

# Sintaks

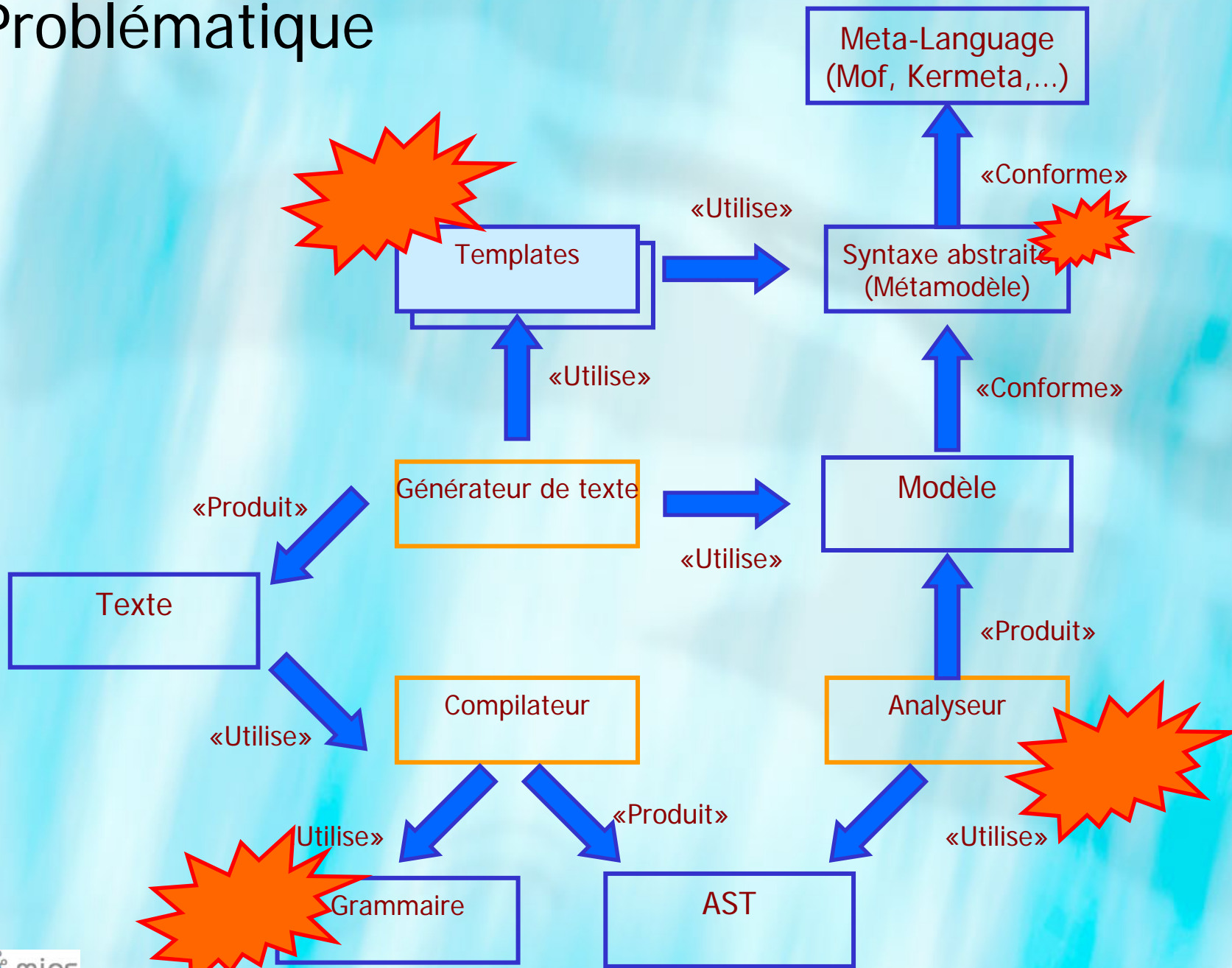
Outil de manipulation de syntaxe textuelle

