La présentation du système TIMINF

1) Introduction

Nous présentons, dans ce chapitre notre projet d'inférence temporelle, nommé TIMINF. Ce projet a pour but de développer et d'évaluer l'apport de l'inférence temporelle dans la reconnaissance de l'inférence textuelle.

L'un des principaux défis de ce type de système est de permettre aux systèmes d'inférences textuelles, d'ouvrir un voile sur l'inférence temporelle et d'explorer cette nouvelle approche. Dans ce cadre, l'objectif de TIMINF est de définir ce que devrait être un système d'inférence textuelle intégrant l'aspect temporel dans son fonctionnement, qui tient en compte la relation entre expression temporelle et relation entre les évènements dans la déduction de l'inférence textuelle.

Nous allons montrer tout au long de ce document comment nous avons concrétisé cet objectif. Nous décrivons alors les principaux modules constituant le système.

2) Architecture informatique de TIMINF

L'architecture générale de TIMINF, telle que déduite de l'analyse du corpus présenté au chapitre précédant, est illustrée dans la figure 4.1 suivante. Cette dernière s'articule autour de trois étapes essentielles qui sont :

- Le prétraitement qui permet de repérer les données temporelles et les composants syntaxiques de la paire de texte (T, H).
- L'inférence textuelle qui contient les modules de test d'inférence textuelle et du balisage des expressions temporelles.
- L'inférence temporelle qui contient les moteurs d'inférence et les règles d'inférences.

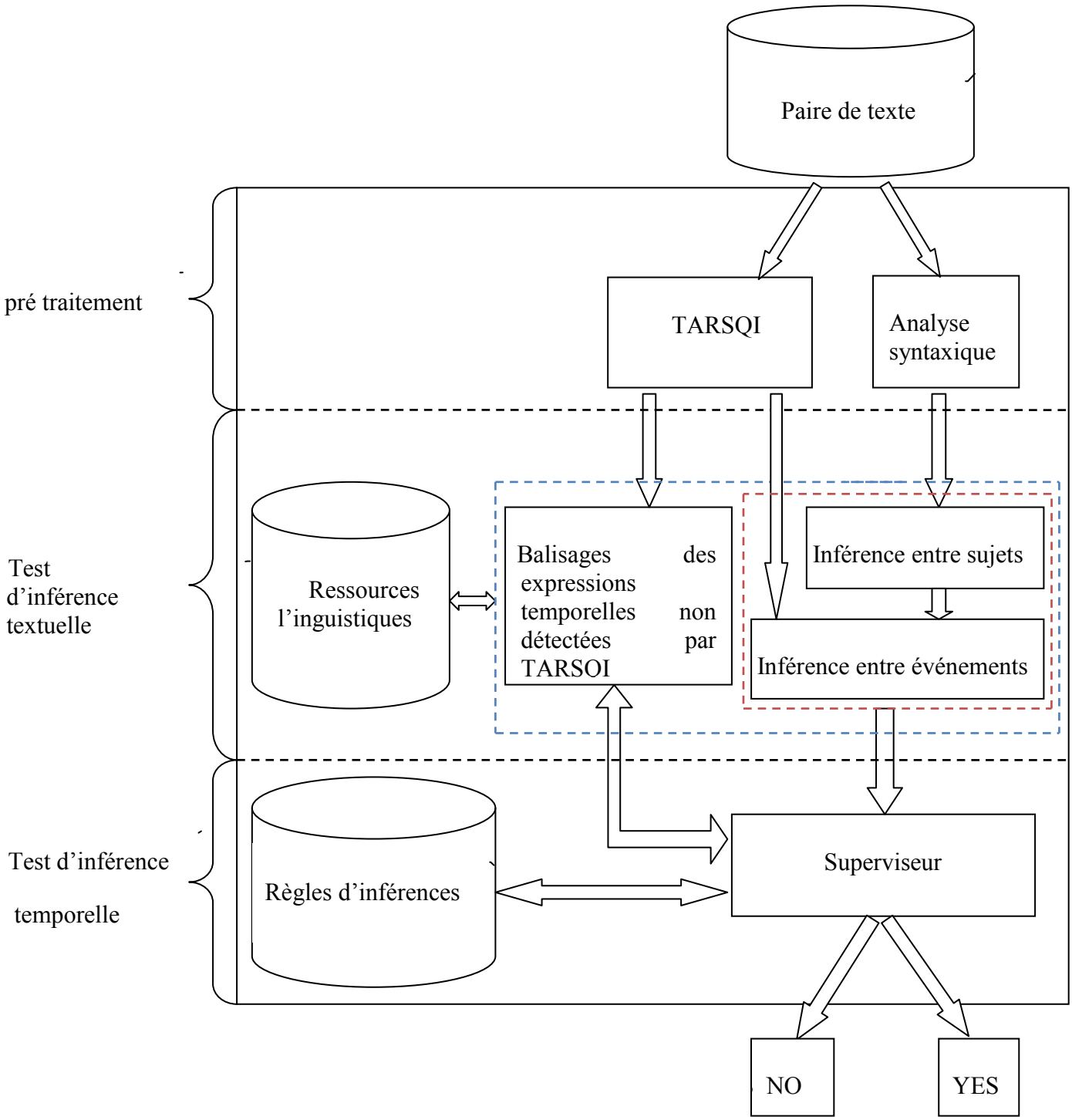

Figure 4.1 : Architecture du système TIMINF

Dans ce qui suit nous présentons les différents modules constituants le système TIMINF.

2.1) Le prétraitement

Le prétraitement est effectué par les deux modules TARSQI et LINK parseur. Ces deux modules s'exécutent en parallèle et nous permettent respectivement de repérer les données temporelles et les composants syntaxiques de la paire de texte (T, H). Nous détaillerons dans ce qui suit les deux modules et leurs utilisations dans notre système.

2.1.1) Le projet TARSQI

TARSQI est un outil permettant d'organiser des textes en langages naturels en fonction de leurs caractéristiques temporelles (Pustejovsky et al., 2003). Son objectif est d'annoter les données temporelles dans un texte en langage naturel, d'extraire des données temporelles à partir de textes et d'effectuer des raisonnements sur les données temporelles (http://www.timeml.org). Afin de répondre à ces différents objectifs, le module TARSQI utilise les balises TimeML pour marquer les expressions temporelles, les événements, les relations temporelles et les Subordinations syntaxiques des événements.
Le système TARSQI est mis en place comme une cascade de modules successivement ajoutés.
L'architecture du système est définie dans le schéma ci-dessous.

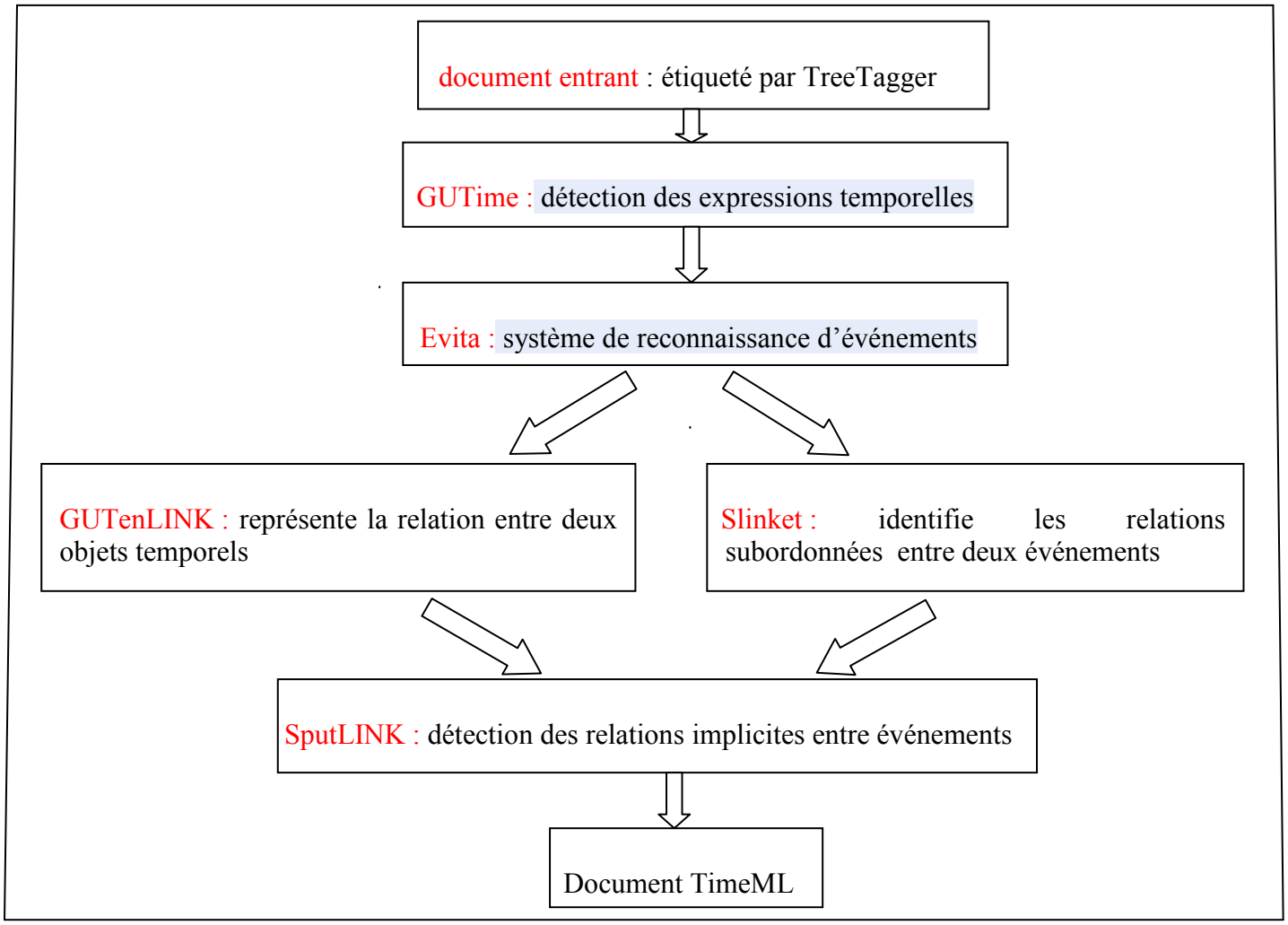

Figure 4.2 : Architecture du module TARSQI

Le module **TARSQI** doit avoir comme entrée des documents prétraités syntaxiquement. Pour cela, les concepteurs de **TARSQI** ont choisi d'utiliser une analyse morphosyntaxique avec le module **TreeTagger**.

Dans ce qui suit nous allons décrire le module **TreeTagger.**

2.1.1.1) TreeTagger

C'est un système d'étiquetage automatique des catégories grammaticales des mots avec lemmatisation et tokenisation (Helmut Schmid, 1994) (www.ims.uni-stuttgart.de/projekte/corplex/**TreeTagger**/).

Le module Treetagger a comme entrée un texte brut et il admet deux types de sorties :

A) Une sortie en forme de tableau

Comme il est montré dans l'exemple suivant (figure 4.3), le mode de sortie est un tableau représentant l'étiquetage des mots dans la phrase.

