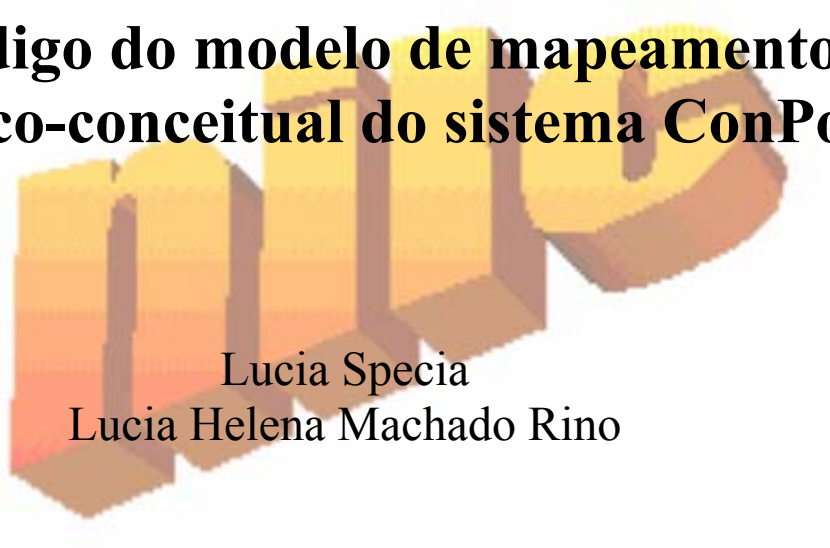
Universidade de São Paulo - USP
Universidade Federal de São Carlos - UFSCar
Universidade Estadual Paulista - UNESP

O código do modelo de mapeamento sintático-conceitual do sistema ConPor



Lucia Specia
Lucia Helena Machado Rino

NILC-TR-03-09

Maio, 2003

Série de Relatórios do Núcleo Interinstitucional de Lingüística Computacional
NILC - ICMC-USP, Caixa Postal 668, 13560-970 São Carlos, SP, Brasil

Resumo

Este relatório apresenta o código do sistema ConPor, cuja função é mapear estruturas sintáticas de sentenças do português em estruturas conceituais UNL. Esse código, em Prolog, diz respeito ao núcleo do processamento do sistema, incluindo a forma de acesso a ele, a estrutura dos dados de entrada, as regras da gramática de projeção e os *templates* de relacionamento (armazenadas no repositório conceitual, um dos resursos lingüísticos específicos), as entradas do léxico enriquecido (o outro recurso lingüístico específico) e os procedimentos auxiliares utilizados por esses recursos.

Este trabalho conta com o apoio
financeiro da CAPES

Índice

Figuras

1 Introdução

O **ConPor** (**Con**ceitualização do **Por**tuguês) (Specia & Rino, 2002a; 2003) é um sistema de interpretação sentencial do português cuja tarefa consiste em gerar estruturas conceituais UNL (*Universal Networking Language*) (UNL, 2001) a partir de estruturas sintáticas produzidas pelo *parser* Curupira (Martins et al., 2002). Essa tarefa, isto é, o mapeamento de estruturas sintáticas em estruturas conceituais (**mapeamento sintático-conceitual**), é realizada pelo módulo **Gerador Conceitual**, a partir do conhecimento lingüístico armazenado no **Repositório Conceitual** e no **Léxico Enriquecido**, conforme mostra a Figura 1. Essa figura representa o ambiente global do ConPor: os módulos pontilhados indicam recursos e processos já existentes e reutilizados nesse sistema e os módulos sombreados indicam seus componentes específicos.

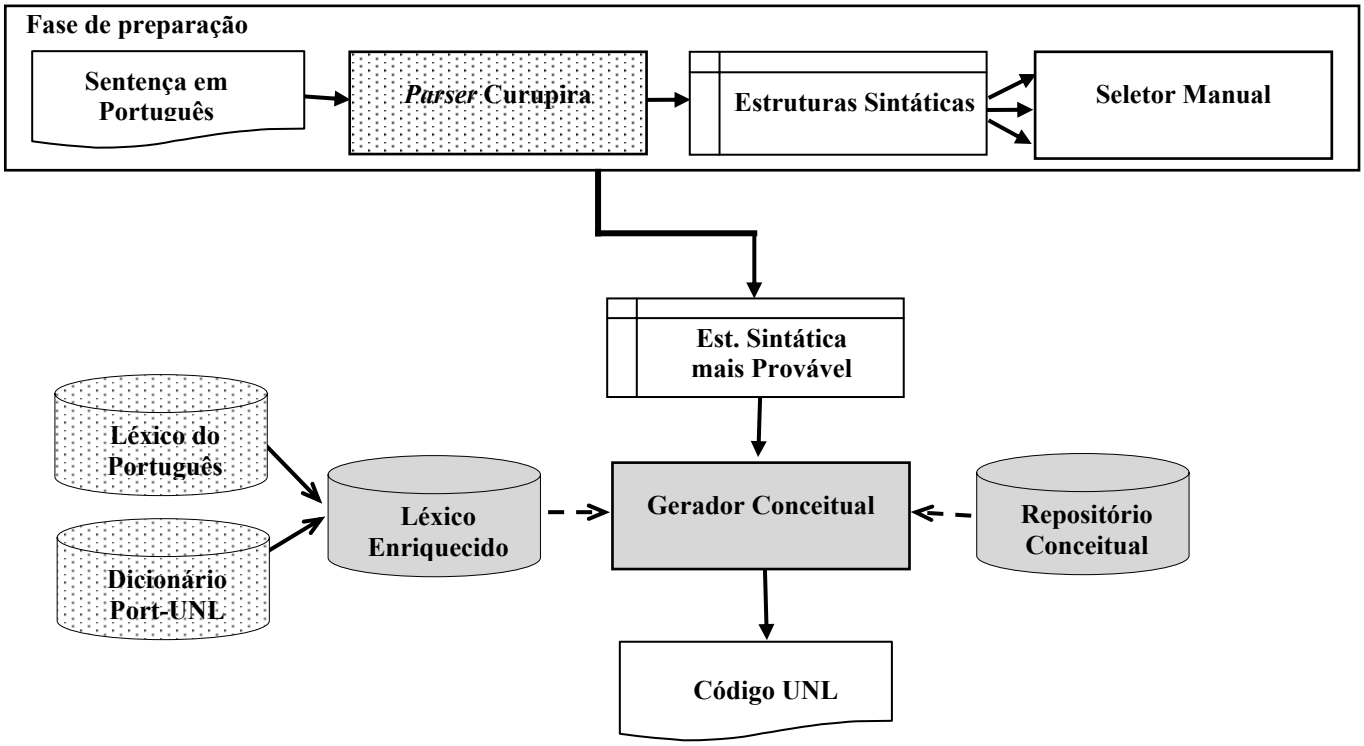


Figura 1 – Ambiente do ConPor

O núcleo do modelo de mapeamento, isto é, os módulos específicos do ConPor, foram implementados na linguagem Prolog (utilizando a ferramenta Amzi Prolog)[1]. Este relatório apresenta o código dos dois recursos lingüísticos do sistema, isto é, do Repositório Conceitual e do Léxico Enriquecido. Apresenta, também, o código para o acesso às regras desse repositório, que corresponde ao módulo Gerador Conceitual, o código das estruturas de entrada para essas regras e o código dos procedimentos auxiliares utilizados pelos módulos principais do ConPor.

O código em Delphi referente ao módulo da interface gráfica para o acesso ao modelo de mapeamento do ConPor não é descrito, uma vez que esse módulo não representa um componente necessário para a execução do sistema, mas somente uma facilidade para a sua utilização, isto é, a tarefa principal do ConPor, que é o maepamento sintático-conceitual, é realizada independentemente dessa interface, por meio do código implementado em Prolog. De modo geral, nesses códigos, os comentários da implementação são mantidos e indicados pelo símbolo "%" no início da linha. Esses comentários indicam, entre outras coisas, a sintaxe de cada predicado. Tanto nos comentários quanto nos predicados,

[1] Disponível em www.amzi.com.

são utilizados, em alguns casos, os termos "ES" para "estrutura sintática" e "EC" para "estrutura conceitual".

