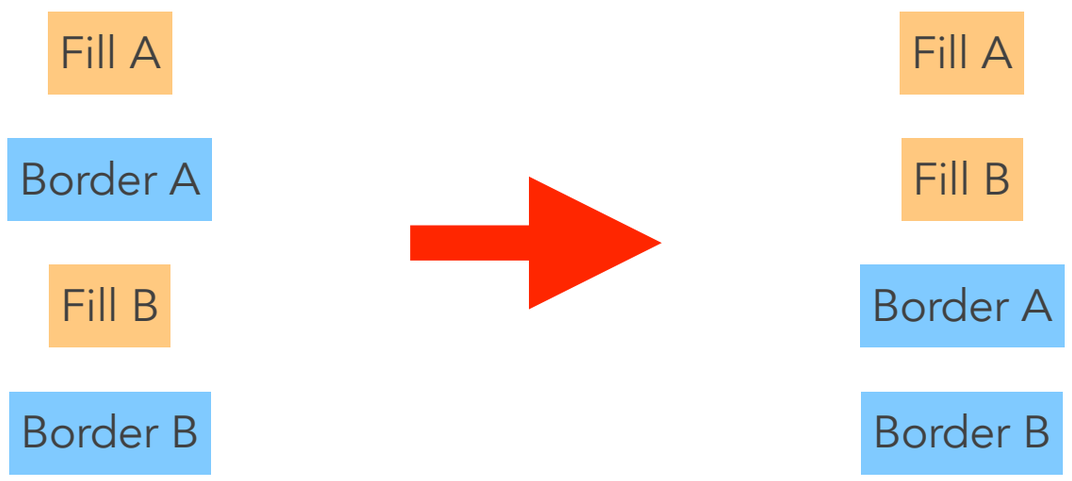
Bubble Cursor

Quand s'exécuter

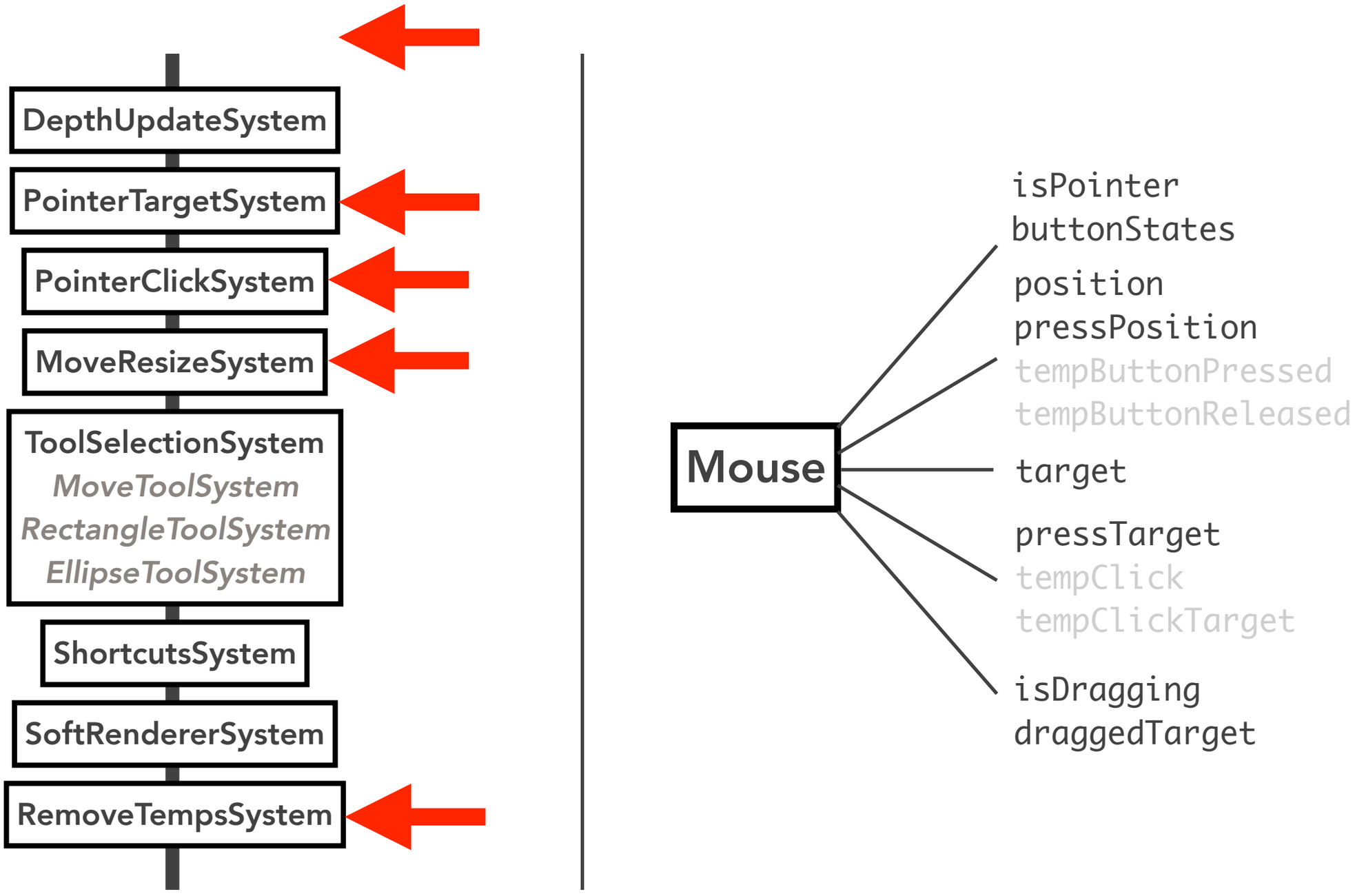
Caractéristiques d'un Système

Exécute **un** type de comportement sur **plusieurs** Entités (ex. tracé de bordures, ciblage de la souris, déclenchement de raccourci clavier)

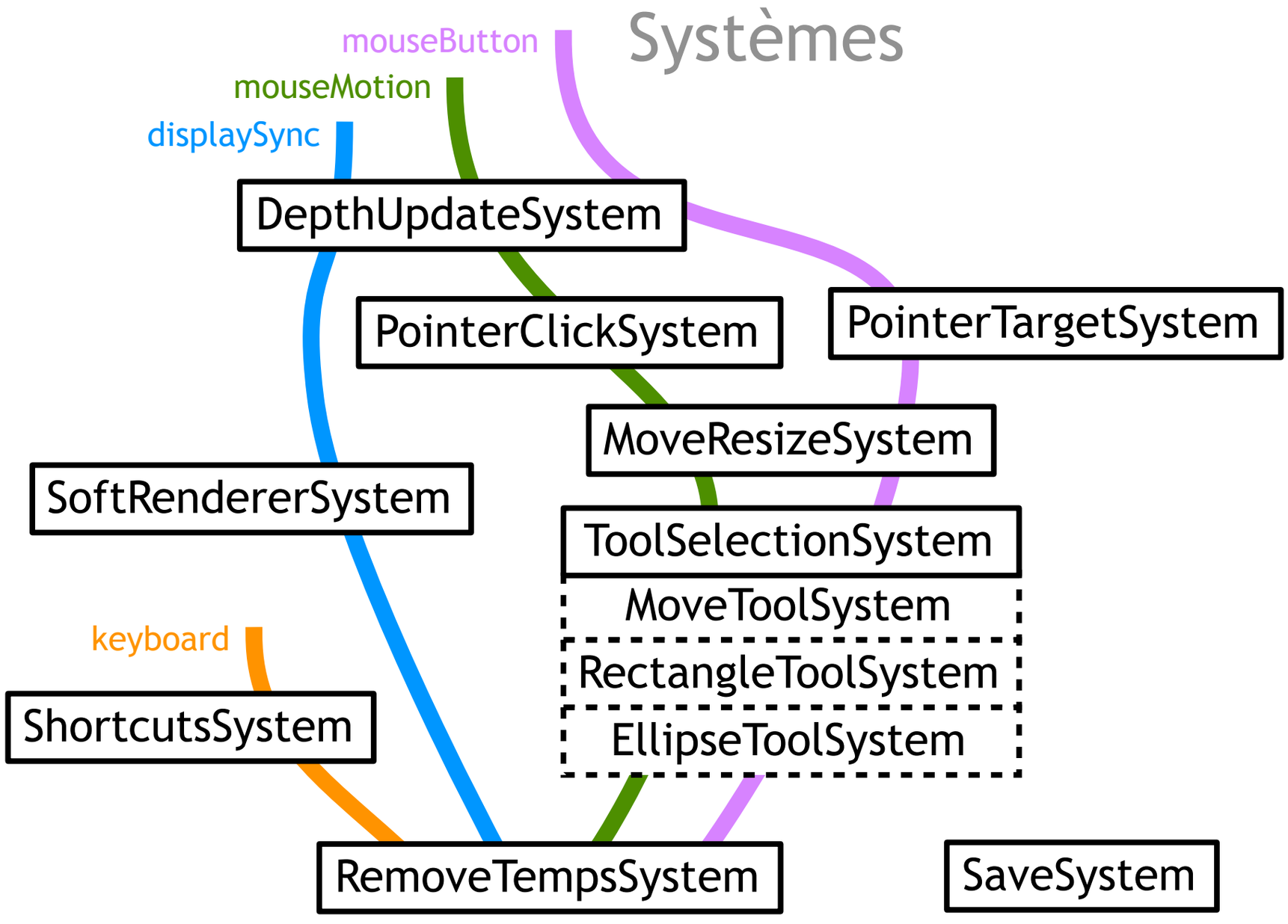
Ordonne les comportements par *type* plutôt qu'*Entité*