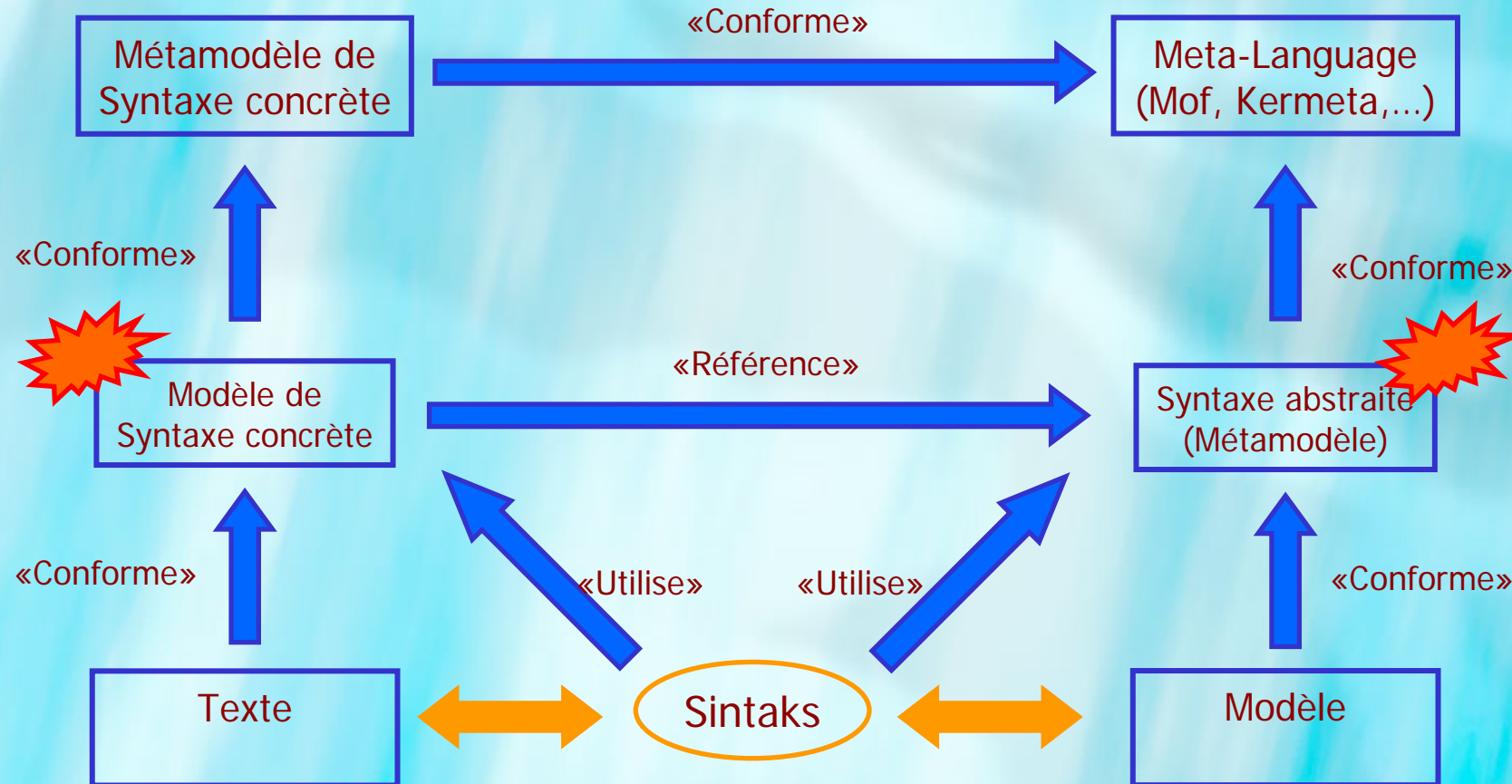
Michel Hassenforder

- Introduction
- Sintaks en résumé
- Exemple avec Media
- Sintaks et HUTN
- Sintaks et plus
- Conclusion