Para utilizar esse código no ambiente do Amzi Prolog, pode-se criar um arquivo de projeto (.ppj), separar os diferentes módulos em arquivos prolog (.pro), construir, compilar e executar o projeto. Opcionalmente, pode-se armazenar todos os módulos em um único arquivo prolog (.pro) e interpretar esse arquivo.

2 Programa principal

O Gerador Conceitual é responsável, basicamente, pela aplicação do conhecimento dos dois recursos específicos do ConPor, utilizando-se do próprio mecanismo de inferência da linguagem Prolog. Para mapear uma estrutura sintática de uma sentença em sua correspondente conceitual, o Gerador Conceitual pode ser disparado por meio de um dos seguintes predicados:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                    Formas de chamada do programa principal                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Chamada Prolog para retornar uma única EC de uma ES (a primeira)
mapear(S) :-     estrutura(S,ES),
                 (r_Acao(ES,UNL); r_Estado(ES,UNL); r_Processo(ES,UNL)),
                 recupera_sentenca(ES,Sent),
                 write(UNL),
                 nl,
                 write(Sent).

% Chamada Prolog para retornar todas as ECs de uma ES
mapear_todasECs(S,Sent,Res) :-   estrutura(S,ES),
                                 recupera_sentenca(ES,Sent),
                                 findall(UNL,mapear(ES,UNL),Res).

mapear(ES,UNL) :- (r_Acao(ES,UNL); r_Estado(ES,UNL); r_Processo(ES,UNL)).
```

O primeiro dos predicados, isto é, "mapear/1", utiliza como argumento somente um código que indica a estrutura sintática de entrada (S), dispara as regras principais para os três tipos de sentenças possíveis (ação, processo ou estado) com a estrutura sintática correspondente a esse código (ES), recupera a sentença dessa estrutura (Sent), mostra o código UNL resultante da aplicação das regras (somente o primeiro encontrado) e a sentença recuperada.

O segundo predicado, isto é, "mapear_todas/3", também utiliza como argumento um índice que indica a estrutura sintática de entrada (S), recupera a sentença da estrutura de entrada (Sent), encontra e retorna todos os códigos UNL possíveis (Res) a partir da aplicação das regras principais para os três tipos de sentenças possíveis com a estrutura sintática da sentença de entrada (ES).

3 Dados de entrada

A entrada efetiva para o processo de mapeamento consiste da estrutura sintática de uma sentença do português, geradas pelo *parser* Curupira, conforme mencionado. No entanto, para ser manipulada pelo Prolog, essa estrutura é convertida em um formato adequado, por um módulo auxiliar de pré-edição, implementado e disponibilizado juntamente à interface de acesso ao sistema, em Delphi. No código em Prolog, as estruturas sintáticas dos córpus de exemplos (denominado córpus base) e de teste do ConPor devidamente convertidas por esse módulo são, ainda, indexadas para facilitar a realização de consultas, de modo que cada estrutura possa ser acessada por meio de um código (ou índice). Por exemplo, a estrutura da primeira sentença do córpus base é acessada pelo código "s1". A seguir, é ilustrado o código para

indexar as estruturas das 122 sentenças do córpus base e, na seqüência, o código para indexar as estruturas das 80 sentenças do córpus de teste.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                 estrutura sintática das sentenças do corpus base indexadas                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% estrutura(indice,estrutura_sintatica).

estrutura(s1,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(considere)),od(od_simples(sn([aadne(adj(seu)),substs(desgaste),aadnd(aadnd_simples(sadj(adj(físico))))])))]))))).

estrutura(s2,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(considere)),od(od_simples(sn([aadne(adj(seu)),subst(desconforto),aadnd(aadnd_simples(sadj(adj(espiritual))))])))]))))).

estrutura(s3,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(considere)),od(od_simples(sn([aadne(adj(sua)),subst(necessidade),cn(sp([preposicao(de),sn([subst(calor),aadnd(aadnd_simples(sadj(adj(humano))))])]))])))]))))).

estrutura(s4,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(considere)),od(od_simples(sn([aadne(adj(sua)),subst(necessidade),cn(sp([preposicao(de),sn(subst(afetividade))]))])))]))))).

estrutura(s5,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(sabe)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(tamanho),cn(sp([preposicao(de),sn([aadne(adj(sua)),subst(perna)])]))])))]))])))).

estrutura(s6,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(horas),aadnd(aadnd_simples(sadj(adj(delicadas))))]))),predicado(predv(svi(verbo(passarão)))),aadvo(aadvo_composto([aadvo_simples(sp([preposicao(em),sn([aadne(adj(relativo)),subst(controle)])])),coordenador(e),aadvo_simples(sp([preposicao(em),sn([aadne(adj(relativa)),subst(paz)])]))]))])))).

estrutura(s7,frase(periodo(periodo_independente(predicado(predv([svtdi(verbo(tire)),oi(oi_simples([poi(de),sn([aadne(sdet(artigo(o))),subst(armário)])])),od(od_simples(sn([aadne(adj(aqueles)),subst(planos),aadnd(aadnd_simples(sadj(adj(malucos))))])))])))))).

estrutura(s8,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(perspectivas),cn(sp([preposicao(de),sn([subst(retorno),cn(sp([preposicao(a),sn([aadne(sdet(artigo(um))),subst(estado),aadnd(aadnd_simples(sadj(adj(anterior))))])]))])]))]))),predicado(predn([svl(verbo(estão)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(longínquas)])))]))])))).

estrutura(s9,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(lua),aadnd(aadnd_simples(sadj(adj(crescente))))]))),predicado(predv(svi(verbo(ocorre)))),aadvo(aadvo_simples(sadv(adv(amanhã))))])))).

estrutura(s10,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(este)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sn([aadne(sdet(artigo(um))),subst(momento),aadnd(aadnd_simples(sadj(adj(delicado))))])))]))])))).

estrutura(s11,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(lua)]))),predicado(predv([svtd(verbo(transita)),od(od_simples(sn([aadne(adj(este)),subst(signo)])))]))])))).

estrutura(s12,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(estará)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(afinado),cn(sp([preposicao(com),sn([aadne(adj(suas)),subst(cores),aadnd(aadnd_simples(sadj(adj(pessoais))))])]))])))]))])))).

estrutura(s13,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predvn([oda(lexico(se)),svtd(verbo(tornará)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(ousado)])))]))])))).

estrutura(s14,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(sol)]))),predicado(predv([svti(verbo(ingressou)),oi(oi_simples([poi(em),sn(subst(sagitário))]))]))])))).

estrutura(s15,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd(verbo(promete)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais)))),subst(otimismo),aadnd(aadnd_simples(sp([preposicao(para),sn(pron_subst(você))])))])))]))])))).

estrutura(s16,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(mudanças),aadnd(aadnd_simples(sadj(adj(domésticas))))]))),predicado(predv(svi([verboaux(podem),verbo(ocorrer)])))])))).

estrutura(s17,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(esse)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sn([aadne(sdet(artigo(um))),subst(ponto),aadnd(aadnd_simples(sadj(adj(importante))))])))]))])))).
```

estrutura(s18,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(sol),cn(sp([preposicao(em),sn(subst(sagitário))])])])),predicado(predv([svtd(verbo(ilumina)),od(od_simples(sn([aadne(adj(seus)),subst(relacionamentos)])))])])))).

estrutura(s19,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(sua)),subst(imagem),aadnd(aadnd_simples(sadj(adj(externa))))])])),predicado(predv(svi(verbo(mudará))))])))).