Chaîne des Systèmes



Chaînes des Systèmes



Liste des périphériques actifs

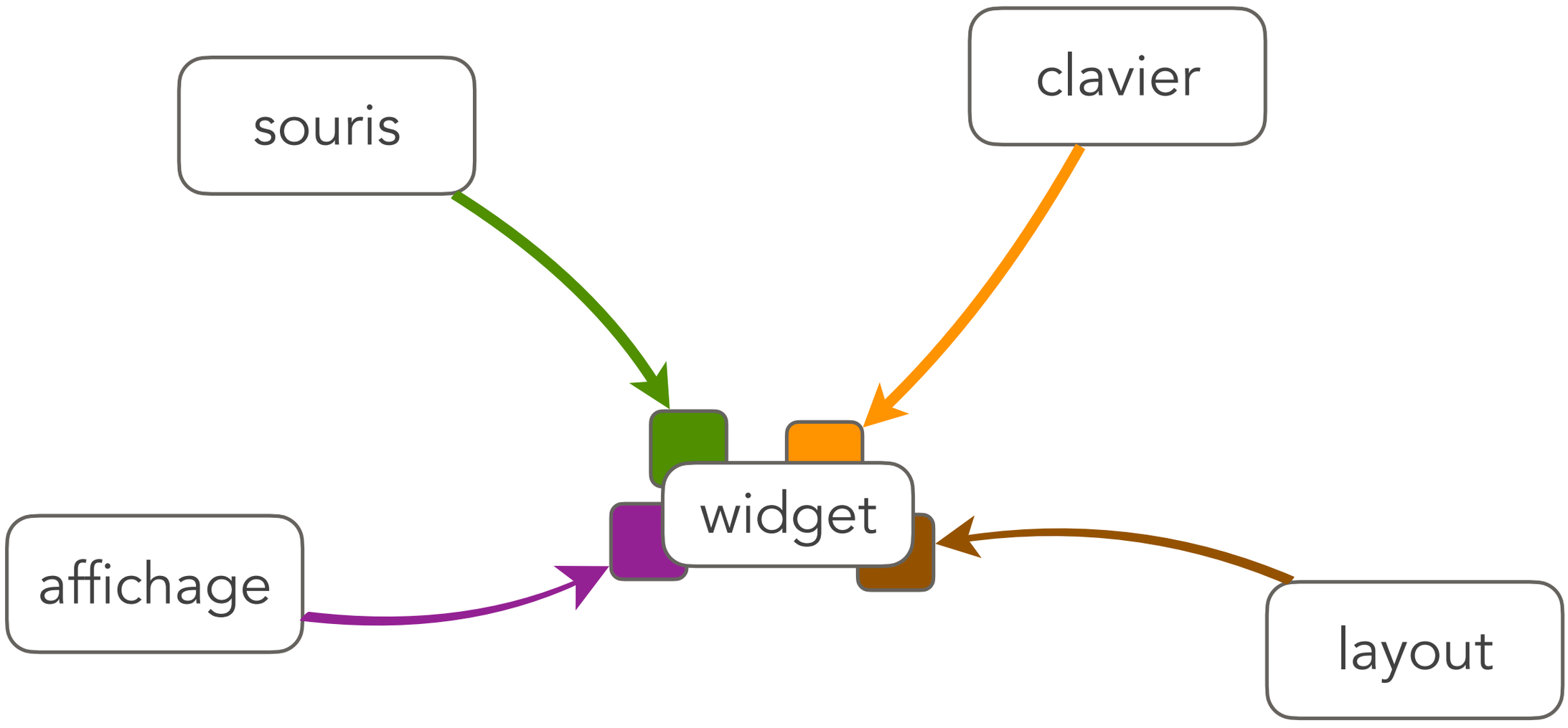
Périphériques

Mouse

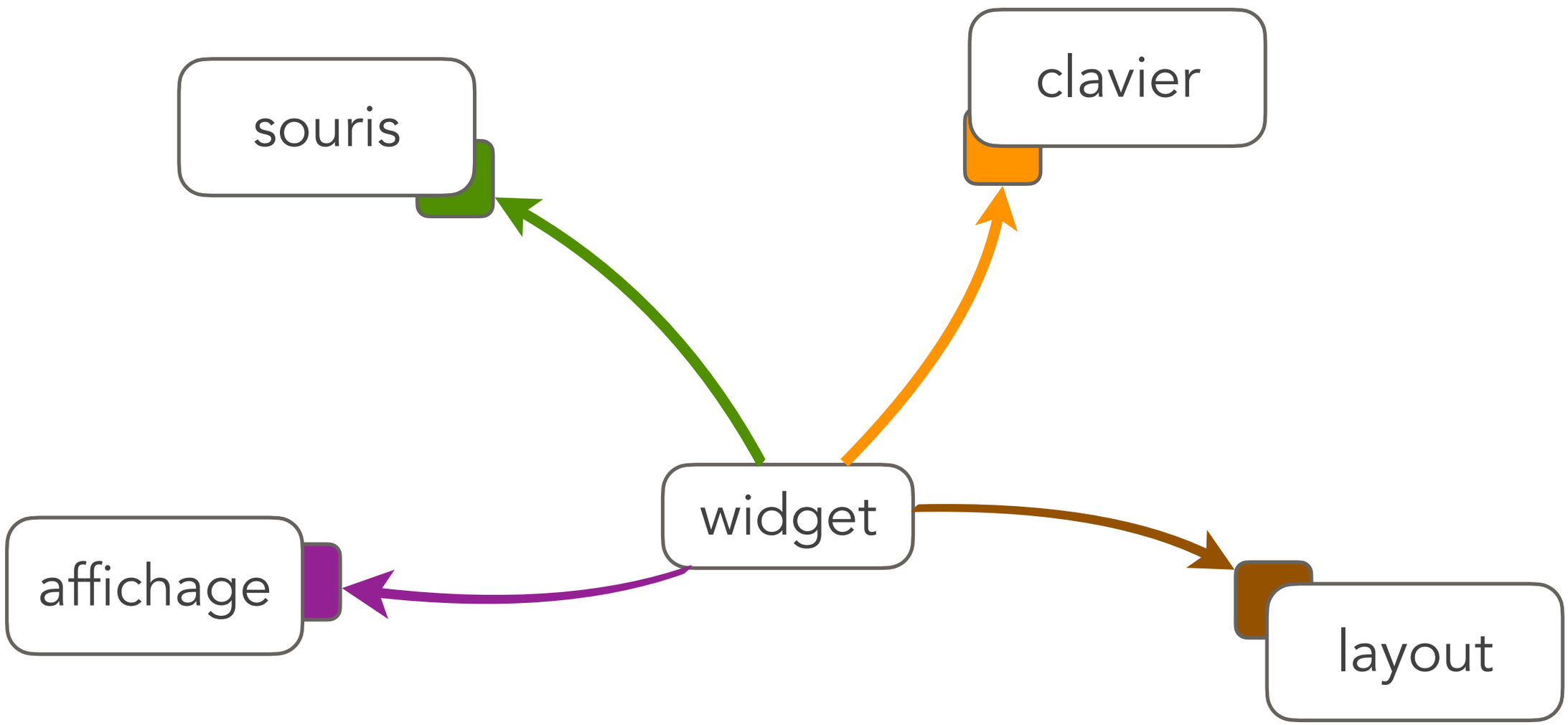
Keyboard

View

OOP



ECS



Contribution

Clarification et extension d'ECS pour la programmation d'IHM

Présentation d'une boîte à outils (Polyphony) et implémentation d'une application de dessin

Étude de 3 implémentations majeures et choix de conception

Perspectives

⇒ Utilisation d'un langage de programmation plus adapté

⇒ Développement d'interactions plus complexes

⇒ Mise à disposition d'autres utilisateurs

Merci de votre attention

Définitions

Entité – identifiant unique (souvent un simple entier) représentant chaque élément du programme

Composant – une donnée du programme (ex. position) appartenant à une entité particulière

Système – processus s'exécutant continuellement, sur les entités possédant un ensemble donné de Composants