Entrée :

```
Le TreeTagger est facile à utiliser.
```

Sortie:

Mot	**POS**	**Lemme**
Le	DT	La
TreeTagger	NP	TreeTagger
Est	VBZ	Être
Facile	JJ	Facile
À	D'	À
Utiliser	VB	Utiliser

Figure 4.3 : Sortie en format tableau de TreeTagger

Sachant que **:**

Mot : représente le mot étiqueté.

POS : représente la catégorie grammaticale du mot par exemple (VB pour verbe, DT pour un déterminant…..).

Lemme : représente la lemmatisation du mot.

B) Sortie format XML

Avec La sortie format XML, chaque mot est tagué avec les balises de TreeTagger.

Exemple d'entrée, sortie TreeTagger :

Entrée:

He also slept on Friday night.

Sortie:

```
<BODY>

<TEXT>

<s> <NG><lex pos="PP">He</lex></NG> <lex pos="RB">also</lex> <VG <lex
pos="VBD">slept</lex></VG> <lex pos="IN">on</lex> <NG> <lex pos="NNP"
>Friday</lex> <lex pos="NN">night</lex> </NG> <lex pos=".">.</lex> </s>

</TEXT>

</BODY>
```

Figure 4.4 : Sortie en format XML de TreeTagger

Les balises utilisées par TreeTagger sont :

- <BODY> contient le corps du document.
- <TEXT> contient le texte.
- Les phrases doivent être marquées d'un <s>.
- Le groupe nominal est balisé avec <NG> et le groupe verbal avec <VG>.
- chaque mot dans la phrase est balisé par <LEX>.

Les attributs utilisés par TreeTagger sont :

- **Stem :** représente la lemmatisation du mot qui est balisé.
- **Pos :** donne la catégorie grammaticale du mot balisé. (**DT** pour déterminant-nom, **PP** pour une préposition…). pour en savoir plus sur les différents symboles utilisés par Treetagger pour étiquetter les différentes catégories grammaticales, toutes les définitions des symboles sont disponibles sur le site (www.ims.uni-stuttgart.de/projekte/corplex/**TreeTagger**/).

2.1.1.2) GUTime

L'étiqueteur GUTime, développé à l'Université de Georgetown, utilise TIMEX3 tag pour représenter les expressions temporelles, telles que : les dates, les heures, les durées, etc (Mani et Wilson, 2000).

Il existe 3 types d'informations temporelles détectées par TIMEX3.

DATE : c'est-à-dire les années, les mois et les jours.

Exemple:

*USA were touched by terrorism in **September 11, 2001**.*

TIME : c'est-à-dire les heures de la journée.

Exemple:

*The building collapsed at **2 o'clock p.m.***

DURATION : représente un intervalle de temps entre deux dates.

Exemple:

*The end of the second world war happened **between 1940 and 1950**.*

Un exemple de sortie du module GUTime est montré ci-dessous :

```
In Washington <TIMEX3 tid="t1" TYPE="DATE" temporalFunction="true"
valueFromFunction="tf1"     anchorTimeID="t0">today</TIMEX3>,     the
Federal Aviation Administration released air traffic control tapes
from the night the TWA Flight eight hundred went down.
```

Figure 4.5: Sortie du module GUTime

Les attributs de TIMEX3 dans l'exemple sont :

Tid : donne l'identifiant de l'expression temporelle, pour chaque expression tagger par TIMEX a son propre identifiant.
Type : chaque TIMEX est assigné à ces différents types {DATE, TIME, DURATION}.
TemporalFunction : c'est un attribut qui retourne si la date est précise dans le temps ou pas.

Exemple:
Next Tuesday → TemporalFunction= true.
September 11,2001→ TemporalFunction= false.

AnchorTimeID : s'il y a un ancrage temporel de l'expression temporelle identifiée par **Tid** avec une autre expression temporelle, **AnchorTimeID** donne son identifiant.

2.1.1.3) Evita

Evita est un système de reconnaissance d'événements, pour cela le module utilise deux balises de TIMEML (EVENT et MAKEINTANCE) qui sont décrites ci-dessous :

A) EVENT

EVENT est utilisé pour annoter les événements dans un texte, syntaxiquement, les évènements sont généralement des verbes, mais un nom peut aussi être utilisé pour dénoter un événement.
Les différentes classes d'événements qui sont détectées sont représentées ci-dessous.

- **occurence :** la plupart des événements font partie de cette classe. Ils décrivent ce qui se produit dans le monde.
- **state :** les états décrivant les circonstances dans lesquelles un événement a lieu et dont l'état peut être modifié ; et les états introduits par les i-action, i-state et reporting.
- **Reporting :** description de l'action d'une personne par un acte narratif.
- **i-action :** une action intentionnelle introduisant un autre événement, comme un essai, une enquête, un rapport, un ordre, une demande, une promesse, une nomination.
- **i-state :** similaire à i-action mais pour identifier un état tel que penser, ressentir. suspecter, douter, vouloir, désirer, détester, être prêt, être capable.
- **aspectual :** un événement débutant, terminant ou continuant une action.
- **Perception :** constatation physique d'un événement telle qu'entendre ou voir l'action.

B) MAKEINSTANCE

MAKEINSTANCE est une réalisation de lien, il indique les différentes instances d'un événement donné.

Dans l'annotation, les <EVENT> ne participe jamais à une relation, c'est la réalisation (<MAKEINSTANCE>) de l'événement qui y participe et chaque EVENT introduit au moins un correspondant MAKEINSTANCE.

Un exemple de sortie du module Evita est montré ci-dessous:

```
In Washington today, the Federal Aviation Administration <EVENT eid="e1"
class="OCCURRENCE">released</EVENT> air traffic control tapes from the
night the TWA Flight eight hundred <EVENT eid="e2"
class="OCCURRENCE">went</EVENT> down.

<MAKEINSTANCE eventID="e1" eiid="ei1" pos="VERB" tense="PAST"
aspect="NONE"/>
<MAKEINSTANCE eventID="e2" eiid="ei2" pos="VERB" tense="PAST"
aspect="NONE"/>
```

Figure 4.6 : Sortie du module Evita

Les attributs de EVENT dans l'exemple sont :
Eid : donne l'identifiant de l'évènement, pour chaque évènement tagger par EVENT a son propre identifiant.
Class : détermine la classe auquel appartient l'évènement.
Les attributs de MAKEINSTANCE dans l'exemple sont :
eventID : donne l'identifiant de l'évènement, pour chaque évènement tagger par EVENT a son propre identifiant.
Eiid : instance de l'événement trouvé dans le texte.
Pos : donne la catégorie grammaticale du mot balisé.
Tense : donne le temps de l'évènement si l'évènement est un verbe.

2.1.1.4) GutenLink

GutenLink est un module de TARSQI qui utilise les balises TLINK de TIMEML pour représenter la relation entre deux objets temporels, que ce soit deux événements, deux marqueurs temporels ou un marqueur temporel et un événement. Il y a quatorze types de relations identifiées par le module, bien que certaines soient simplement l'inverse d'autre :

- **before** et **after** spécifient qu'un objet temporel précède ou suit l'autre objet temporel de la relation ;
- **ibefore** et **iafter** spécifient qu'un objet temporel est immédiatement avant ou après un autre.
- **includes** et **is-included** spécifient qu'un objet temporel inclut ou est inclus dans un autre, p. *ex. John arrived in Montreal yesterday.*
- **during** spécifie que l'état ou l'événement se poursuit durant une période de temps, *p. ex. John taught for 90 minutes.*
- **during-inv** est l'inverse de la relation précédente.
- **simultaneous** spécifie que deux instances d'événements semblent coïncider dans le Temps.
- **identity** indique que deux objets temporels représentent le même événement.
- **begins** spécifie qu'un événement débute par l'objet temporel avec lequel il est lié.
- **begun-by** est l'inverse de begin, elle relie un objet temporel à un événement débutant par l'objet temporel.
- **ends** et **ended-by** sont similaires aux deux relations précédentes sauf qu'elles Spécifient la fin de l'événement.

Un exemple de sortie du module **GutenLink** est montré ci-dessous :

```
In Washington <TIMEX3 tid="t1" TYPE="DATE" VAL="PRESENT_REF"
temporalFunction="true" valueFromFunction="tf1"
anchorTimeID="t0">today</TIMEX3>, the Federal Aviation Administration
<EVENT eid="e1" class="OCCURRENCE">released</EVENT> air traffic control
tapes from the night the TWA Flight eight hundred <EVENT eid="e2"
class="OCCURRENCE">went</EVENT> down. There's nothing new on why the plane
<EVENT eid="e3" class="OCCURRENCE">exploded</EVENT>, but you <EVENT
eid="e4" class="OCCURRENCE">cannot</EVENT> <EVENT eid="e5"
class="OCCURRENCE">miss</EVENT> the moment. ABC's Lisa Stark <EVENT
eid="e6" class="OCCURRENCE">has</EVENT> more.

<MAKEINSTANCE eventID="e1" pos="VERB" eiid="ei1" tense="PAST"
aspect="NONE"/>
<MAKEINSTANCE eventID="e2" pos="VERB" eiid="ei2" tense="PAST"
aspect="NONE"/>
<MAKEINSTANCE eventID="e3" pos="VERB" eiid="ei3" tense="PAST"
aspect="NONE"/>
<MAKEINSTANCE eventID="e4" pos="VERB" eiid="ei4" tense="PRESENT"
aspect="NONE"/>
<MAKEINSTANCE eventID="e5" pos="VERB" eiid="ei5" tense="INFINITIVE"
aspect="NONE"/>
<MAKEINSTANCE eventID="e6" pos="NONE" eiid="ei6" tense="PRESENT"
aspect="NONE"/>

<TLINK eventInstanceID="ei1" relatedToTime="t1" relType="IS_INCLUDED"
rule="2-1"/>
<TLINK eventInstanceID="ei2" relatedToTime="t1" relType="IS_INCLUDED"
rule="2-1"/>
<TLINK eventInstanceID="ei1" relatedToEventInstance="ei3" relType="BEFORE"
rule="3-19"/>
<TLINK eventInstanceID="ei3" relatedToEventInstance="ei4" relType="BEFORE"
rule="6-1"/>
<TLINK eventInstanceID="ei3" relatedToEventInstance="ei6" relType="BEFORE"
rule="3-23"/>
```

Figure 4.7 : Sortie du module GutenLink

Les attributs de TLINK dans l'exemple sont :

eventInstanceID : donne l'identifiant de l'évènement.
relatedToTime : donne l'identifiant de l'expression temporelle.
relType : donne la relation temporelle existant entre les l'expressions temporelles, ils utilisent pour cela les relations d'Allen.

2.1.1.5) Slinket

Les liens subordonnants <SLINK> identifient les relations entre deux événements. Ils sont habituellement introduits par des verbes modaux qui impliquent une confirmation.

Les liens subordonnants sont définis selon six types de relations qui interagissent avec les classes d'événements reporting, i-state et i-action (modal introduit la possibilité d'un événement, *p. ex. John promised Mary to buy some beer)*.

Les différentes classes d'événements qui sont détectées sont représentées ci-dessous.