estrutura(s20,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn([aadne(sdet(artigo(o))),subst(trabalho),aadnd(aadnd_simples(sadj(adj(diário))))])),coordenador(e),suj_simples(sn([aadne(sdet(artigo(a))),subst(vida),aadnd(aadnd_simples(sadj(adj(cotidiana))))])])])),predicado(predv([svtd(verbo(exigirão)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais)))),subst(iniciativa),aadnd(aadnd_simples(sp([preposicao(de),sn([aadne(adj(sua)),subst(parte)])])))])])])])))).

estrutura(s21,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(exiba)),od(od_composto([od_simples(sn(subst(misericórdia))),coordenador(e),od_simples(sn(subst(intuição)))])])))))).

estrutura(s22,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn(subst(viagens))),coordenador(e),suj_simples(sn(subst(estudos)))])),predicado(predn([svl(verbo(são)),psuj(psuj_simples(sadj(adj(bem_vindos))))])])))).

estrutura(s23,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(hoje)))),suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(sol),cn(sp([preposicao(em),sn(subst(sagitário))])])])),predicado(predv([svtd(verbo(iluminará)),od(od_simples(sn([aadne(adj(seu)),subst(instinto)])))])])))).

estrutura(s24,frase(periodo(periodo_independente(predicado(predv([svti(verbo(aposte)),oi(oi_simples([poi(nos),sn([aadne(adj(seus)),subst(talentos)])])])))))).

estrutura(s25,frase(periodo(periodo_independente(predicado(predn([svl(verbo(seja)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(versátil)])))])))))).

estrutura(s26,frase(periodo(periodo_independente([predicado(predv([svtd(verbo(comece)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(fase)])))])),aadvo(aadvo_simples(sp([preposicao(com),sn([aadne(sdet(artigo(o))),subst(pé),aadnd(aadnd_simples(sadj(adj(esquerdo))))])])))])))).

estrutura(s27,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(relações),cn(sp([preposicao(com),sn([aadne(sdet(artigo(os))),subst(superiores)])])])))),predicado(predn([svl(verbo(estarão)),psuj(psuj_simples(sadj(adj(críticas))))])])))).

estrutura(s28,frase(periodo(periodo_independente([predicado(predv(svi(verbo(existem)))),aadvo(aadvo_simples(sadv(adv(também)))),suj(suj_simples(sn([aadne(sdet(artigo(os))),subst(desacertos),cn(sp([preposicao(por),sn(subst(distração))])])])))])))).

estrutura(s29,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(novos)),subst(interesses)]))),predicado(predv(svi(verbo(despontam))))])))).

estrutura(s30,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv(svi(verbo(acontecerá)))),aadvo(aadvo_simples(sp([preposicao(com),sn([aadne(adj(alguma)),subst(pressão)])])))])))).

estrutura(s31,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(conflito),cn(sp([preposicao(com),sn([aadne(sdet(artigo(os))),subst(conservadores)])])))])),predicado(predn([svl(verbo(será)),psuj(psuj_simples(sadj(adj(inevitável))))])])))).

estrutura(s32,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(espere)),od(od_simples(sn([subst(mudanças),cn(sp([preposicao(em),sn([aadne(adj(esse)),subst(setor)])])))])))])))))).

estrutura(s33,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(lua),aadnd(aadnd_simples(sadj([adj(crescente),cn(sp([preposicao(em),sn(subst(peixes))])])))])))),predicado(predv([svtd(verbo(sugere)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais)))),subst(atenção),cn(sp([preposicao(às),sn([aadne(adj(suas)),subst(emoções)])])))])))])))).

estrutura(s34,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(exiba)]),od(od_simples(sn([aadne(adj(sua)),subst(impaciência)])))])))))).

estrutura(s35,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(há)]),od(od_simples(sn(subst(trégua))))])))))).

estrutura(s36,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(liberdade)))),predicado(predn([svl([aadvl(aadvl_simples(sadv(adv(não)))),verbo(é)]),psuj(psuj_simples(sn([subst(rebeldia),aadnd(aadnd_simples(sadj(adj(gratuita))))])))])])))).

estrutura(s37,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn(subst(sol))),coordenador(e),suj_simples(sn(subst(plutão)))])),predicado(predn([svl(verbo(estão)),psuj(psuj_simples(sadj(adj(próximos))))])])))).

estrutura(s38,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svti([verboaux(vai),verbo(ceder)]),oi(oi_simples([poi(em),sn([aadne(sdet(artigo(um))),subst(princípio),aadnd(aadnd_simples(sadj(adj(importante))))])])])])))))).

estrutura(s39,frase(periodo(periodo_independente([predicado(predv([svti(verbo(chega)),oi(oi_composto([oi_simples([poi(de),sn(subst(conflito))]),coordenador(e),oi_simples([poi(de),sn(subst(obrigação))])]))])),aadvo(aadvo_simples(sadv([adv(por),aadvl(aadvl_simples(sadv(adv(hoje))))])))))).

estrutura(s40,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(tudo)))),predicado(predn([svl([verboaux(pode),verbo(ser)]),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais))),adj(lindo)]))))])))))).

estrutura(s41,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(esse)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sn([aadne(sdet(artigo(o))),subst(clima)]))))]))))).

estrutura(s42,frase(periodo(periodo_independente(predicado(predv([svti(verbo(cuide)),oi(oi_simples([poi(de),sn([aadne(sdet(artigo(o))),subst(visual)])]))]))))))).

estrutura(s43,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(desdenhe)),od(od_simples(sn([aadne(sdet(artigo(as))),subst(críticas),cn(sp([preposicao(de),sadv(adv(sempre))])])))]))))))).

estrutura(s44,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv([adv(muitas),aadvl(aadvl_simples(sadv(adv(vezes))))]))),suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(só)))),verbo(expressa)]),od(od_simples(sn(subst(desagrado))))])))))).

estrutura(s45,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne([sdet([canonica(todo),artigo(a)]),adj(sua)]),subst(generosidade)]))),predicado(predv(svi(verbo(aparecerá))),aadvo(aadvo_simples(sp([preposicao(com),sn(subst(brilho))])))))))).

estrutura(s46,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd([verboaux(vai),verbo(esquentar)]),od(od_simples(sn([aadne(adj(qualquer)),subst(programa)])))])))))).

estrutura(s47,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(terá)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais))),subst(ânimo)]))))]))))).

estrutura(s48,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(dia)]))),predicado(predn([svl([verboaux(pode),verbo(ser)]),psuj(psuj_simples(sadj([adj(bacana),cn(sp([preposicao(para),sn(pron_subst(você))]))]))))])))))).

estrutura(s49,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd(verbo(diminui),od(od_simples(sn([aadne(sdet(artigo(a))),subst(tensão)]))))]))))).

estrutura(s50,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd(verbo(espanta)),od(od_simples(sn([aadne(sdet(artigo(as))),subst(sombras)]))))]))))).

estrutura(s51,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(dia),aadnd(aadnd_simples(sp([preposicao(de),sadv(adv(sábado))]))))]))),predicado(predv([svtd(verbo(promete)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais)))),subst(agitação)]))))]))))).

estrutura(s52,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(dia)]))),predicado(predv([svtd(verbo(favorece)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(troca),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(informações))]))))]))))]))))).

estrutura(s53,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(hoje)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sn([subst(tempo),aadnd(aadnd_simples(sp([preposicao(para),sn(subst(meditação))]))))]))))]))))).

estrutura(s54,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd(verbo(fará)),od(od_simples(sn([aadne(adj(ótimo)),subst(efeito),cn(sp([preposicao(sobre),sn([aadne(adj(seu)),subst(humor)])]))]))))]))))).

