M2 LITL - Programmation pour le TAL
Corrigé de la modélisation *dépendances syntaxiques*

Franck Sajous - CLLE-ERSS

Ce document est disponible à l'adresse : `http://fsajous.free.fr/`

1 Multiplicités

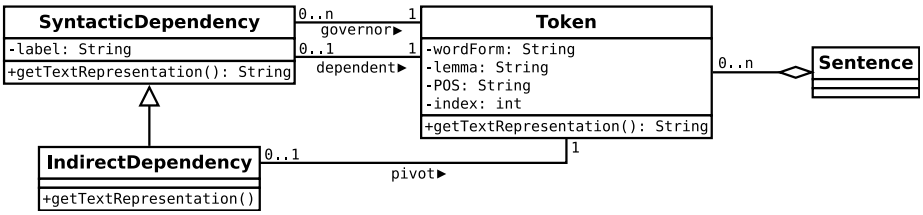

FIGURE 1 – Multiplicités pour les dépendances syntaxiques

— une phrase est composée d'un ou plusieurs (1..n) mots. On considère néanmoins que lors de la lecture d'une ligne vide, l'outil peut générer une phrase vide. On peut également penser à l'implémentation et donner la possibilité de créer une phrase initialement vide.

— un dépendance syntaxique *donnée* fait intervenir (a pour) un et un seul gouveneur, un et un seul dépendant (→ les deux multiplicités **1** côté Token).

— Un token (un article, par exemple) peut ne gouverner aucun autre token. Un token peut gouverner un ou plusieurs autres tokens, e.g. un verbe peut gouverner son sujet et un objet. (→ multiplicité **0..n** de la relation *governor*.)
Attention : si cette formulation paraît naturelle quand on pense à la syntaxe (un mot est dépendent de/gouverne un autre mot), du point de vue de la réflexion sur un diagramme de classes, elle convient plus au diagramme de la section 2 (relation Token/Token). Ici, il convient de réfléchir à la relation Token/SyntacticDependency : considérant **un** token donné, de combien de dépendance(s) syntaxique(s) (d'instance(s) de la classe SyntacticDependency) peut-il (au maximum)/doit-il (au minimum) être le gouverneur/le dépendant ?

— Tout token a un gouverneur (i.e. est gouverné, i.e. est un dépendant)... sauf la racine de la phrase. Il y a donc un token qui n'a pas de gouverneur/qui n'est pas un dépendant. Un token ne peut avoir plusieurs

gouverneurs (i.e. intervenir dans plus d'une relatation de dépendance comme dépendant). → multiplicité **0..1** de la relation *governor*.
— Un token donné peut être ou ne pas être (...) le pivot d'une relation de dépendance. Pour une relation de dépendance indirecte, il y a un et un seul pivot.

2 Relation réflexive

Une première remarque concerne le nom de la relation (*governor*) : il serait plus adapté comme nom de rôle (le rôle réciproque serait *dependent*). Si on souhaite garder un nom de relation (orientée), on peut choisir *governorOf* ou *governs*. Les trois représentations de la figure 2 sont équivalentes.

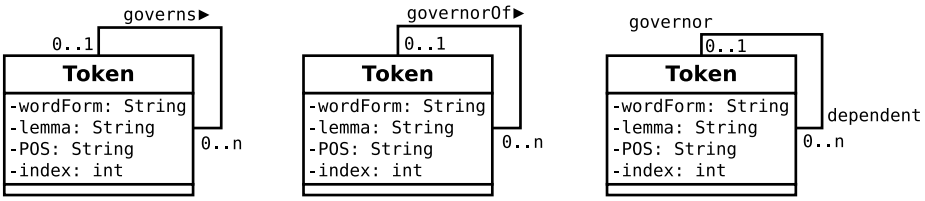

FIGURE 2 – Relation réflexive token → token

Remplacer le nom de la relation par la relation réciproque (gouverneur-dépendant, employeur-salarié, etc.) implique l'inversion de l'orientation de la relation et des multiplicités (cf. fig. 3).

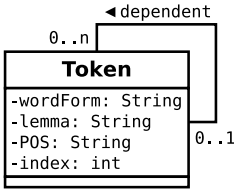

FIGURE 3 – Relation réflexive token → token

La pertinence de cette modélisation est discutable : il faut de toutes façons maintenir une (ou deux) classes pour stocker le type (label) de dépendance et éventuellement le pivot.

3 Diagramme d'objets

Illustrer si possible les différents scénarios, les différentes multiplicités discutées en section 1 et les cas particuliers :
— une dépendance syntaxique « simple » et une dépendance syntaxique indirecte (impliquant deux dépendances de Talismane, *via* un « pivot ») ;
— un token qui est à la fois gouverneur et dépendant : tok8 (*type*) ;
— un token qui est le gouverneur d'au moins deux dépendances syntaxiques : tok6 (*constituer*).

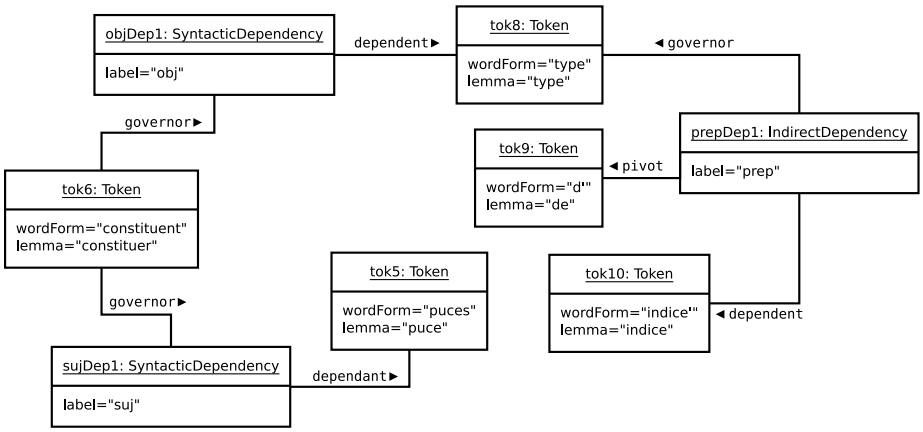

FIGURE 4 – Diagramme d'objets pour les relations de dépendance syntaxique