- **evidential** introduit la perception ou le compte-rendu de l'événement, *p. ex. Johnsaid he bought a pack of beer.*
- **neg-evidential** introduit la perception ou rapporte que l'événement ne s'est pas réalisé, *p. ex. John denied he bought beers*
- factive est une action qui implique ou présuppose qu'un événement a déjà eu lieu, *p. ex. John managed to leave the party.*
- **counter-factive** est la négation de la relation précédente *p. ex. John forgot to buy beers.*
- **conditional** indique que la réalisation de l'action entraînera l'événement en relation.

Un exemple de sortie du module SLINKET est montré ci-dessous:

```
The Soviet Union <EVENT eid="e12" class="REPORTING">said</EVENT>
today it had <EVENT eid="e13" class="OCCURRENCE">sent</EVENT> an envoy to
the Middle East.

<MAKEINSTANCE eventID="e12" eiid="ei12" tense="PAST" aspect="NONE"
pos="VERB"/>
<MAKEINSTANCE eventID="e13" eiid="ei13" tense="PAST" aspect="PERFECTIVE"
pos="VERB"/>

<SLINK relType="EVIDENTIAL" eventInstanceID="ei12"
subordinatedEventInstance="ei13" "/>
```

Figure 4.8 : Sortie du module SLINKET

Les attributs de SLINK dans l'exemple sont :

eventInstanceID : c'est l'identifiant de l'évènement concerné par la relation de subordination.
subordinatedEventInstance : c'est l'identifiant de l'évènement subordonné.
relType : donne la relation temporelle existante entre entités.

2.1.1.6) SputLink

Le module SputLink effectue des inférences temporelles en tenant compte des relations temporelles déjà générées par les modules qui le précèdent, c'est-à-dire (GUTenLINK et Slinket) et génère de nouvelles relations temporelles.
SputLink est fondé sur l'algèbre d'intervalle fondé par James Allen's en 1983.
Allen réduit tous les évènements et expressions de temps à 13 intervalles de bases et identifie les relations entre les intervalles. Les informations temporelles dans un document sont représentées comme un graphe où les événements et les expressions temporelles forment les nœuds, les relations temporelles forment les arcs.

Exemple

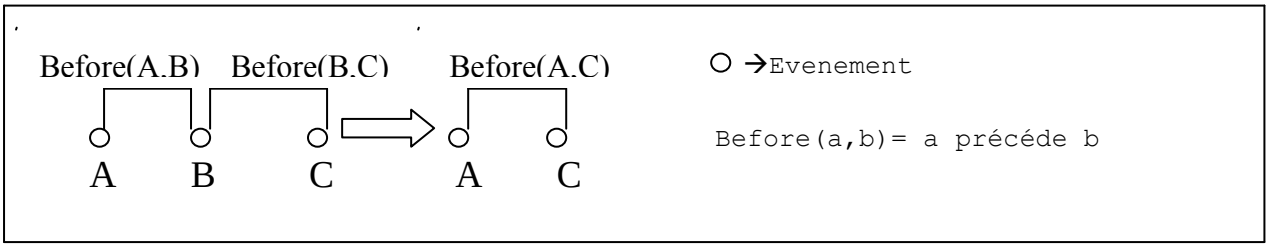

Figure 4.9: Inférence effectué par le module SputLINK

Ainsi, si A précède B et B précède C. des deux relations, on déduit que A précède C.

2.1.1.7) L'utilisation de TARSQI

Afin de permettre la portabilité du module TARSQI, les concepteurs ont proposé deux formats d'entrée possible à TARSQI qui sont décrits ci-dessous:

A) Format simple–xml

Avec ce format, l'analyse morphosyntaxique est incluse dans le module TARSQI. L'entrée est représentée par le format suivant:

Exemple d'entrée simple-xml.

```
<DOC>

<DOCID> Simple Test </DOCID>

<TEXT>

 In the afternoon, the building collapsed.

</TEXT>

</DOC>
```

Figure 4.10 : Entrée format simple-xml

Les balises de simpe_xml :

- <DOC> pour annoter le début et la fin du document.
- <DOCID> contient le type du document.
- <TEXT> contient le texte.

B) Format RTE3

Avec ce format, l'analyse morphosyntaxique n'est pas incluse dans le module TARSQI et nous avons comme entrée le format RTE3, qui est le résultat d'un prétraitement effectué par le groupe COGEX qui travaille sur le RTE. Le groupe a choisi de développer son propre prétraitement.

Nous présentons dans ce qui suit un exemple de sortie du format RTE3 :

Exemple :

```
<XML version="1.0" ?>

<pair length="short" task="IE" id="1">

<t><s>text1</s></t>

<br/><h><s><NG>

<HEAD><lex start="0" end="12" pos="NNP" stem="Le Beau Serge">Le Beau
Serge</lex> </HEAD></NG><VG><lex start="14" end="16" pos="VBD"
stem="be">was</lex>

<HEAD><lex start="18" end="25" pos="VBN" stem="direct">directed</lex>
</HEAD></VG>

<HEAD><lex start="27" end="28" pos="IN" stem="by">by</lex> </HEAD><NG>

<HEAD><lex start="30" end="36" pos="NNP" stem="Chabrol">Chabrol</lex>
</HEAD></NG><lex start="37" end="37" pos="." stem=".">.</lex> </s></h>

</pair>
```

Figure 4.11 : Sortie du module GutenLink

Les balises du format RTE3 sont :

- Les phrases doivent être marquées d'<s>.
- Les groupes nominaux sont balisés avec <NG> et les groupes verbaux avec <VG>.
- Les débuts de phrases sont marqués par des balises <HEAD>.
- <t> représente la premier phrase et <s> représente la deuxième phrase.
- <pair> représente la paire de phrases.

Les attributs :

- **Start :** représente la position du caractère de début de la chaine dans le texte.
- **End :** représente la position du caractère de fin de la chaine dans le texte.
- **Stem :** représente la lemmatisation du mot qui est balisé.

- **Pos :** donne la catégorie grammaticale du mot balisé.

2.1.1.8) L'intégration de TARSQI au système TIMINF

Dans notre système d'inférence, nous avons utilisé le format simple-xml au lieu de RTE3 car nous avons choisi d'utiliser le module TreeTagger pour l'analyse morphosyntaxique qui est intégré dans le module TARSQI dans le format simple-xml.

En plus de la détection des expressions temporelles, la phase de prétraitement intègre l'analyse syntaxique pour détecter la relation grammaticale entre les mots dans une phrase. Dans ce qui suit, nous allons présenter l'outil que nous avons choisi pour effectuer l'analyse syntaxique.

2.1.2) L'analyse syntaxique

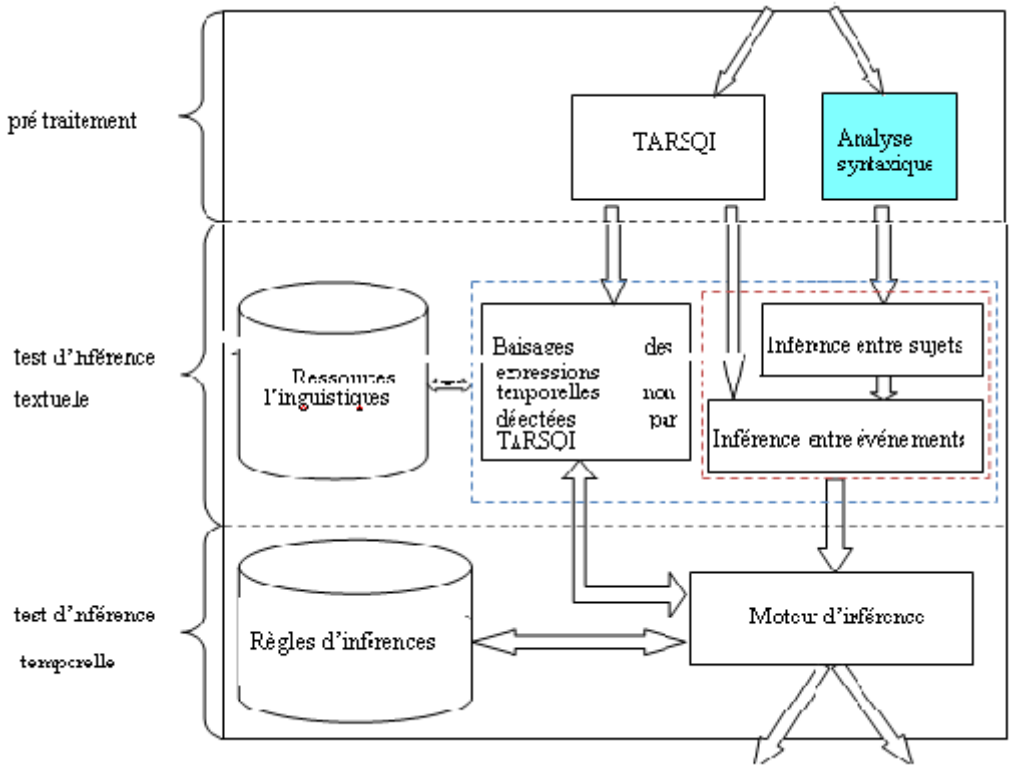

Figure 4.12 : L'analyse syntaxique

2.1.2.1) Présentation de link grammar parser

Nous avons utilisé le *Link Grammar Parser* (Sleator et Temperley, 1991) qui est un analyseur syntaxique de la langue anglaise, basé sur la dépendance syntaxique.
Partant d'une phrase fournie en entrée, cet analyseur produit un ou plusieurs graphes de dépendances, qui consistent en un ensemble de liens reliant des paires de mots.
Les **nœuds** du graphe sont les mots de la phrase. Certains d'entre eux ont un suffixe qui indique la partie du discours (nom, verbe, adjectif,adverbe, préposition, etc.).