estrutura(s55,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(talvez)))),suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl([verboaux(precise),verbo(estar)]),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(presente),cn(sp([preposicao(em),sn([aadne(sdet(artigo(a))),subst(vida),aadnd(aadnd_simples(sadj(adj(familiar))))])]))]))))]))))).

estrutura(s56,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn(subst(irritação))),coordenador(e),suj_simples(sn([subst(variação),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(humor))])))])]))),predicado(predv([svtd([verboaux(podem),verbo(estragar)]),od(od_simples(sn([aadne(adj(seu)),subst(dia)]))))]))))).

estrutura(s57,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(essa)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sn([aadne(sdet(artigo(a))),subst(verdade)]))))]))))).

estrutura(s58,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(palavras),aadnd(aadnd_simples(sadj(adj(ríspidas))))]))),predicado(predv([svti(verbo(acabam)),oi(oi_simples([poi(com),sn([aadne([sdet(artigo(as)),adj(boas)]),subst(intenções)])])])])))))).

estrutura(s59,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(faça)),od(od_simples(sn([aadne(adj(suas)),subst(intervenções),aadnd(aadnd_simples(sadj(adj(apropriadas))))]))))])))).

estrutura(s60,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(sua)),subst(força)]))),predicado(predn([svl(verbo(será)),psuj(psuj_simples(sn([aadne(sdet(artigo(o))),subst(sonho),cn(sp([preposicao(com),sn([subst(mundos),aadnd(aadnd_simples(sadj(adj(melhores))))])])))])))])))).

estrutura(s61,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(mantenha)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(crença)]))))])))).

estrutura(s62,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(ninguém)))),predicado(predv([svtd([verboaux(pode),verbo(roubar)]),od(od_composto([od_simples(sn([aadne(adj(sua)),subst(força)])),coordenador(e),od_simples(sn([aadne(adj(sua)),subst(determinação)]))]))])))])))).

estrutura(s63,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([verboaux(vai),verbo(dar)]),od(od_simples(sn([subst(lições),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(vivência))]))])))]))])))).

estrutura(s64,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([verboaux(precisará),verbo(usar)]),od(od_simples(sn([aadne(sdet(artigo(a))),subst(mente)]))]))])))).

estrutura(s65,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn(subst(marte))),coordenador(e),suj_simples(sn(subst(urano)))])),predicado(predv([svtd(verbo(atrapalham)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(clima)]))])]))))).

estrutura(s66,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(cenas),aadnd(aadnd_simples(sp([preposicao(de),sn([subst(drama),cn(sp([preposicao(em),sn([aadne(sdet(artigo(a))),subst(vida),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(casal))]))])])])])])])])))),predicado(predv([svti(verbo(estão)),oi(oi_simples([poi(em),sn([aadne(sdet(artigo(o))),subst(horizonte)])]))])])))).

estrutura(s67,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predn([svl([verboaux(pode),verbo(ser)]),psuj(psuj_simples(sadj(adj(irreversível))))])])))).

estrutura(s68,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(lua)]))),predicado(predv([svtd(verbo(atravessa)),od(od_simples(sn(subst(libra))))])])))).

estrutura(s69,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(seu)),subst(olhar)]))),predicado(predv([svtd(verbo(reflete)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(mundo)]))])])))).

estrutura(s70,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(ali)))),suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(formas)]))),predicado(predv(svi(verbo(oscilam))),aadvo(aadvo_simples(sadv(adv(demais))))])))).

estrutura(s71,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(vê)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(reflexo),aadnd(aadnd_simples(sp([preposicao(de),sn([aadne(sdet(artigo(a))),subst(pressão),aadnd(aadnd_simples(sadj(adj(externa))))])])))])))])))).

estrutura(s72,frase(periodo(periodo_independente([suj(osss(oss(ori(periodo(periodo_independente(predicado(predv([svtd(verbo(fixar)),od(od_simples(sn([aadne(adj(poucos)),subst(objetivos)]))])))))))),predicado(predn([svl(verbo(seria)),psuj(psuj_simples(sadj(adj(bom))))])])))).

estrutura(s73,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predn([svl(verbo(seria)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(realista)]))])])))).

estrutura(s74,frase(periodo(periodo_independente([suj(osss(oss(ori(periodo(periodo_independente([predicado(predv(svi(verbo(pretender))))),aadvo(aadvo_simples(sadv(adv(mais))))])))))),predicado(predv([svtd(verbo(aumentaria)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(adrenalina)]))])])))).

estrutura(s75,frase(periodo(periodo_independente(predicado(predv([svti(verbo(divirja)),oi(oi_simples([poi(de),sn([aadne(sdet(artigo(o))),subst(comum)])])])))))).

estrutura(s76,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sadj([adj(capaz),cn(osscn([pcn(de),oss(ori(periodo(periodo_independente(predicado(predv([svtd(verbo(conquistar)),od(od_simples(sn([aadne(sdet(artigo(um))),subst(espaço),aadnd(aadnd_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais))),adj(confortável)]))))])))))))])))])))])))).

estrutura(s77,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sadj(adj(hábil))))])])))).

estrutura(s78,frase(periodo(periodo_independente([predicado(predv([svtd(verbo(deixe)),od(od_simples(sn([aadne(sdet(artigo(as))),subst(preocupações)]))))])),aadvo(aadvo_simples(sadv(adv(longe))))])))).

estrutura(s79,frase(periodo(periodo_independente(predicado(predv([svti(verbo(cuide)),oi(oi_simples([poi(de),sn([aadne(sdet(artigo(o))),subst(cansaço)])])]))))))).

estrutura(s80,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(desprezе)]),od(od_simples(sn([aadne(sdet(artigo(os))),subst(sinais),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(tristeza))]))))]))])))))).

estrutura(s81,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(carga)]))),predicado(predv(svi(verbo(aumentou))))])))).

estrutura(s82,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predvn([svtd(verbo(deixou)),od(od_simples(sn(pron_subst(você)))),pobj(sadj(adj(impaciente)))])])))).

estrutura(s83,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(pressão)]))),predicado(predv([svtdi([verboaux(pode),verbo(empurrar)]),od(od_simples(sn(pron_subst(você)))),oi(oi_simples(sp([preposicao(para),sn([aadne(sdet(artigo(a))),subst(solidão)])]))])])))))).

estrutura(s84,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([verboaux(pode),verbo(escolher)]),od(od_composto([od_simples(sn([aadne(sdet(artigo(o))),subst(deserto)])),coordenador(ou),od_simples(sn([aadne(sdet(artigo(um))),subst(palco),aadnd(aadnd_simples(sadj(adj(suspeitíssimo))))])])])])))))).

estrutura(s85,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(hoje)))),predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(há)]),od(od_simples(sn([aadne(adj(meio)),subst(termo)]))])])))))).

estrutura(s86,frase(periodo(periodo_independente(predicado(predn([svl(verbo(esteja)),psuj(psuj_simples(sadj([adj(aberto),cn(sp([preposicao(a),sn([aadne(sdet(artigo(os))),subst(ventos),aadnd(aadnd_simples(sp([preposicao(de),sn([aadne(sdet(artigo(a))),subst(mudança)])])))])])])])])))))))).

estrutura(s87,frase(periodo(periodo_independente([suj(suj_simples(lexico(eles))),predicado(predv(svi([verboaux(estão),verbo(soprando)]))),aadvo(aadvo_simples(sp([preposicao(a),sn([aadne([sdet(artigo(a)),adj(sua)]),subst(volta)])]))))])))).

estrutura(s88,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(algo)))),predicado(predv(svi([verboaux(pode),verbo(acontecer)]))))])))).

estrutura(s89,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(algo)))),predicado(predv(svi([verboaux(pode),verbo(acontecer)]))),aadvo(aadvo_simples(sp([preposicao(para),sn([aadne(adj(sua)),subst(alegria)])]))))])))).