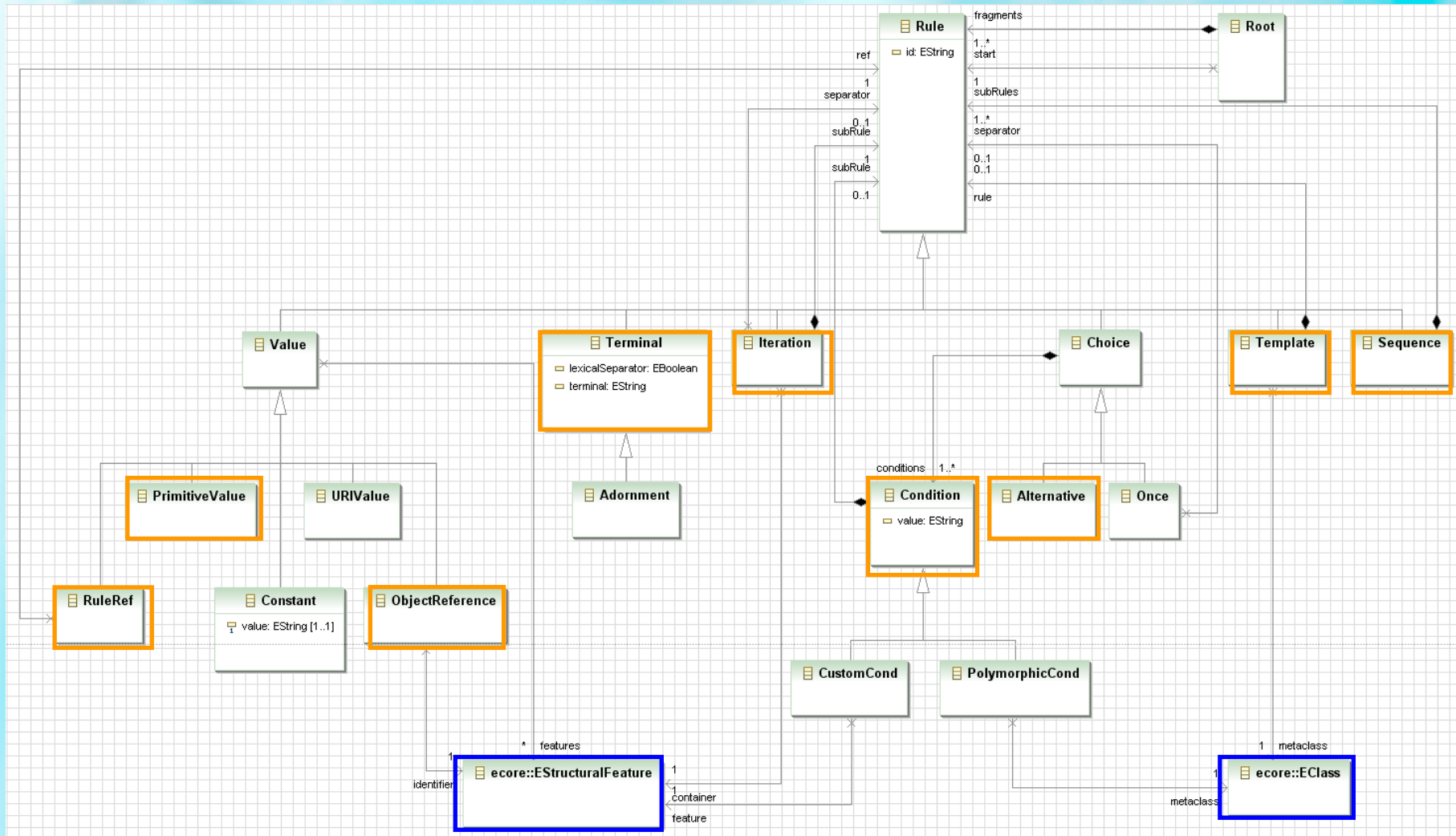
- En général :
  - Comment créer un modèle conforme à un métamodèle
  - Editeurs générés
  - Editeur réflexif
  - GMF
  - Etc..
- Dans le cadre de la création/lecture de programme conforme à un métamodèle ?