Les **arcs étiquetés** relient les nœuds du graphe. Chaque étiquette précise un rôle grammatical (**D** pour déterminant-nom, **S** pour sujet-verbe…). Dans ce qui suit, nous montrons un exemple de sortie du parseur *Link Grammar Parser*.
Exemple :

Entrée:

```
Jone works very hard.
```

Sortie:

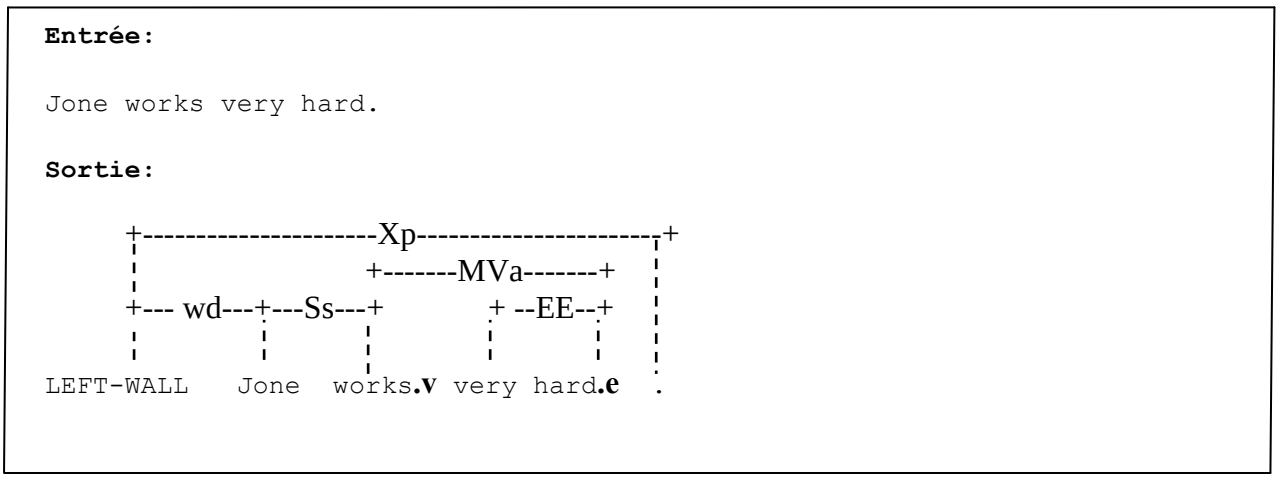

Figure 4.13 : Sortie du module Link Grammar Parser

Les définitions des différents liens représentés dans le graphe sont les suivantes :

EE→ adverbe se connecte à un autre adverbe.
Ss→connecte le sujet au verbe.
Xp→connecte le début et la fin de la phrase.
MVa→connecte le verbe à l'adverbe.
Wd→ le premier mot est un sujet.
.v→ verbe.
.e→ adverbe.
LEFT-WALL → détermine le début de la phrase.

Pour en savoir plus sur les différents symboles utilisés par *Link Grammar Parser* pour étiqueter les différents liens grammaticaux, toutes les définitions des symboles sont disponibles sur le lien suivant (www.**link**.cs.cmu.edu/**link**/).

2.1.2.2) L'intégration du link parser à notre système

Concrètement l'analyseur syntaxique nous a permis de détecter les sujets dans les deux segments de textes (T, H) et de les baliser avec nos propres balises comme il figure dans l'exemple suivant.

Exemple:

<pair id="28" value="TRUE" >
<s> ***<syntax type: sujet>****Poland****</syntax>*** *became a communistic state in 1945.</s><s>* ***<syntax type: sujet>****Poland****</syntax>*** *has become a communistic state since the invasion of Russians.</s>*

La balise **<syntaxe type: sujet>***sujet***</syntaxe>** est choisi pour baliser les sujets dans la paire(T, H).

Apres l'analyse syntaxique et le traitement par TARSQI, la paire de texte est prête à être soumise au test d'inférence textuelle qui a besoin des prétraitements effectués précédemment pour tester l'inférence textuelle. Dans ce qui suit nous décrivons les différents constituants de la phase de test d'inférence textuelle.

2.2) Les tests d'inférences textuelles

La deuxième phase s'articule autour de deux modules. Le premier permet de tester l'inférence textuelle pour savoir s'il y a une inférence textuelle ou pas et le deuxième module permet de détecter les expressions temporelles non détectées par TARSQI. Les deux modules exploitent des ressources linguistiques.

Dans ce qui suit nous allons présenter les deux modules et les ressources linguistiques utilisées :

2.2.1) Les tests d'inférences entre événements et entre sujets

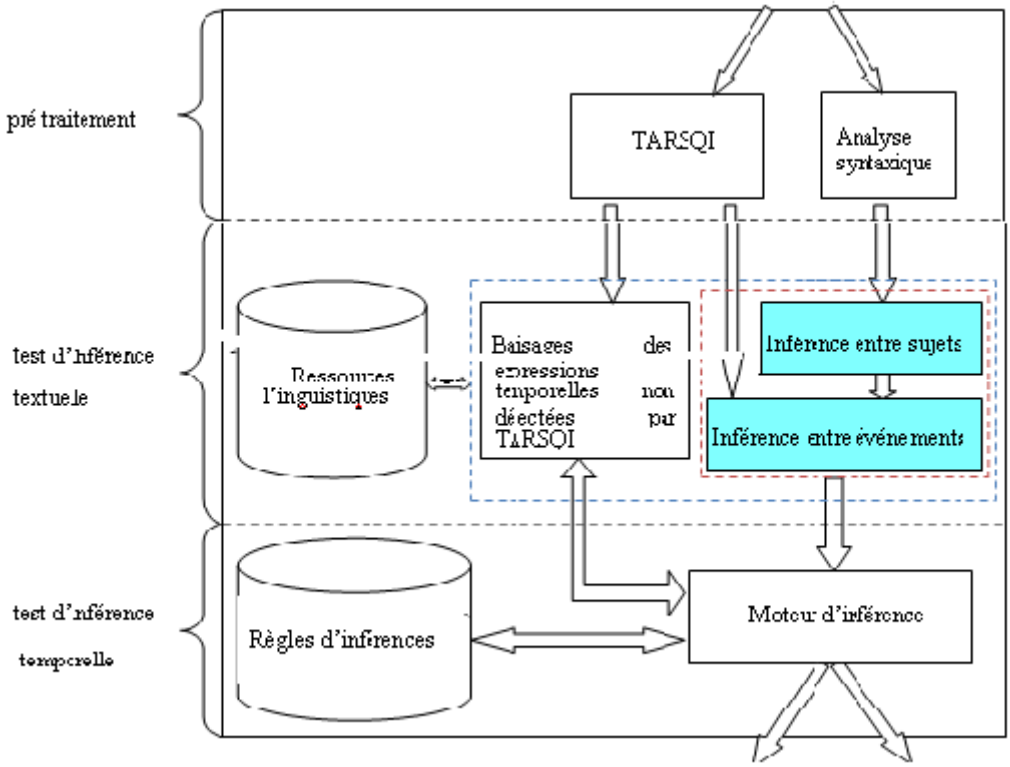

Figure 4.14 : L'inférence entre évènements et sujets

Le but de ce module est de détecter s'il y a une inférence textuelle entre les deux paires de textes (T, H).

Le module est mis en place comme une cascade de sous modules successives. Le premier module détecte les inférences textuelles entre les sujets des deux segments de textes (T, H) et

le deuxième détecte les inférences textuelles entre évènements des deux segments de textes (T, H).

Nous décrivons ci-dessous les deux modules d'inférences :

2.2.1.1) L'inférence entre sujets

Le module d'**inférence entre sujets** détecte s'il y a une inférence textuelle entre les sujets du texte H avec les sujets du texte T. Pour cela, le module utilise les sorties du module *LINK parser* c'est-à-dire que pour chaque sujet détecté, dans le texte H nous recherchons s'il y a une relation de synonymie avec un des sujets du texte T. Pour cela, le module emploie WordNet pour retrouver toutes les relations ontologiques qui lient les deux entités (l'utilisation de wordNet dans notre système est détaillée dans le chapitre cinq). Aussi nous utilisons le comptage de mots pour comparer des groupes de mots (l'algorithme de comptage de mots est expliqué dans l'exemple (1)).

Ce module accepte comme entrée au module le résultat de l'analyse syntaxique des paires de texte T et H et en sortie il existe deux possibilités :

- Si le module trouve une équivalence entre deux sujets, il déclenche le module d'inférence entre événements en lui envoyant les événements correspondants aux deux sujets.

- Si le module ne trouve pas d'équivalence entre sujets, le module envoie le message « pas d'inférence » au module du test d'inférence.

Exemple (1) : paire numéro 3 du corpus de développement.

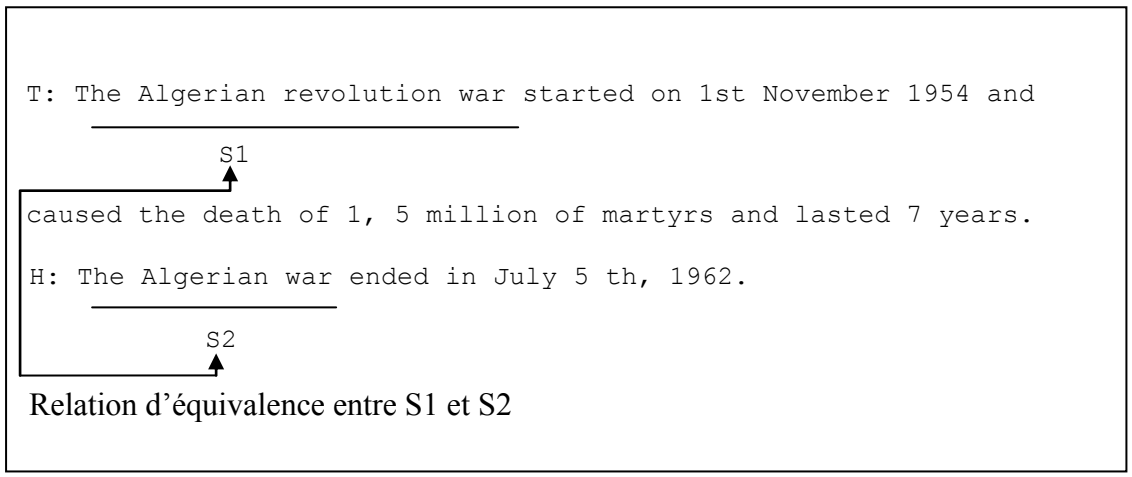

Figure 3.15 : Exemple d'inférence entre sujets

Dans l'exemple suivant le module détecte tous les sujets contenus dans le segment T et le segment H et les met dans deux listes différentes, ensuit il effectue la comparaison entre les évènements des deux listes.

Dans notre exemple la première liste ne contient qu'un seul sujet {S1} et la deuxième liste contient le sujet {S2}. Une relation d'équivalence est détectée entre les évènements S1 et S2.

Pour détecter l'équivalence le module utilise l'algorithme de comptage de mots pour déduire l'inférence entre «`the Algerian war` » et « `the Algerian revolution war`».

Le comptage de mots : L'algorithme récupère les deux groupes deux mots dans deux listes différents et compare chaque mot d'une liste avec les mots contenu dans la deuxième liste et s'il y a un seul mot qui est semblable ou sous mot d'un mot de la deuxième liste, il considère qu'il y a une inférence entre sujets.

2.2.1.2) L'inférence entre évènements

Le module d'inférence entre évènements détecte s'il y a une relation ontologique entre les deux évènements reçus du module d'inférence entre sujets. Pour cela, le module emploie WORDNET pour retrouver toutes les relations qui lient les deux entités.

Le module a comme entrée les évènements reçus du module d'inférence entre sujets et le balisage de TARSQI et comme sortie les résultats suivants :

- S'il trouve une équivalence entre deux évènements, il envoie le message « oui» au module de test d'inférence.
- S'il trouve deux évènements contraires, il envoie le message « non » au module de test d'inférence.
- S'il ne trouve pas de relation entre événements, il envoie le message « pas d'inférence » au module de test d'inférence.