estrutura(s90,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(trabalho),aadnd(aadnd_simples(sadj(adj(cotidiano))))])))),predicado(predv([svti(verbo(precisa)),oi(oi_composto([oi_simples([poi(de),sn(subst(inovação))]),coordenador(e),oi_simples([poi(de),sn(subst(mudança))])]))])))))).

estrutura(s91,frase(periodo(periodo_independente(predicado(predv([svtdi(verbo(dedique)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(dia)])))),oi(ossoi([poi(para),oss(ori(periodo(periodo_independente(predicado(predv([svti(verbo(tratar)),oi(oi_simples([poi(de),sn([aadne([sdet(artigo(as)),adj(suas)]),subst(coisas)])])])))))))])))))))).

estrutura(s92,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(resto)]))),predicado(predv(svi([aadvl(aadvl_simples(sadv(adv(não)))),verbo(importa)]))),aadvo(aadvo_simples(sadv(adv(muito))))])))).

estrutura(s93,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(seu)),subst(poder),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(barganha))]))))])))),predicado(predn([svl(verbo(está)),psuj(psuj_simples(sadj(adj(limitado))))])))))).

estrutura(s94,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(distração)]))),predicado(predv(svi(verbo(aumenta))))])))).

estrutura(s95,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([verboaux(pode),verbo(dar)]),od(od_simples(sn([aadne(sdet(artigo(um))),subst(passo),cn(sp([preposicao(em),sn(subst(falso))])])))])))])))).

estrutura(s96,frase(periodo(periodo_independente(predicado(predn([svl(verbo(fique)),psuj(psuj_simples(sadj(adj(atento))))])))))).

estrutura(s97,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(suas)),subst(antenas)]))),predicado(predv([svtd(verbo(captam)),od(od_simples(sn([pron_subst(algo),aadnd(aadnd_simples(sadj(adj(novo))))])))])))))).

estrutura(s98,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne([sdet(artigo(a)),adj(sua)]),subst(curiosidade)]))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sadj(adj(satisfeita))))])),aadvo(aadvo_simples(sadv(adv(sempre))))])))).

estrutura(s99,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(vida),aadnd(aadnd_simples(sadj(adj(social))))]))),predicado(predn([svl(verbo(está)),psuj(psuj_simples(sadj(adj(eminente))))])),aadvo(aadvo_simples(sadv(adv(hoje))))])))).

estrutura(s100,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(irmão),aadnd(aadnd_simples(sadj(adj(próximo))))]))),predicado(predv([svti(verbo(precisa)),oi(oi_simples([poi(de),sn(subst(atenção))]))]))])))).

estrutura(s101,frase(periodo(periodo_independente([suj(osss(oss(ori(periodo(periodo_independente(predicado(predv([svtd(verbo(adotar)),od(od_simples(sn([aadne(sdet(artigo(uma))),subst(postura),aadnd(aadnd_simples(sadj(adj(aberta))))]))))]))))))))),predicado(predn([svl(verbo(seria)),psuj(psuj_simples(sadj(adj(bom))))]))])))).

estrutura(s102,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(lua)]))),predicado(predv([svtd(verbo(transita)),od(od_simples(sn(subst(touro))))]))])))).

estrutura(s103,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(mantenha)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(instabilidade),aadnd(aadnd_simples(sadj(adj(controlada))))]))))])))))).

estrutura(s104,frase(periodo(periodo_independente([predicado(predv(svi(verbo(reflita)))),aadvo(aadvo_simples(sadv([adv(mais),aadvl(aadvl_simples(sadv([adv(antes),cn(osscn([pcn(de),oss(ori(periodo(periodo_independente(predicado(predv(svi(verbo(falar))))))))]))]))]))]))])))).

estrutura(s105,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(hoje)))),suj(suj_simples(sn([aadne([sdet(artigo(os)),adj(seus)]),subst(desejos)]))),predicado(predv(svi([verboaux(podem),verbo(aparecer)]))),aadvo(aadvo_simples(sp([preposicao(com),sn([aadvl(aadvl_simples(sadv(adv(mais))),subst(força)])]))]))])))).

estrutura(s106,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(talvez)))),suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(fique)),psuj(psuj_simples(sadj([adj(ligado),cn(sp([preposicao(em),sn([aadne(sdet(artigo(a))),subst(insatisfação)])]))]))]))])))).

estrutura(s107,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(espreite)),od(od_simples(sn([aadne(sdet(artigo(os))),subst(ouvidos)]))]))])))).

estrutura(s108,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(tempo)]))),predicado(predv([svtd([verboaux(pode),verbo(trazer)]),od(od_simples(sn([aadne(sdet(artigo(uma))),subst(chance)]))]))]))))).

estrutura(s109,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(use)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(teimosia),aadnd(aadnd_simples(sadj([adj(dosada),cn(sp([preposicao(com),sn([aadne(sdet(artigo(a))),subst(suavidade)])]))]))]))]))])))).

estrutura(s110,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn([aadne(sdet(artigo(o))),subst(toque)])),coordenador(e),suj_simples(sn([aadne(sdet(artigo(a))),subst(presença)]))])),predicado(predn([svl(verbo(serão)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais))),adj(eficientes)]))]))]))))).

estrutura(s111,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(inesperado)]))),predicado(predv([svti(verbo(está)),oi(oi_simples([poi(em),sn([aadne(sdet(artigo(o))),subst(horizonte)])]))]))])))).

estrutura(s112,frase(periodo(periodo_independente([suj(suj_simples([sn(nome_proprio(mercúrio)),virgula(,),aposto(sn([aadne(sdet(artigo(o))),nome_proprio(hermes),aadnd(aadnd_simples(sadj(adj(grego))))])),virgula(,)])),predicado(predv(svi(verbo(trabalha)))),aadvo(aadvo_simples(sp([preposicao(para),sn(pron_subst(você))]))]))])))).

estrutura(s113,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(anote)),od(od_composto([od_simples(sn(subst(sonhos))),coordenador(e),od_simples(sn(subst(idéias)))])]))])))).

estrutura(s114,frase(periodo(periodo_independente(predicado(predv([svti(verbo(capriche)),oi(oi_simples([poi(em),sn([aadne(sdet(artigo(o))),subst(visual)])]))]))])))).

estrutura(s115,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(colegas),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(trabalho))]))))])),predicado(predv([svtd([verboaux(podem),verbo(confundir)]),od(od_simples(sn([aadne(sdet(artigo(as))),subst(coisas)]))))]))])))).

estrutura(s116,frase(periodo(periodo_independente([suj(suj_simples(lexico(eles))),predicado(predv([svtd([verboaux(podem),verbo(atrapalhar)]),od(od_simples(sn([aadne(adj(seu)),subst(cotidiano)]))))]))])))).

estrutura(s117,frase(periodo(periodo_independente([aadvo(aadvo_composto([aadvo_simples(sadv(adv(hoje))),coordenador(e),aadvo_simples(sadv(adv(amanhã)))])),suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(terá)),od(od_simples(sn([aadne(sdet(artigo(um))),subst(dom),aadnd(aadnd_simples(sadj(adj(especial))))]))])]))])))).

estrutura(s118,frase(periodo(periodo_independente(predicado(predv([svti(verbo(confie)),oi(oi_composto([oi_simples([poi(em),sn([aadne([sdet(artigo(a)),adj(sua)]),subst(competência)])])],coordenador(e),oi_simples([poi(em),sn([aadne([sdet(artigo(a)),adj(sua)]),subst(experiência)])])])))])))).