# Problématique





Transformation  
Bidirectionnelle



Les classes essentielles

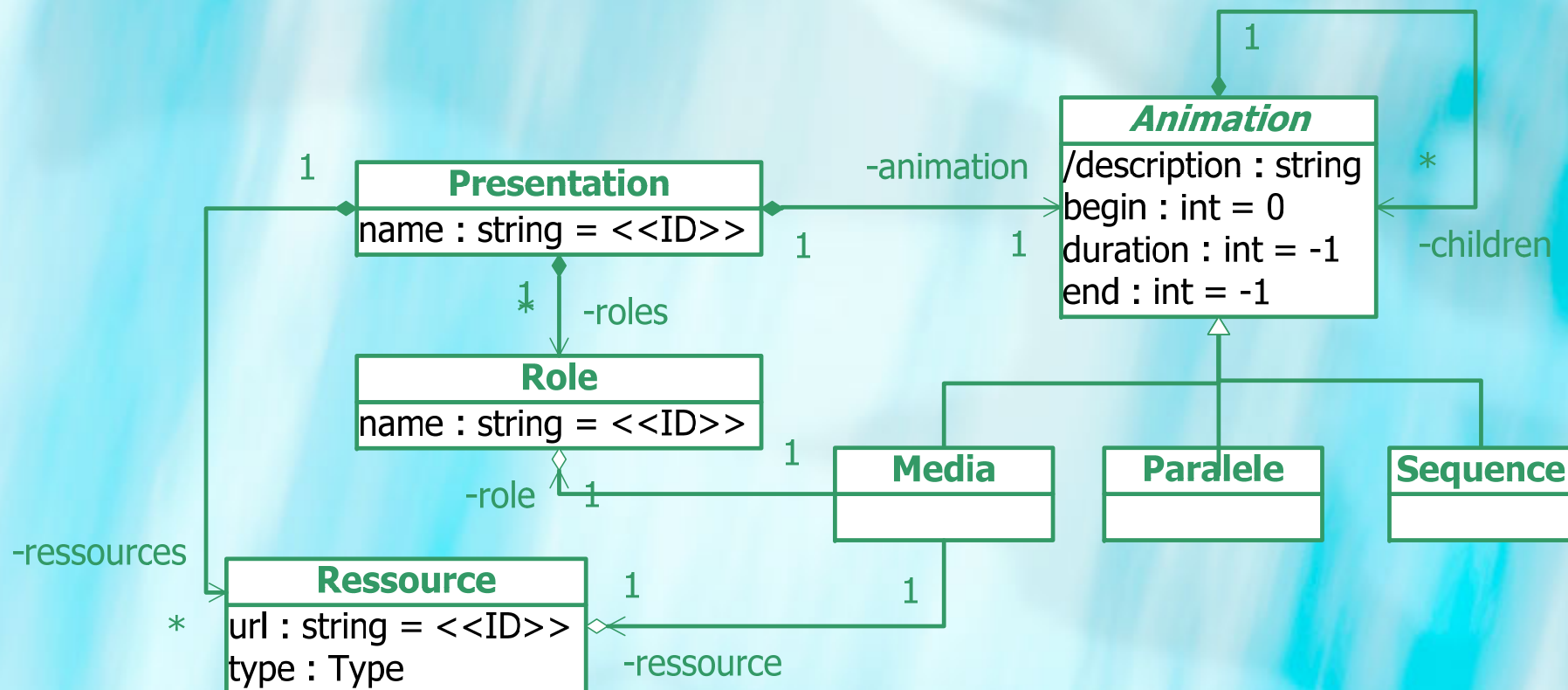


Les ponts

- **Sequence**  
Une collection ordonnée de règles
- **Template**  
Spécifie la classe à utiliser du métamodèle cible  
Spécifie la règle à appliquer
- **Terminal**  
Un texte connu et fixe lors de la modélisation  
Un symbole terminal dans le grammarware
- **PrimitiveValue**  
Une portion de texte qui provient d'une ou de plusieurs propriétés  
Assure la manipulation des types de base : Integer, String, Real, Boolean, ...
- **ObjectReference**  
Permet de manipuler les propriétés qui référencent un autre objet en utilisant une de ses propriétés comme clé (unique)
- **RuleRef**  
Permet l'invocation d'une règle typiquement un fragment possédé par Root  
Ce concept permet le partage de règles

- Iteration  
Permet de gérer une propriété de type container  
Assure la gestion d'un éventuel séparateur
- Alternative  
La liste de conditions est exploitée une fois pour en sélectionner la plus pertinente
- CustomCondition  
Permet de décrire la valeur qu'une propriété doit prendre
- PolymorphicCondition  
Permet de décrire la classe qu'il faut tester