Exemple :

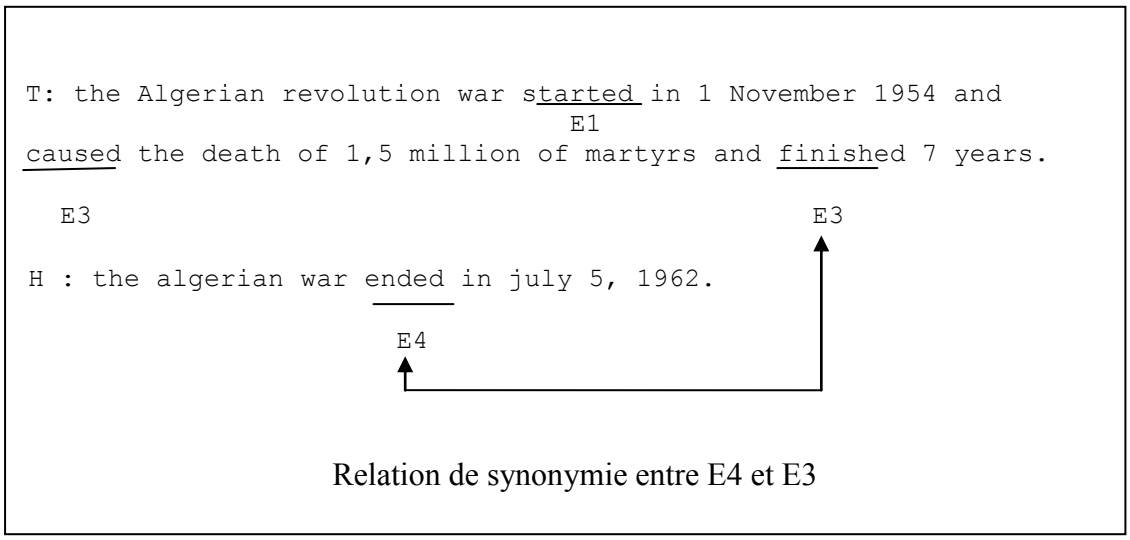

Figure 4.16 : Exemple d'inférence entre évènements

Dans l'exemple ci-dessus le module détecte une liste d'évènements dans le texte T {E1, E2, E3} et une autre liste d'évènements dans le texte H {E1} et effectue la comparaison entre les évènements des deux listes. Une relation ontologique (synonymie) est détectée entre les évènements E4 et E3.

2.2.2) Le balisage des expressions temporelles non détectées par TARSQI

Nous avons remarqué qu'au niveau de la détection des expressions temporelles, les modules de balisages existant ont un manque au niveau de la détection des entités nommées et des adverbiaux temporels.

Dans ce qui suit nous montrons les différentes balises utilisées :

Entités nommées : elles sont balisées par **<NE TYPE=" " Val='' ''>**entité nommée**</NE>.**

- TYPE contient le type d'expression temporelle {date, durée}.
- Val contient la date ou la durée correspondante.

Exemple :

```
T: Germany has become unified since t2: the fall down of the Berlin Wall.

H: Germany unified t1: 19 years ago.

T2 est balisé ainsi:

 <NE TYPE=" date "      Val='' 1989 ''> the fall down of the Berlin Wall</NE>.
```

Figure 4.17 : Exemple de balisages d'expressions temporelles

Notre objectif avec le balisage de « `the fall down of the Berlin Wall`» est de repérer l'évènement dans le temps.

Dans l'exemple précédent le balisage avec le module TARSQI ne détecte que « fall » comme événement et ne le relie pas à une date.

Adverbiaux temporels : ils sont balisés par <TIMEX3 tid="t" TYPE=" " VAL="" >

- **TYPE** contient le type d'expression temporelle {date, durée}.
- **Val** contient les entités que nous avons mises pour représenter les expressions temporelles.

Dans ce qui suit nous relions à chaque expression les symboles correspondants.

- Les jours de la semaine c'est à dire **{Monday, Tuesday, Wednesday, Thursday, Friday Saturday, Sunday}** sont représentés respectivement par des nombres de 1 à 7,

- **{Day before yesterday, two days ago, Yesterday},** sont représentés respectivement par {-2, -2, -1}
- **{everyday often}** sont représentés avec {often}.
- **{Someday, Many days, morning**, **evening, Afternoon}** sont représentés respectivement par PSD, PMD, aMORNING, aNIGHT et AFTERNOON.

2.3) Les Ressources linguistiques

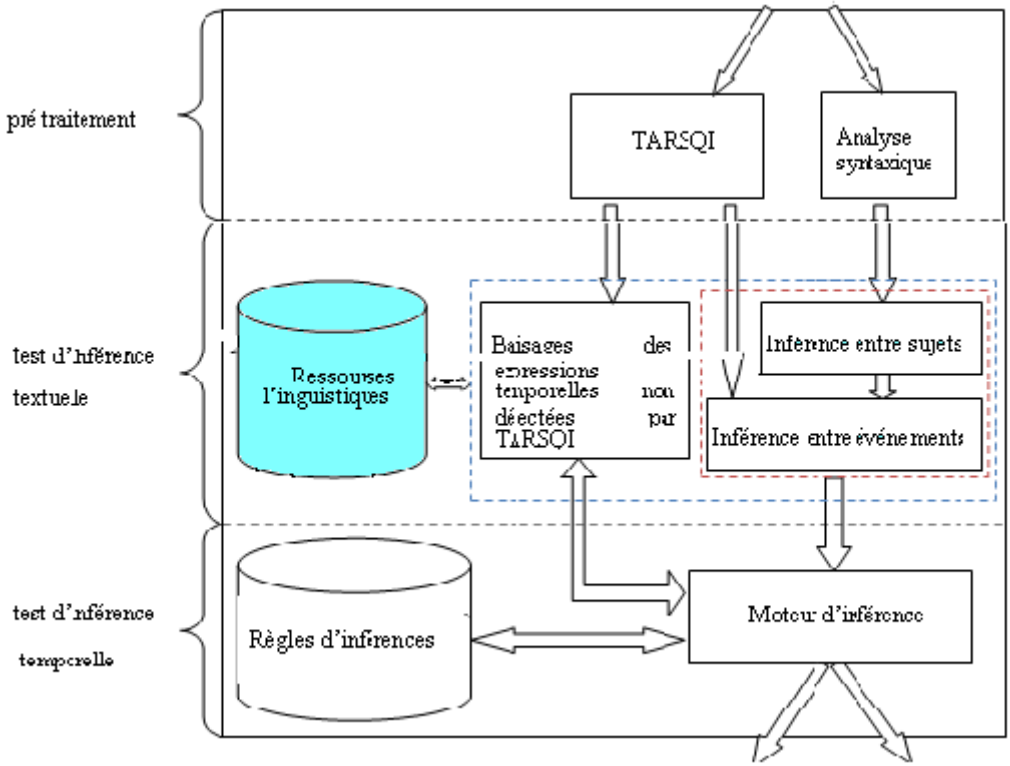

Figure 4.18 : Ressources linguistiques

Deux types de ressources sont utilisés :

2.3.1) Les ressources externes

Dans la conception de notre module d'inférence, l'utilisation d'une ressource lexicale est indispensable au bon fonctionnement des deux modules (inférence entre sujets et inférence entre événements). Pour cela, nous avons choisi d'utiliser **Wordnet** qui est la **base de données lexicale** qui correspond le plus à notre besoin en termes de relations ontologiques entre mots.

2.3.1) Les ressources internes

Ce module est en fait une base de données lexicale contenant les différentes entités nommées qui sont utilisées par le module de balisage pour annoter les expressions temporelles non détectées par le module TARSQI. Puisque notre objectif est de se focaliser sur l'inférence entre expressions temporelles, non pas sur leur détection, nous avons effectué une annotation manuelle de ces expressions temporelles sachant qu'il existe des logiciels payant qui peuvent effectuer la détection.

2.4) Les tests d'inférences temporelles

Cette phase permet de détecter s'il y a une inférence temporelle et aussi textuelle entre les deux segments de textes T et H. Pour cela, nous utilisons un superviseur qui communique avec une base de règles d'inférences et d'après les résultats de la phase précédente (phase de test d'inférence textuelle), il décide de la règle à utiliser.

Dans ce qui suit nous décrivons les modules constituants cette phase.

2.4.1) Les règles d'inférences

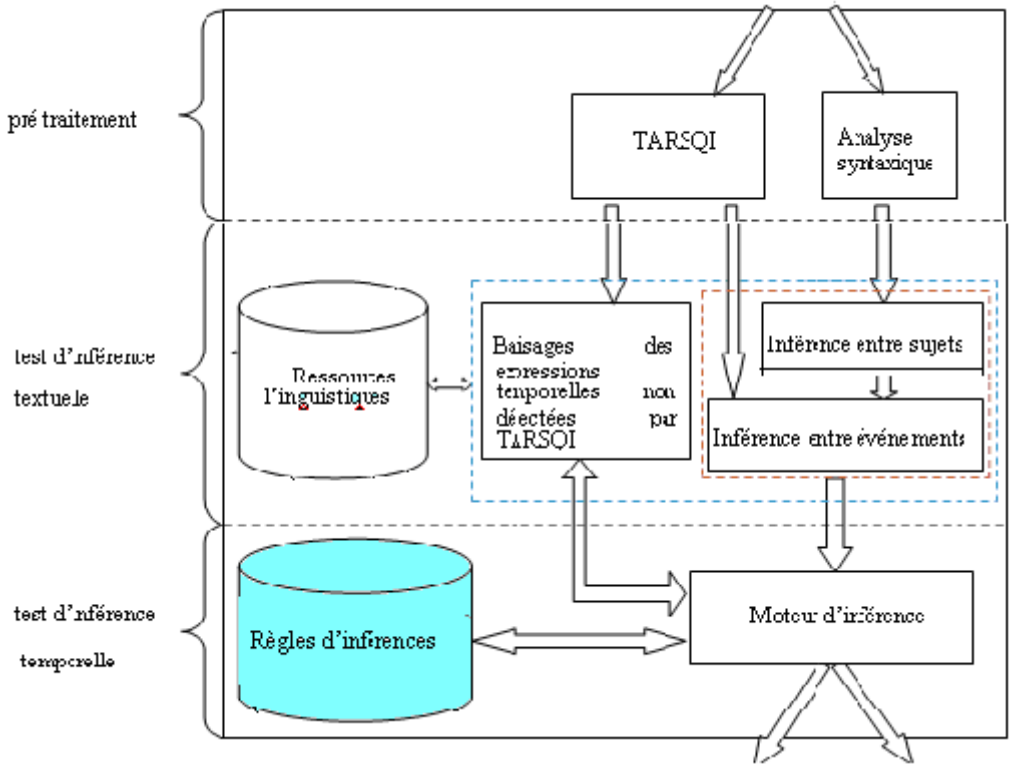

Figure 4.19 : Règles d'inférences

Les règles d'inférences sont divisées en deux groupes.

Groupe 1 : contient les fonctions qui testent si les événements ont un ancrage temporel identique.

Groupe 2 : contient les fonctions qui testent si les événements ont un ancrage temporel différent**.**

2.4.1.1) Définition des fonctions utilisées dans l'abstraction des règles d'inférences

Dans ce qui suit, nous allons définir toutes les fonctions que nous avons utilisées dans l'abstraction de nos règles d'inférences.