estrutura(s119,frase(periodo(periodo_independente(predicado(predn([svl(verbo(fique)),psuj(psuj_simples(sadj([adj(atento),cn(sp([preposicao(a),sn([aadne(sdet(artigo(as))),subst(notícias)])]))]))]))])))).

```
estrutura(s120,frase(periodo(periodo_independente(predicado(predn([svl(verbo(fique)),psuj(psuj_simples(sadj([adj(atento),cn(s
p([preposicao(a),sn([aadne(sdet(artigo(o))),subst(ambiente)])]))])))]))))))).

estrutura(s121,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(sua)),subst(mente)]))),predicado(predv([svti
([verboaux(pode),verbo(tratar)]),oi(oi_simples([poi(de),sn([aadne(adj(muitos)),subst(assuntos)])]))])]))))).

estrutura(s122,frase(periodo(periodo_independente(predicado(predn([svl(verbo(continue)),psuj(psuj_simples(sadj(adj(indomáv
el))))])))))).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                    estrutura sintatica das sentencas do corpus testes indexadas                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

estrutura(t1,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(levante)),od(od_simples(sn([aadne(sdet(artigo(a)
)),subst(cabeça)])))])))))).

estrutura(t2,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(conhece)
),od(od_simples(sn([aadne(sdet(artigo(o))),subst(campo),cn(sp([preposicao(de),sn([aadne(adj(sua)),subst(consciência)])]))])))]))
])))).

estrutura(t3,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(é)),psuj(p
suj_simples(sn([aadne(sdet(artigo(o))),subst(templo),aadnd(aadnd_simples(sadj(adj(sagrado))))])))]))])))).

estrutura(t4,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(experiências)]))),predicado(
predv(svi(verbo(crescem))))])))).

estrutura(t5,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(veja)),od(od_simples(sn([aadne(adj(sua)),subst(e
xpressão),aadnd(aadnd_simples(sadj(adj(radiante))))])))])))))).

estrutura(t6,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(subestime
)]),od(od_simples(sn([aadne(adj(seus)),subst(silêncios)])))])))))).

estrutura(t7,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(subestime
)]),od(od_simples(sn([aadne(sdet(artigo(os))),subst(períodos),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(inatividade))])
))])))])))))).

estrutura(t8,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd(verbo(perpetua)
),od(od_simples(sn([aadne(sdet(artigo(a))),subst(ignorância)])))]))])))).

estrutura(t9,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([verboaux(preci
sa),verbo(destruir)]),od(od_simples(sn([aadne(sdet(artigo(o))),subst(anterior)])))]))])))).

estrutura(t10,frase(periodo(periodo_independente([suj(osss(oss(ori(periodo(periodo_independente(predicado(predv([svtd(verbo
(ocultar)),od(od_simples(sn([aadne(sdet(artigo(os))),subst(sentimentos),aadnd(aadnd_simples(sadj(adj(bons))))])))])))))))))),predi
cado(predn([svl(verbo(é)),psuj(psuj_simples(sadj(adj(ruim))))]))])))).

estrutura(t11,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(nada)))),predicado(predn([svl(verbo(é)),psuj(
psuj_simples(sadj(adj(fácil))))]))])))).

estrutura(t12,frase(periodo(periodo_independente([predicado(predv([svtd(verbo(encare)),od(od_simples(sn([aadne(sdet(artigo(
os))),subst(acontecimentos)])))])),aadvo(aadvo_simples(sp([preposicao(com),sn([aadne(sdet(artigo(o))),subst(humor),aadnd(aa
dnd_simples(sadj(adj(bom))))])])))])))).

estrutura(t13,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd(verbo(transfor
mará)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(realidade)])))]))])))).

estrutura(t14,frase(periodo(periodo_independente([suj(suj_simples(sn(sn(beleza)))),predicado(predn([svl(verbo(é)),psuj(psuj_si
mples(sadj(adj(fundamental))))]))])))).

estrutura(t15,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(atraia)),od(od_composto([od_simples(sn(subst(r
ecursos))),coordenador(e),od_simples(sn(subst(associados)))]))])))))).

estrutura(t16,frase(periodo(periodo_independente([suj(suj_simples(lexico(eles))),predicado(predv([svtd([verboaux(podem),verb
o(mudar)]),od(od_simples(sn([aadne(adj(sua)),subst(vida)])))]))])))).

estrutura(t17,frase(periodo(periodo_independente(predicado(predv([svti(verbo(mexa)),oi(oi_simples([poi(em),sn([aadne(sdet(art
igo(as))),subst(coisas)])]))])))))).
```

estrutura(t18,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(sucesso),aadnd(aadnd_simples(sadj(adj(pessoal))))]))),predicado(predn([svl(verbo(está)),psuj(psuj_simples(sadj(adj(eminente))))])),aadvo(aadvo_simples(sadv(adv(hoje))))])))).

estrutura(t19,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(aproveite)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(sentimento)]))))])))).

estrutura(t20,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(realize)),od(od_composto([od_simples(sn(subst(estudos))),coordenador(e),od_simples(sn(subst(pesquisas)))])))])))).

estrutura(t21,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você))))],predicado(predv([svtd([verboaux(precisará),verbo(fazer)]),od(od_simples(sn([aadne(sdet(artigo(uma))),subst(parte)]))))]))))).

estrutura(t22,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne([sdet(artigo(a)),adj(sua)]),subst(comunicação)]))),predicado(predn([svl(verbo(está)),psuj(psuj_simples(sadj(adj(clara))))])),aadvo(aadvo_simples(sadv(adv(hoje))))])))).

estrutura(t23,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso))))],predicado(predvn([svtd(verbo(deixou)),od(od_simples(sn(pron_subst(você))))],pobj(sadj(adj(tranqüilo)))])))))).

estrutura(t24,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(controle)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(perfil),aadnd(aadnd_simples(sadj(adj(melodramático))))]))))])))).

estrutura(t25,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(supere)),od(od_simples(sn([aadne(sdet(artigo(o))),subst(retraimento)]))))])))).

estrutura(t26,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(sua)),subst(capacidade),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(adaptação))]))))]))),predicado(predn([svl(verbo(é)),psuj(psuj_simples(sadj(adj(imediata))))])))))).

estrutura(t27,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(seu)),subst(romantismo)]))),predicado(predv([svtd(verbo(encontra)),od(od_simples(sn([subst(receptividade),aadnd(aadnd_simples(sadj(adj(expressiva))))]))))])))))).

estrutura(t28,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(urano),cn(sp([preposicao(em),sn(subst(sagitário))]))]))),predicado(predv([svtd(verbo(acentua)),od(od_simples(sn([aadne(adj(seu)),subst(magnetismo)]))))])))))).

estrutura(t29,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(privilegie)),od(od_simples(sn([aadne(sdet(artigo(as))),subst(atividades),aadnd(aadnd_simples(sadj(adj(criativas))))]))))])))).

estrutura(t30,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(busque)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(natureza)]))))])))).

estrutura(t31,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(dê)),od(od_simples(sn([subst(sugestões),cn(sp([preposicao(em),sn([aadne(adj(essa)),subst(área)])]))]))))])))).

estrutura(t32,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(circunstâncias)]))),predicado(predv([svtd(verbo(favorecem)),od(od_simples(sn([aadne(sdet(artigo(um))),subst(avanço),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(profissão))]))))]))))])))))).

estrutura(t33,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(sua)),subst(reputação)]))),predicado(predv([svtd(verbo(joga)),od(od_simples(sn([subst(papel),aadnd(aadnd_simples(sadj(adj(importante))))]))))])))))).

estrutura(t34,frase(periodo(periodo_independente(predicado(predn([svl(verbo(seja)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(maleável)]))))])))).

estrutura(t35,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(anote)),od(od_simples(sn([aadne(sdet(artigo(os))),subst(sonhos),cn(sp([preposicao(de),sadv(adv(hoje))]))))]))))))).

estrutura(t36,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso))))],predicado(predn([svl([verboaux(pode),verbo(ser)]),psuj(psuj_simples(sadj(adj(importante))))])))))).