- Métamodèle d'une animation multimédia



# Media

- ◆ Presentation simple presentation
  - ◆ Parallel [ duration 12]
    - ◆ Media audio:music.aac [ no timing ]
    - ◆ Sequence sequence [ from 1 duration 10]
      - ◆ Media audio:chimes.wav [ duration 1]
        - ◆ Parallel [ duration 8]
          - ◆ Media video:video1.3gp [ from 1 to end]
          - ◆ Media video:video2.3gp [ from 1 to end]
        - ◆ Media image:picture.jpg [ duration 2]
        - ◆ Media audio:ringout.wav [ no timing ]
    - ◆ Resource video1.3gp
    - ◆ Resource video2.3gp
    - ◆ Resource ringout.wav
    - ◆ Resource chimes.wav
    - ◆ Resource picture.jpg
    - ◆ Resource music.aac
    - ◆ Role LeftScreen
    - ◆ Role Effects
    - ◆ Role RightScreen
    - ◆ Role timer
    - ◆ Role Music

```
Presentation "simple presentation" {  
  Parallel ( duration = 12 ){  
    Media () {  
      Resource = audio music.aac ;  
      Roles = Music ;  
    }  
    Sequence ( begin = 1 duration = 10 ){  
      Media ( duration = 1 ){  
        Resource = audio chimes.wav ;  
        Roles = Effects ;  
      }  
      Parallel ( duration = 8 ){  
        Media ( begin = 1 ){  
          Resource = video video1.3gp ;  
          Roles = LeftScreen ;  
        }  
        Media ( begin = 1 ){  
          Resource = video video2.3gp ;  
          Roles = RightScreen ;  
        }  
      }  
      Media ( duration = 2 ){  
        Resource = image picture.jpg ;  
        Roles = RightScreen , LeftScreen ;  
      }  
      Media () {  
        Resource = audio ringout.wav ;  
        Roles = Effects ;  
      }  
    }  
  }  
}
```

- Problème

On a un Métamodèle (☺ ... ecore ...)

Sintaks est difficile d'accès ☹

On a peur de se perdre, difficile de démarrer ☹

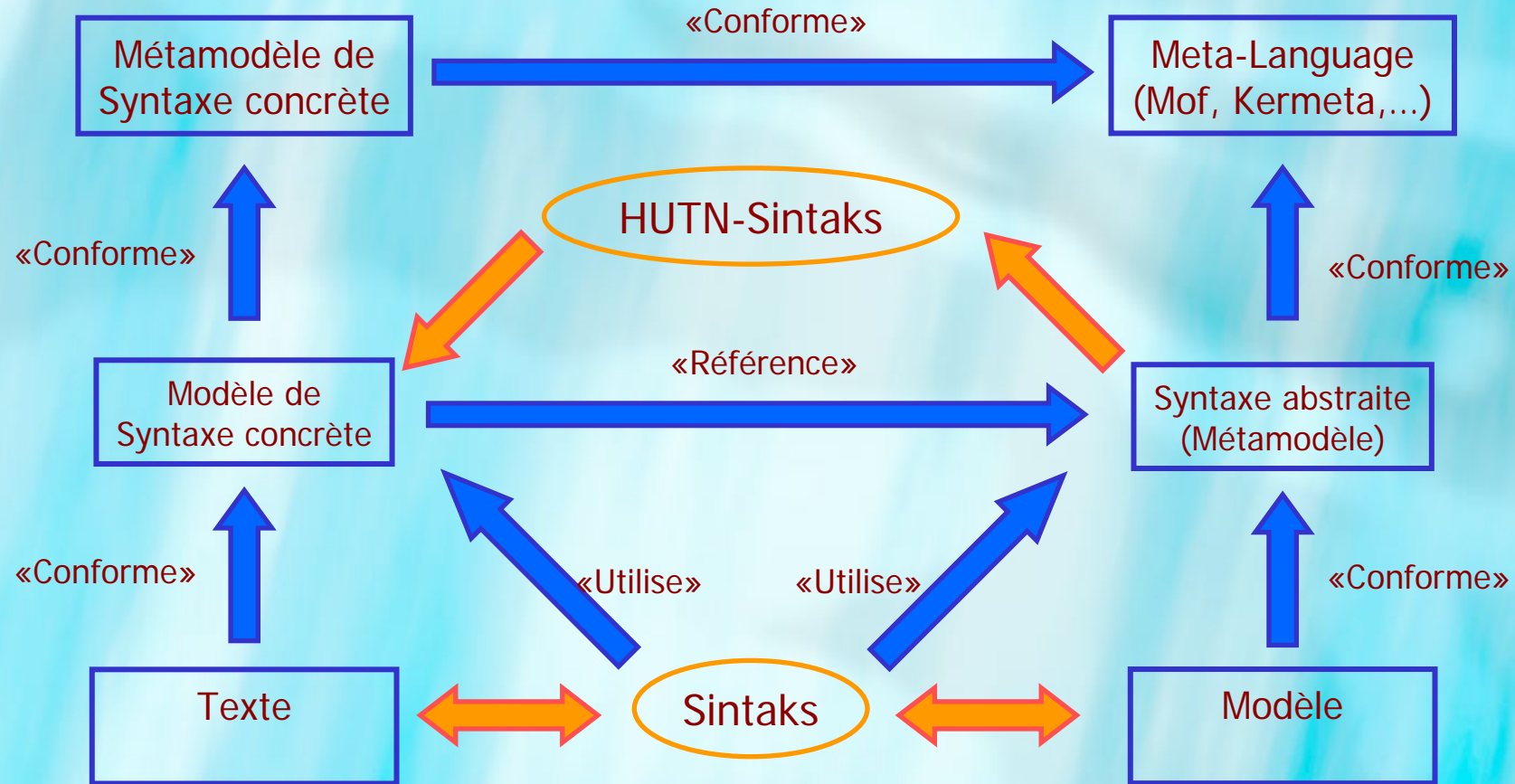
On veut une syntaxe robuste pour commencer