Sachant que **S** représente un des textes **T** ou **H** de la paire, **E** représente un événement dans le texte et **t** représente une expression temporelle**.**

- **<S, E>:** indique que l'événement **E** est dans le segment de texte **S**.
- **Subj** (**<S, E>**): la fonction retourne le **sujet** de l'évènement **E** dans le segment **S**.
- **Equivalent** (**<S,E1>**,**<S,E2>**): la fonction retourne s'il y a une équivalence entre **E1** et **E2** ou pas.
- **Contraire** (**<S,E1>**,**<S,E2>**): la fonction retourne s'il y a une antonymie entre **E1** et **E2** ou pas.
- **Inclut** (**<S,t>**,**<S,t>**): la fonction retourne si **t** est inclut dans l'intervalle de **t'** ou pas.
- **Egale** (**<S, t1>**,**<S, t2>**): la fonction retourne s'il y a une équivalence entre **t1** et **t2** ou pas.
- **Début** (**<S,t>**,**<S,E>**,type): la fonction et booléen et renvoi vrai ou faux, si t est la date de debut de l'événement **E** ou pas.
- **After**(**<S,E1>**,**<S,E2>**,type): la fonction retourne s'il y a une équivalence entre **t1** et **t2** ou pas.
- **before**(**<S,E1>**,**<S,E2>**,type): la fonction retourne si **E1** est avant **E2** ou pas .
- **Fin** (**<S,E>**,**<S, t>**,type): la fonction retourne si **t** est la date de fin de l'événement **E** ou pas.
- **Relation** (**<S, E>**,**<S,t>**,type): indique s'il y a une relation TLINK entre l'événement **E** et la date **t**. Tel que l'argument **type** indique le type d'expression temporelle {date, durée}.
- **Inf** (**T**, **H**): indique, en sortie s'il y à une inférence entre les segments de textes T et H ou pas.
- **Somme**(**<S, t1>**,**<S,t2>**): renvoi en sortie la somme des deux dates.
- **Différence**(**<S, t1>**,**<S,t2>**): renvoi en sortie la différence entre les deux dates.

Les symboles utilisés dans les schémas de représentation de nos règles d'inférences sont présentés ci-dessous :

: représente le lien entre les deux évènements.

: représente le lien entre les deux expressions temporelles.

: représente les éléments du texte T.

: représente les éléments du texte H.

: représente le lien entre les événements et expressions temporelles.

: représente l'événement.

: représente l'expression temporelle.

.t : représente une date.

.e : représente un évènement.

.d : représente une durée.

Ainsi, les différentes règles d'inférences conçues sont reparties comme suit :

4.1.1.2) Les règles du groupe 1

Ces règles permettent de savoir s'il y a un ancrage temporel entre évènements.

Si **équivalent** (<T, e1>, <H, e2>) ^ **équivalent** (<T, **Subj** (<T, e1>)>, <H, **Subj** (<H,e2>)>) alors :

A) Règle R1

Si les différentes conditions se réunissent c'est-à-dire :

- détecter une équivalence entre les deux évènements (**e1**, **e2**)
- chaque événement est relie avec la même relation TLINK avec une date **e1→t1** et **e2→t2**
- les dates sont égales.

Nous aurons une inférence temporelle et textuelle entre les segments **T** et **H.**

L'abstraction de la règle R1 est représentée dans ce qui suit :

- Si **relation** (<T,e1>,< T,t> ,date) ^ **relation** (<H,e2>, <H,t'>,date)

  alors **Inf**(T,H)= Vraie SSi **inclus**(<T,t> ,<H, t'>) v **égale**(<T,t> ,<H, t'>)

  sinon **inf**(T,H)=Faux

Les numéros des exemples dans le corpus de développement où les règles peuvent s'appliquer : 8, 7, 15, 14, 18, 28, 29, 30.

Cette figure représente la règle d'inférence R1:

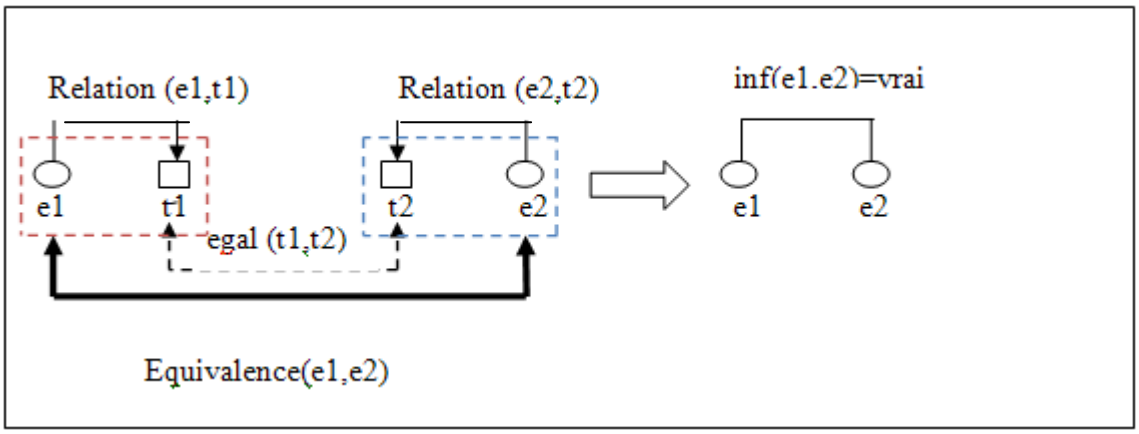

Figure 4.20 : Règle R1 de l'inférence temporelle

Dans l'exemple qui suit, nous allons appliquer la règle R1 sur la paire numéro 8 du corpus.

8) <pair id="9" value="TRUE" >
T: since its e1: **creation** in 1948, Israel had faced a lot of conflict with the Arabic countries.
H: Israel e2: **was conceived** in 1948.

Puisque l'événement e1 se déroule en 1948 et l'événement e2 se déroule entre 1948 et puisque e2 est le synonyme de e1 alors il y a une inférence temporelle entre e1 et e2.

B) Règle R2

Si les différentes conditions se réunissent c'est-à-dire :

- détecter une équivalence entre les deux évènements (**e1**, **e2**)
- **t1** est la date de début de l'évènement **e1**, **t2** est la date de fin de l'évènement **e1** et l'événement **e2** est relié a une durée **e2→d**.
- la différence entre les dates **t1** et **t2** est égale à la durée **d.**

Nous aurons une inférence temporelle et textuelle entre les segments **T** et **H.**

L'abstraction de la règle R2 est représentée dans ce qui suit :

- Si **debut**(<T,e1>,<T,t>,date) ^ **fin**(<T,e1>,<T,t'>,date) ^ **relation**(<H,e2>, <H,t''>,durée)

  Alors **Inf**(T,H)= égale(**Différence** (<T,t>,<T,t'>) , <H,t''>)

  sinon **Inf**(T,H)=Faux

Les numéros des exemples dans le corpus de développement où les règles peuvent s'appliquer: 4, 12.

Cette figure représente la règle d'inférence R2 :

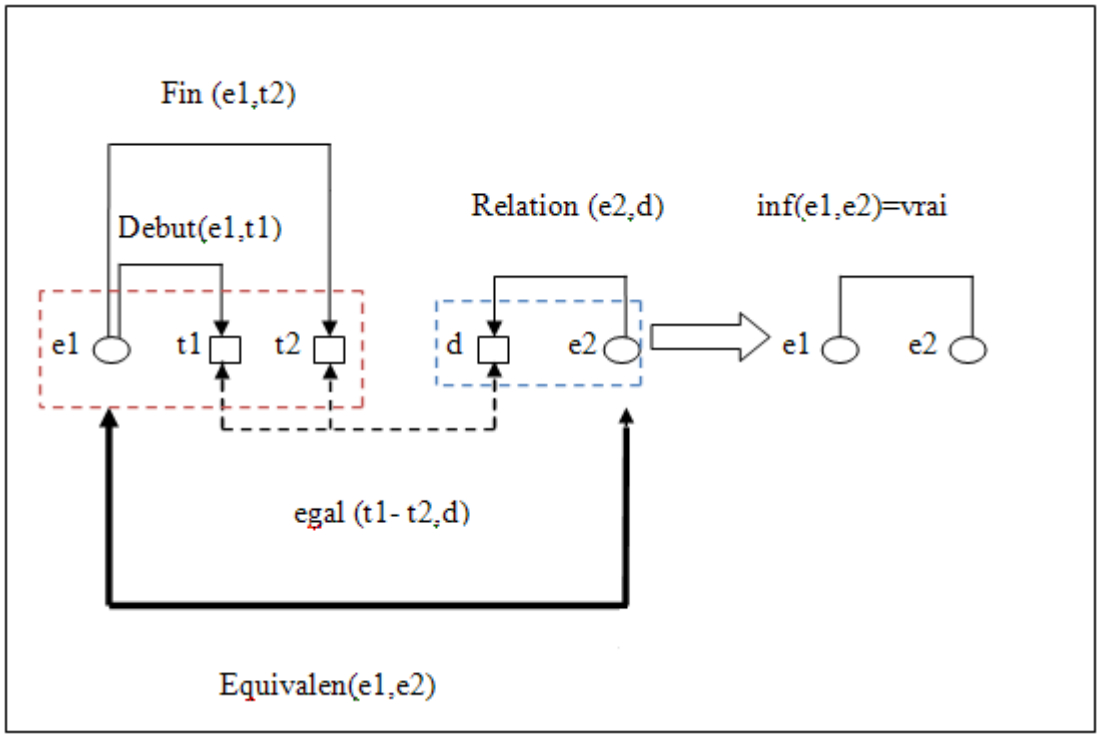

Figure 4.21 : Règle R2 de l'inférence temporelle

Dans l'exemple qui suit, nous allons appliquer la règle R2 sur la paire numéro 4 du corpus.

4) <pair id="4" value="TRUE" >
T: Pasteur began looking for the germ that causes rabies in 1880, and in july 1885 he **found** the efficient vaccine against the illness.
H: **to find** the vaccine, Pasteur's researches took five years.

Puisque l'événement e1 se déroule au même moment que l'évènement e2 car si nous ajoutons « **7 years»** à « **november the first 1954»** nous serions en 1962 qui est la date où s'est déroulé l'évènement e2 et puisque e2 est le synonyme de e1 alors il y a une inférence temporelle entre e1 et e2.

C) Règle R3

Si les différentes conditions se réunissent c'est-à-dire :

- détecter une équivalence entre les deux évènements (**e1**, **e2**)
- **t1** est la date de début de l'évènement **e1**, événement **e1** est relié a une durée **e1→d** et **t2** est la date de fin de l'évènement **e1**.
- la somme entre la date **t1**et la durée **d** est égale à **t2.**

Nous aurons une inférence temporelle et textuelle entre les segments **T** et **H.**

L'abstraction de la règle R3 est représentée dans ce qui suit :

- Si **debut**(<T,e1>,<T,t>,date) ^ **relation**(<T,e1>,<T,t'>,durée) ^ **fin**(<H,e2>, <H,t''>,date)

  Alors **Inf**(T,H)= **egal**(**Somme**(<T,t>, <T,t'>) , <H,t''>)

  sinon **Inf**(T,H)= Faux

Les numéros des exemples dans le corpus de développement où les règles peuvent s'appliquer : 3, 11.