estrutura(t37,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn(subst(criatividade))),coordenador(e),suj_simples(sn(subst(expressão)))]))],predicado(predv([svtd(verbo(facilitam)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(comunicação)]))))]))))))).

estrutura(t38,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(essa))))],predicado(predn([svl(verbo(será)),psuj(psuj_simples(sn([aadne(sdet(artigo(uma))),subst(arma),aadnd(aadnd_simples(sadj(adj(valiosa))))]))))]))))).

estrutura(t39,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(evite)),od(od_simples(sn([aadne(sdet(artigo(as))),subst(cobranças)]))))])))).

estrutura(t40,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(comunicação),cn(sp([preposicao(com),sn([aadne(sdet(artigo(a))),subst(amada)])])])),predicado(predn([svl(verbo(estará)),psuj(psuj_simples(sadj(adj(favorecida))))])])])))).

estrutura(t41,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(exerça)),od(od_simples(sn([aadne(adj(seu)),subst(poder),cn(sp([preposicao(de),sn(subst(sedução))])])))])))))).

estrutura(t42,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(amanhã)))),suj(suj_simples(sn([aadne(sdet(artigo(a))),subst(lua),cn(sp([preposicao(em),sn(subst(marte))])])))),predicado(predv([svtd(verbo(aumenta)),od(od_simples(sn([aadne(adj(sua)),subst(coragem)])))])])))).

estrutura(t43,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv([svtd([verboaux(vai),verbo(aquecer)]),od(od_simples(sn([aadne(adj(qualquer)),subst(programa)])))])])))).

estrutura(t44,frase(periodo(periodo_independente(predicado(predn([svl(verbo(seja)),psuj(psuj_simples(sadj([adj(dedicado),cn(sp([preposicao(a),sn([aadne(sdet(artigo(a))),subst(amada)])])))])))])))).

estrutura(t45,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(use)),od(od_simples(sn([aadne(adj(seu)),subst(dom),cn(sp([preposicao(de),sn(subst(comunicação))])))])))])))).

estrutura(t46,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(oscilações),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(humor))])))])))),predicado(predv([svtd([verboaux(podem),verbo(prejudicar)]),od(od_simples(sn([aadne(sdet(artigo(os))),subst(contatos)])))])])))).

estrutura(t47,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(tem)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais)))),subst(determinação)])))])])))).

estrutura(t48,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predn([svl(verbo(será)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(benéfico)])))])])))).

estrutura(t49,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([verboaux(poderá),verbo(curtir)]),od(od_composto([od_simples(sn([aadne(sdet(artigo(a))),subst(tristeza)])),coordenador(ou),od_simples(sn([aadne(sdet(artigo(os))),subst(momentos),aadnd(aadnd_simples(sadj(adj(gratificantes))))])])])])))))).

estrutura(t50,frase(periodo(periodo_independente(predicado(predv([svti(verbo(confie)),oi(oi_composto([oi_simples([poi(em),sn([aadne([sdet(artigo(o)),adj(seu)]),subst(talento)])]),coordenador(e),oi_simples([poi(em),sn([aadne([sdet(artigo(a)),adj(sua)]),subst(força)])])])])])))))).

estrutura(t51,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(isso)))),predicado(predv(svi([verboaux(pode),verbo(despontar)])))])))).

estrutura(t52,frase(periodo(periodo_independente([aadvo(aadvo_composto([aadvo_simples(sadv(adv(hoje))),coordenador(e),aadvo_simples(sadv(adv(amanhã)))])),suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(esperará)),od(od_simples(sn([aadne(sdet(artigo(um))),subst(conselho),aadnd(aadnd_simples(sadj(adj(proveitoso))))])])])])))).

estrutura(t53,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(use)),od(od_composto([od_simples(sn(subst(talentos))),coordenador(e),od_simples(sn(subst(capacidades)))])])))))).

estrutura(t54,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(seu)),subst(poder),aadnd(aadnd_simples(sp([preposicao(de),sn(subst(comunicação))])))])),predicado(predn([svl(verbo(está)),psuj(psuj_simples(sadj(adj(favorecido))))])])))))).

estrutura(t55,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(os))),subst(contatos)]))),predicado(predn([svl([verboaux(poderão),verbo(ser)]),psuj(psuj_simples(sadj([adj(agradáveis),cn(sp([preposicao(para),sn(pron_subst(você))])])))])])))))).

estrutura(t56,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(desperdice)]),od(od_simples(sn([aadne(adj(sua)),subst(oportunidade)])))])))))).

estrutura(t57,frase(periodo(periodo_independente([suj(suj_composto([suj_simples(sn([aadne(sdet(artigo(o))),subst(ciúme)])),coordenador(e),suj_simples(sn([aadne(sdet(artigo(a))),subst(possessão)])])),predicado(predn([svl(verbo(estarão)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(atenuados)])))])])))))).

estrutura(t58,frase(periodo(periodo_independente(predicado(predv([svtd(verbo(use)),od(od_simples(sn([aadne(adj(seu)),subst(dom),cn(sp([preposicao(de),sn([subst(comunicação),aadnd(aadnd_simples(sadj(adj(elevada))))])])))])))))))).

estrutura(t59,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(sua)),subst(intuição)]))),predicado(predv([svtd(verbo(apontará)),od(od_simples(sn([aadne(sdet(artigo(os))),subst(caminhos)])))])])))).

estrutura(t60,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(verá)),od(od_simples(sn([aadne(sdet(artigo(um))),subst(esboço),aadnd(aadnd_simples(sp([preposicao(de),sn([aadne(sdet(artigo(as))),subst(pessoas),aadnd(aadnd_simples(sadj(adj(próximas))))])])))])])))])))).

estrutura(t61,frase(periodo(periodo_independente(predicado(predv([svti(verbo(cuide)),oi(oi_simples([poi(de),sn([aadne(sdet(artigo(a))),subst(saúde)])])))])))).

estrutura(t62,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(o))),subst(fim_de_semana)]))),predicado(predn([svl(verbo(será)),psuj(psuj_simples(sadj(adj(agitado))))])])))).

estrutura(t63,frase(periodo(periodo_independente(predicado(predn([svl(verbo(esteja)),psuj(psuj_simples(sadj([adj(receptiva),cn(sp([preposicao(a),sn([aadne(sdet(artigo(os))),subst(clamores),aadnd(aadnd_simples(sp([preposicao(de),sn([aadne(sdet(artigo(a))),subst(novidade)])])))])])))])))])))))).

estrutura(t64,frase(periodo(periodo_independente([suj(osss(oss(ori(periodo(periodo_independente([predicado(predv(svi(verbo(viajar))))),aadvo(aadvo_simples(sadv(adv(mais))))])))))),predicado(predv([svtd(verbo(favorecerá)),od(od_simples(sn([aadne(sdet(artigo(a))),subst(paixão)])))])])))).

estrutura(t65,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(sdet(artigo(as))),subst(manifestações)]))),predicado(predv([svtd([verboaux(poderiam),verbo(gerar)]),od(od_simples(sn([aadne(sdet(artigo(os))),subst(conflitos)])))])])))).

estrutura(t66,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv(svi(verbo(partirá)))),aadvo(aadvo_simples(sp([preposicao(com),sn([aadne(adj(boa)),subst(clareza)])]))])])))).

estrutura(t67,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(estará)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(introspectiva)])))])])))).

estrutura(t68,frase(periodo(periodo_independente(predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(haverá)]),od(od_simples(sn(subst(otimismo))))])))))).

estrutura(t69,frase(periodo(periodo_independente([suj(suj_simples(sn(subst(sexta)))),predicado(predn([svl(verbo(será)),psuj(psuj_simples(sn([aadne(sdet(artigo(um))),subst(dia),aadnd(aadnd_simples(sadj(adj(difícil))))])))])])))).

estrutura(t70,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(tudo)))),predicado(predn([svl([verboaux(poderá),verbo(ser)]),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(divertido)])))])])))).