Puis on peut bidouiller autour et faire converger

HUTN est candidat potentiel

Mais trop gr(a/o)s pour juste du texte ☺

Petites simplifications



- HUTN pour sintaks

Pour une **classe mère abstraite** on génère automatiquement les alternatives pour propager l'analyse vers les classe filles

```
▲ x|y Alternative Animation
  ▲ PC Polymorphic Condition Sequence == equals
    f() Rule Ref ==> Sequence
  ▲ PC Polymorphic Condition Parallel == equals
    f() Rule Ref ==> Parallel
  ▲ PC Polymorphic Condition Media == equals
    f() Rule Ref ==> Media
```

Pour les **propriétés partagées** (d'une classe mère) on génère des règles de partage de syntaxe concrète

```
▲ {...} Sequence Animation_children
  ▲ x* Iteration : children
    ▲ {...} Sequence
      f() Rule Ref ==> Animation
      ⊕ Adornment "eoln"
```

Sequence

```
f() Rule Ref ==> Animation_children
```

Parallel

```
f() Rule Ref ==> Animation_children
```

- HUTN pour sintaks

Pour une **classe concrète** on peut utiliser l'un de ces schémas selon que les blocs adjectif et/ou propriété sont vides.

«nom» id ;

«nom» id '(' adjectif+ ')' ;

«nom» id '{' propriété+ '}'

«nom» id '(' adjectif+ ')' '{' propriété+ '}'

nom : le nom de la classe

id : un identifiant significatif de la classe (attribut de type ID)

adjectif : un attribut a multiplicité 0 ou 1

propriété : une propriété quelconque (le reste)

- HUTN pour sintaks

**Adjectif** est décrit directement avec sa valeur sans préciser à quel attribut il se réfère. Cela permet de faire des :

class A (static virtual private) ;

Contraintes : la valeur et la séquence doivent être non ambiguës.

**Propriété** est décrite selon sa nature et sa multiplicité avec un schéma plus complexe

x = 100;

pour un attribut

v : [ 10, 20, 30 ]

pour une collection

Media { type="audio"; url="music.aac"; }

pour un container

Les propriétés **dérivées/non modifiables/...** sont ... oubliées

Pour une référence

Génère des ObjectRef sur le ID de la classe cible

Pour des références bidirectionnelles

génère le container, sinon génère les deux ... faudra adapter

- Hutn pour sintaks :
  - Piloté par sintaks (template)
  - Pour du sintaks
  - Accompagné par un moteur spécifique
- Sous réserve de savoir écrire un template
  - Il est possible de l'adapter complètement ...
- Sintaks travaille sur ecore
- Sintaks est en ecore
- Sintaks peut être décrit avec une syntaxe concrète
- Ecore peut être HUTNifié
  - M'enfin s'ils renseignent l'attribut ID ☹ correctement

- Sintaks est disponible @ INRIA/triskell  
<http://www.kermeta.org/sintaks>
- On peut jouer avec des MM...
- Maintenant 😊
- Après 😊
- Plus tard 😞
- Demain 😞 😞
- Dans la démo  
Média, si il y a du temps TinyJava