Cette figure représente la règle d'inférence R3:

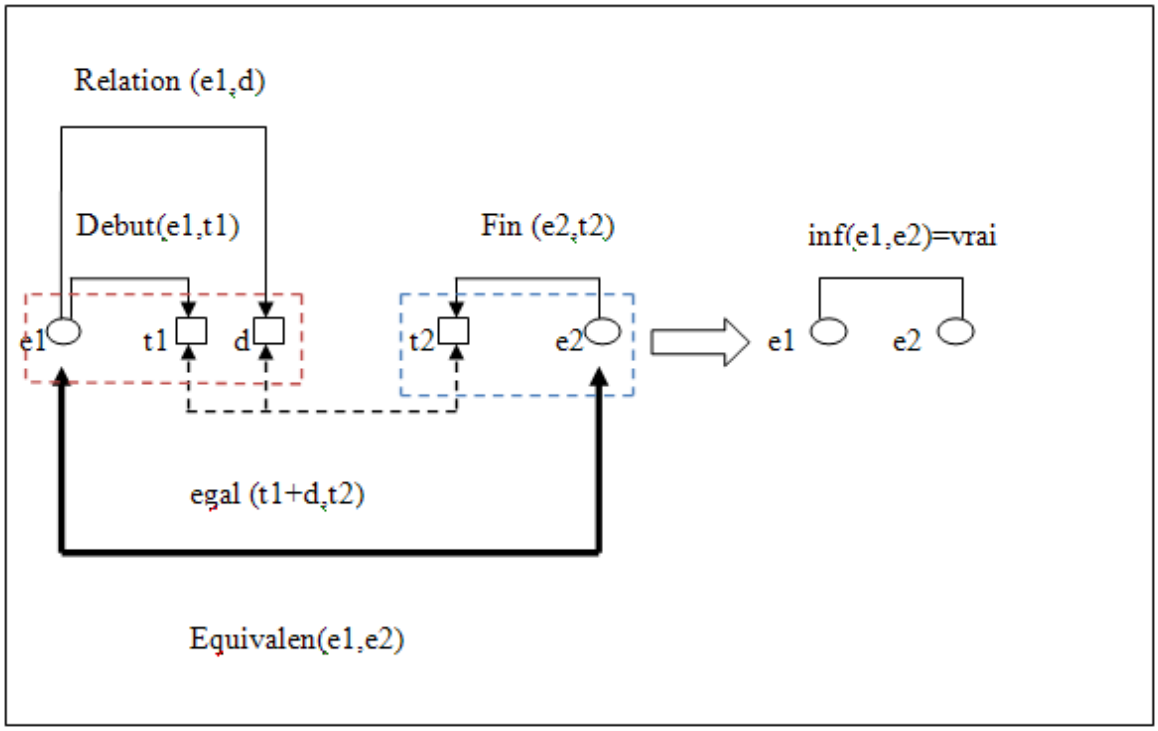

Figure 4.22 : Règle R3 de l'inférence temporelle

Dans l'exemple qui suit, nous allons appliquer la règle R3 sur la paire numéro 3 du corpus de développement.

3) <pair id="3" value="TRUE" >

T: the Algerian revolution war started on **november the first 1954**, it caused the death of 1,5 million of martyrs and it e1: **lasted 7 years**.
H: the Algerian revolution war e2: **ended** on **july the fifth 1962**.

L'événement e1 se déroule au même moment que l'événement e2 car si on ajoute « **7 years»** à « **november the first 1954»** nous serions en 1962 qui est la date où se déroule l'évènement e2 et puisque e2 est le synonyme de e1 alors il y a une inférence temporelle entre e1 et e2.

D) Règle R4

Si les différentes conditions se réunissent c'est-à-dire :

- détecter une équivalence entre les deux évènements (**e1**, **e2**)
- **t1** est la date de fin de l'évènement **e1**, événement **e1** est relié à une durée **e1→d** et **t2** est la date de début de l'évènement **e1**.
- la somme entre la date **t1** et la durée **d** est égale a **t2.**

Nous aurons une inférence temporelle et textuelle entre les segments **T** et **H.**

L'abstraction de la règle R4 est représentée dans ce qui suit :

- Si **relation**(<T,e1>,<T,t>,durée) ^ **fin**(<T,e1>,<T, t'>,date) ^ **debut**(<H,e2>, <H, t''>,date)

  Alors **Inf**(T, H)= **égale**(**Différence** (<T, t'>,<T, t>),<H, t''>)

  sinon **Inf**(T, H)=Faux

Les numéros des exemples dans le corpus de développement où les règles peuvent s'appliquer : 10, 20.

Cette figure représente la règle d'inférence R4:

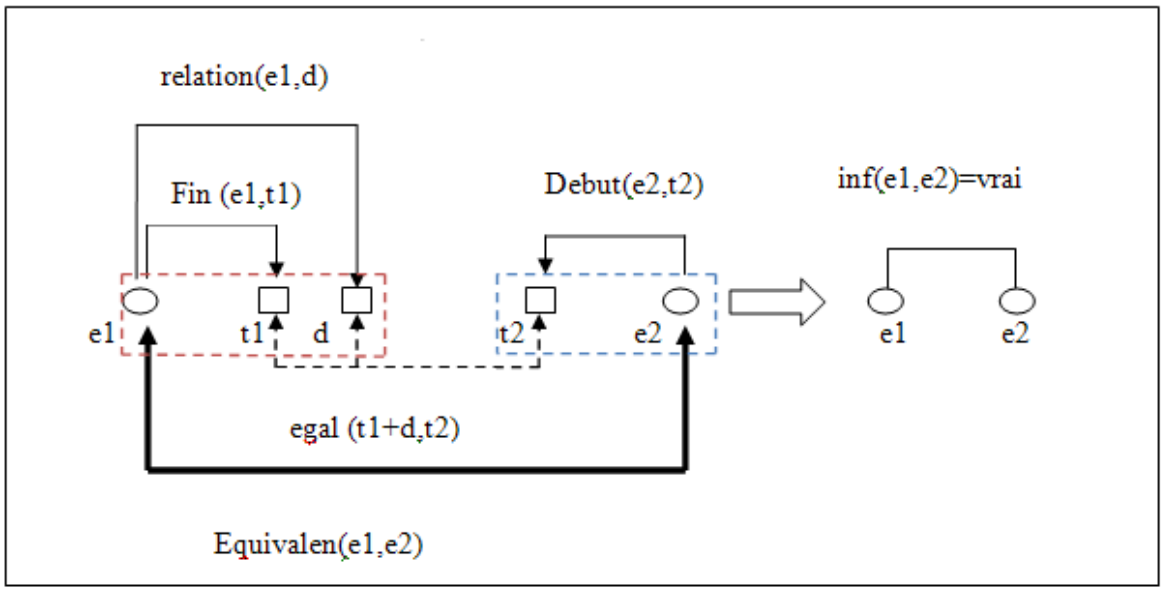

Figure 4.23: Règle R4 de l'inférence temporelle

Dans l'exemple qui suit, nous allons appliquer la règle R4 sur la paire numéro 10 du corpus.

10) <pair id="11" value="TRUE" >
T: on **t1:** december 2nd 1804, Napoleon Bonaparte became the emperor of the French, before **d1:** one year exactly, he **e1: won** the battle of Austerlitz.
H: in t2: 1803, Napoleon **e2: won** the battle of Austerlitz.

L'événement e1 se déroule au même moment que l'événement e2 car si on réduit « **one year exactly** » à « **december 2nd 1804** » nous serions en 1803 qui est la date où se déroule l'évènement e2 et puisque e2 est le synonyme de e1 alors il y a une inférence temporelle entre e1 et e2.

E) Règle R5

Si les différentes conditions se réunissent c'est-à-dire :

- détecter une équivalence entre les deux évènements (**e1**, **e2**)
- l'événement **e1** est relié avec une relation TLINK **e1→t2** et l'événement **e1** est relié avec la même relation TLINK à une durée **e2→d.**
- inclusion entre la date **t1** et la durée **d.**

Nous aurons une inférence temporelle et textuelle entre les segments **T** et **H.**

L'abstraction de la règle R5 est représentée dans ce qui suit :

- **Relation** (<T, e1>, < T,t> ,date) ^ **relation**(<H,e2>, <H,t'>,durée)

  alors **Inf**(T, H)= Vraie SSi **inclus**(<T,t> ,<H, t'>)

  sinon **inf**(T, H)=Faux

Le numéro de l'exemple dans le corpus de développement où cette règle peut être appliquée: 1.
Cette figure représente la règle d'inférence R5:

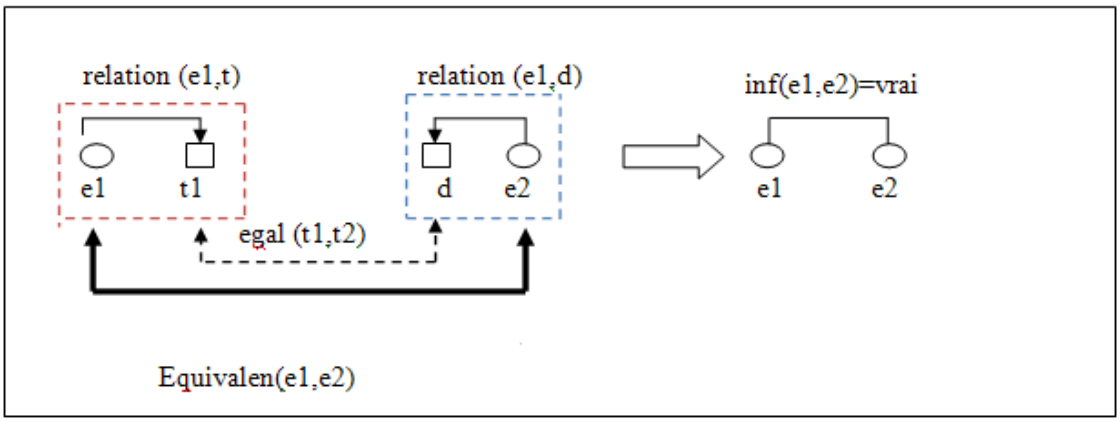

Figure 4.24 : Règle R5 de l'inférence temporelle

Application de la règle 5 sur la paire numéro 1 du corpus.

1) <pair id="1" value="TRUE" >
T: the second world war e1: **finished** in 1945.
H: the end of the second world war e2: **took part** between 1940 and 1950.

Puisque l'événement e1 ce déroule en 1945 et l'événement e2 se déroule entre 1940 et 1950 et puisque e2 est le synonyme de e1 alors il y a une inférence temporelle entre e1 et e2.

4.1.1.3) Groupe 2

Cette règle permet de savoir s'il n'y a pas d'ancrage temporel entre évènements.

Si **contraire**(<T,e1> ,<H,e2>) ^ **équivalent**(<T ,**Subj**(e1)> **,** <T,**subj**(e2)>) =<H,**Subj**(e2)>)

A) Règle R6

Si les différentes conditions se réunissent c'est-à-dire :

- détecter que l'évènement **e1** est le contraire de l'évènement **e2**.
- l'événement **e1** se produit soit avant ou après **e2**.

Nous aurons une inférence temporelle et textuelle entre les segments **T** et **H.**

L'abstraction de la règle R6 est représentée dans ce qui suit :

Si **relation** (<T, e1>, <T, t>, date) ^ **relation**(<H, e2>, <H, t'>, date)

Alors

Inf(T, H) = Vraie SSi **before**(**relation** (<T,e1>,<T,t>, date), **relation**(<H, e2>, <H, t'>, date)

v

after(**relation** (<T, e1>,<T, t>, date) **, relation**(<H, e2>, <H, t'>, date)

sinon **inf**(T, H)=Faux

Les numéros des exemples dans le corpus de développement où les règles peuvent s'appliquer : 2, 5.

Cette figure représente la règle d'inférence R6:

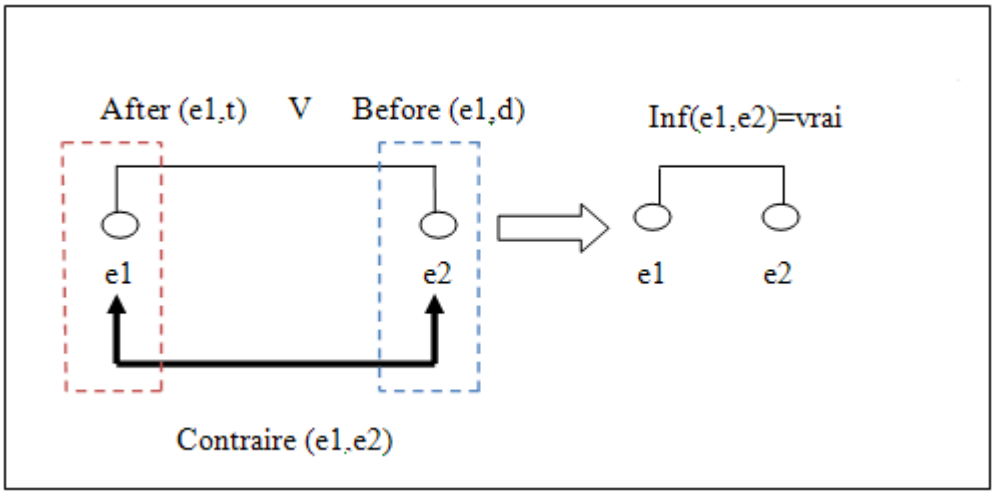

Figure 4.25 : Règle R6 de l'inférence temporelle

Application de la règle 6 sur la paire 2 du corpus.

2) <pair id="2" value="TRUE" >

T: Algeria got its e1: **independence** in **1962**.

H: **Before 1962** Algeria was e2: **colonized**.

Dans cet exemple ci-dessus, puisque l'événement e1 se déroule en 1962 et l'événement e2 se déroule avant l'évènement e1 et puisque e2 est l'antonyme de e1 alors il y a une inférence temporelle entre e1 et e2.

4.2.2) Le superviseur

Ce module, accepte en entrée, les résultats du module « inférence entre événements », le résultat de « TARSQI », « les ressources » à ajouter et « les règles d'inférences » et en sortie, il indique s'il y une inférence textuelle ou pas.

Le superviseur permet de choisir les règles d'inférences temporelles à appliquer et de décider de l'existence ou pas de l'inférence textuelle.

Ainsi, le superviseur applique la procédure suivante:

- Si le module a comme message « pas d'inférence » de la phase précédente c'est-à-dire du module de test d'inférence le superviseur va afficher, « pas d'inférence textuelle ».

- Si le module a comme message « non » qui veut dire qu'il y a une relation d'antonymie entre les évènements le module va exécuter les règles d'inférences temporelles qui détectent si les deux évènements ne sont pas ancrés temporellement.

- Si le module a comme message « oui » qui veut dire qu'il y a une relation de synonymie entre les évènements, le module va exécuter les règles d'inférences temporelles qui détectent si les événements sont ancrés temporellement.

Nous représentons dans la figure suivante l'architecture du superviseur :

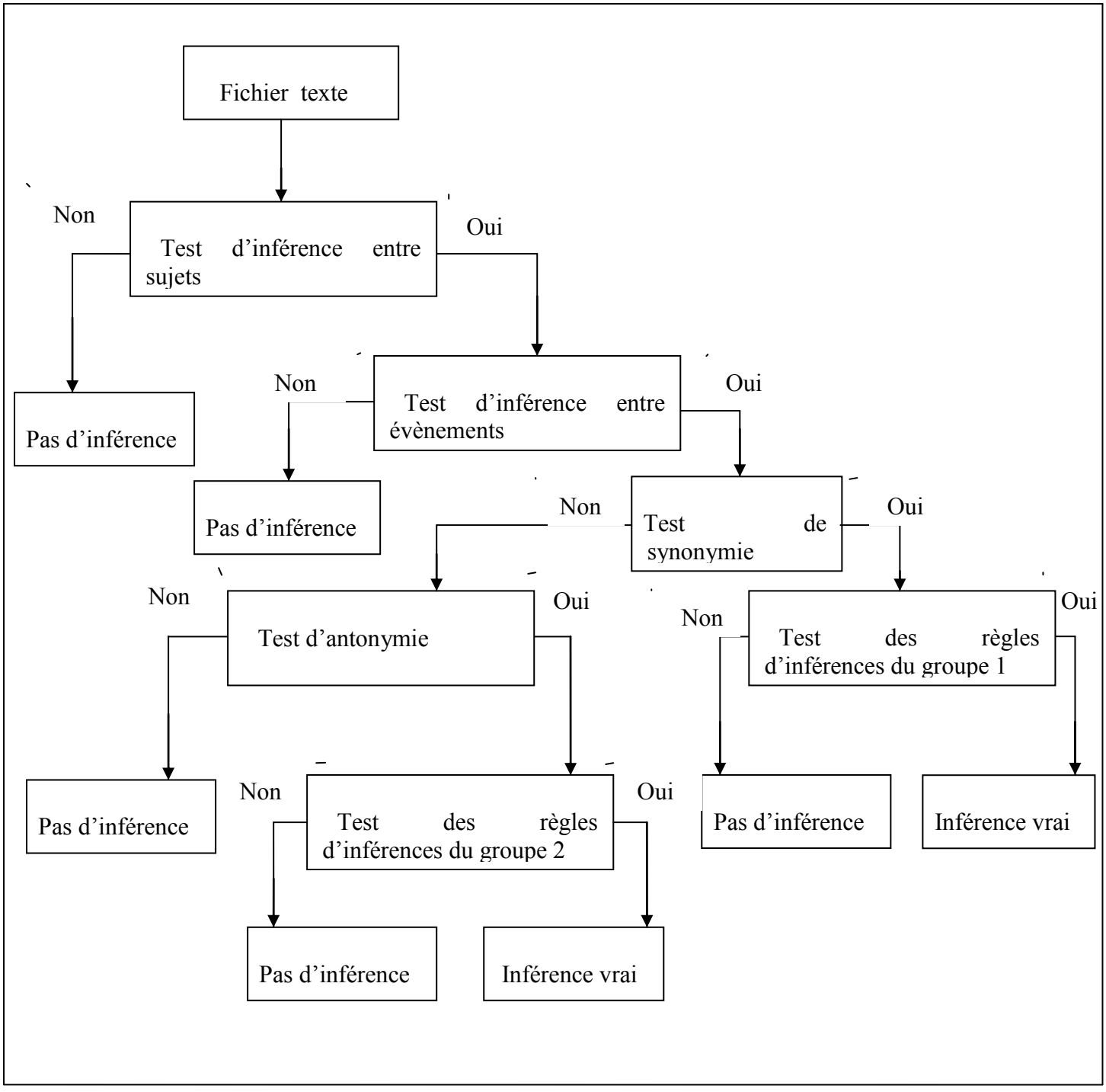

Figure 4.26 : Architecture du superviseur

Il existe des cas où plusieurs règles peuvent s'appliquer. Pour cela, le superviseur prend les mesures suivantes :

- S'il existe une fonction qui retourne une fausse inférence temporelle, cela implique qu'il n'y a pas d'inférence textuelle entre les segments T et H.
- Si toutes les fonctions retournent une inférence temporelle, cela implique qu'il y a une inférence textuelle entre les segments T et H.

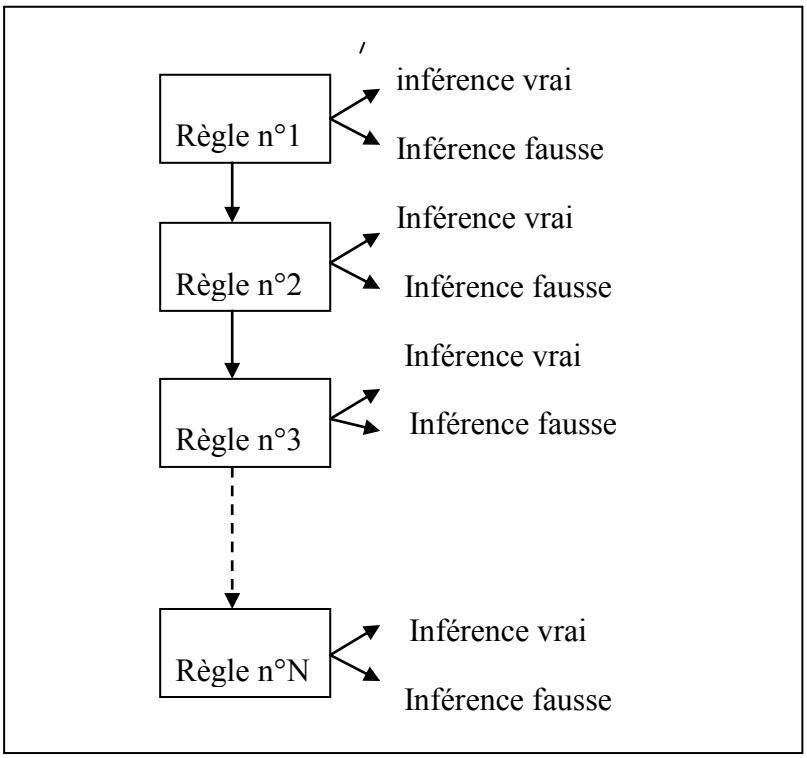

Figure 4.27: Test des règles d'inférences

Comme il est montré dans la Figure **4.27** le superviseur exécute les règles du même groupe une par une.

4.4) Conclusion

Nous avons présenté dans ce chapitre, notre projet TIMINF. Son architecture informatique se base sur cinq modules principaux. Les deux modules TARSQI et *Link Grammar Parser* constituent la phase de prétraitement indispensable à la phase de test d'inférence, qui nous permet de détecter l'inférence entre évènements et sujets. Le module de balisage qui est inclue dans la deuxième phase est utilisé pour baliser les expressions temporelles non détectées par TARSQI.

Le superviseur est le dernier module de notre système. Celui ci communique avec une base de règle et décide des choix des règles d'inférences temporelles a appliqué. Il a aussi le rôle de tester l'inférence textuelle entre les phrases T et H d'après les données reçues de tous les composants du système. Nous allons présenter dans le chapitre qui suit les différentes étapes de la mise en œuvre du système TIMINF ainsi qu'une étude expérimentale.