estrutura(t71,frase(periodo(periodo_independente(predicado(predv([svti(verbo(acredite)),oi(oi_simples([poi(nos),sn([aadne(adj(seus)),subst(sonhos)])])))])))).

estrutura(t72,frase(periodo(periodo_independente([predicado(predv([svtd(verbo(responda)),od(od_simples(sn([aadne(sdet(artigo(as))),subst(perguntas)])))])),aadvo(aadvo_simples(sadv(adv(cordialmente))))])))).

estrutura(t73,frase(periodo(periodo_independente(predicado(predn([svl(verbo(fique)),psuj(psuj_simples(sadj(adj(calmo))))])))))).

estrutura(t74,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(tudo)))),predicado(predn([svl(verbo(estará)),psuj(psuj_simples(sadj([aadvl(aadvl_simples(sadv(adv(mais)))),adj(favorável),cn(sp([preposicao(com),sn([aadne(adj(seus)),subst(familiares),aadnd(aadnd_simples(sadj(adj(importantes))))])])])))])])))))).

estrutura(t75,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv(adv(hoje)))),predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(não)))),verbo(há)]),od(od_simples(sn([aadne(adj(certo)),subst(descontrole)])))])])))).

estrutura(t76,frase(periodo(periodo_independente([suj(suj_simples(sn([subst(responsabilidades),aadnd(aadnd_simples(sadj(adj(novas))))]))),predicado(predv(svi([verboaux(podem),verbo(chegar)])))])))).

estrutura(t77,frase(periodo(periodo_independente([aadvo(aadvo_simples(sadv([adv(muitas),aadvl(aadvl_simples(sadv(adv(vezes))))]))),suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd([aadvl(aadvl_simples(sadv(adv(só)))),verbo(mostrará)]),od(od_simples(sn(subst(alegrias))))])])))).

estrutura(t78,frase(periodo(periodo_independente([suj(suj_simples(sn([aadne(adj(novas)),subst(oportunidades)]))),predicado(predv(svi(verbo(surgem))))])))).

estrutura(t79,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predv([svtd(verbo(tem)),od(od_simples(sn([aadvl(aadvl_simples(sadv(adv(mais)))),subst(controle)])))])])))).

estrutura(t80,frase(periodo(periodo_independente([suj(suj_simples(sn(pron_subst(você)))),predicado(predn([svl(verbo(está)),psuj(psuj_simples(sadj(adj(charmoso))))])])))).

4 Repositório Conceitual

O Repositório Conceitual fornece os dois componentes para o mapeamento sintático-conceitual: (a) uma gramática livre de contexto de regras de projeção, para mapear cada constituinte sintático em um constituinte conceitual, associado ao seu papél semântico, produzindo uma estrutura conceitual intermediária; e (b) um conjunto de *templates* de relacionamento, para combinar cada par de constituintes conceituais da estrutura conceitual intermediária, de acordo com seus papéis semânticos, em uma relação binária UNL. O esquema geral da aplicação desses dois componentes é ilustrado na Figura 2.

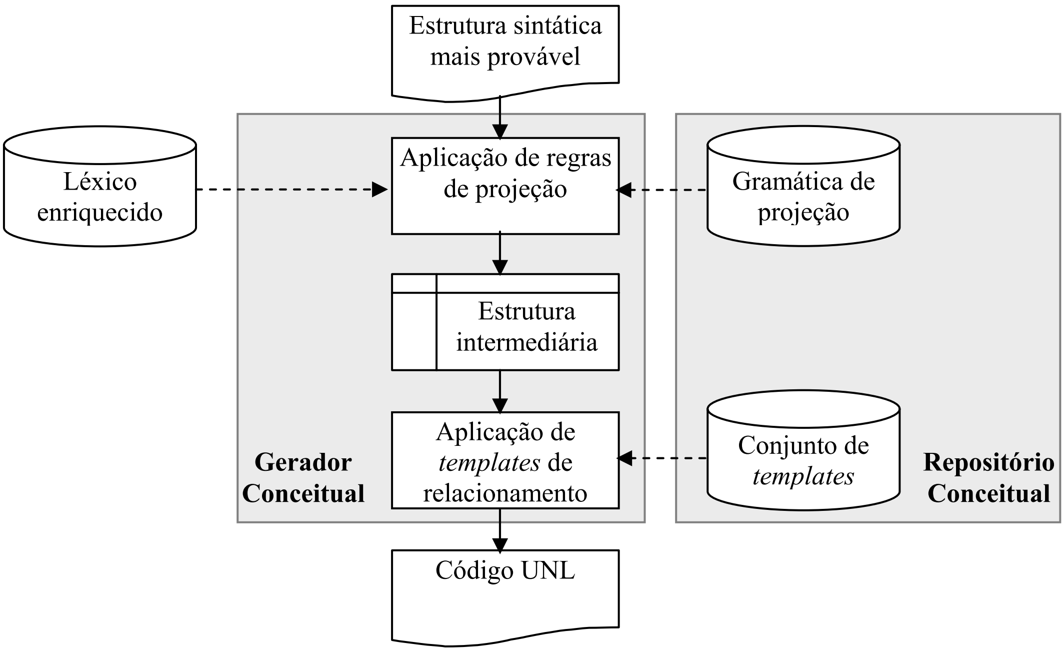


Figura 2 – Os módulos de geração conceitual do ConPor

4.1 Regras de Projeção

As regras de projeção são divididas, aqui, de acordo com o tipo de constiuinte que mapeiam (sentença toda, sujeito, predicado, etc.), incluindo **regras intermediárias** (somente decompõem a estrutura sintática de entrada e disparam outras regras) e **regras terminais** (acessam o léxico enriquecido para mapear as palavras em conceitos com seus papéis semânticos). Ao todo, são **292** regras, sendo **104** intermediárias e **188** terminais. Na listatem a seguir, como comentário, além da sintaxe geral das regras, são indicados os códigos das sentenças para as quais cada regra é aplicada.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                         Regras de projecao para sentencas - sintaxe                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% nome_regra(Estrutura_sintatica,Estrutura_UNL) :- regra_para_constituinte1(Argumentos,Conceito1),
%                                                  regra_para_constituinte2(Argumentos,Conceito2),
%                                                                  ...
%                                                  regra_para_constituinten(Argumentos,Conceiton),
%                                                  gera(estrutura_conceitual_intermediaria,Estrutura_UNL).


% "gera" dispara os templates de relacionamento, passando como argumentos os conceitos com seus papeis semânticos
% retornados pelas regras para os constituintes (a estrutura conceitual intermediaria), para produzir a estrutura UNL final.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                    Regras de projecao para sentenças                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S1, S2, S3, S4, S7, S21, S24, S34, S42, S43, S59, S61, S79, S80, S91, S103, S107, S109, S113
r_Acao(frase(periodo(periodo_independente(predicado(PRED)))),UNL) :-
        verificaPredAcao(PRED,RestSuj,ListaConceitos),
        gera(ListaConceitos,UNL).

% S11, S14, S18, S33, S46, S49, S52, S54, S56, S58, S62, S63, S64, S65, S68, S69, S71, S74, S82, S83, S84, S95,
% S97, S102, S108, S115, S116, S121
r_Acao(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED)]))),UNL) :-
        verificaPredAcao(PRED,RestSuj,ListaPred),
        verificaSujAcao(SUJ,RestSuj,ListaSuj),
        gera([ListaSuj,ListaPred],UNL).

% S23, S44
r_Acao(frase(periodo(periodo_independente([aadvo(AADVO),suj(SUJ),predicado(PRED)]))),UNL) :-
        verificaPredAcao(PRED,RestSuj,ListaPred),
        verificaSujAcao(SUJ,RestSuj,ListaSuj),
        verificaAadvo(AADVO,ListaAadvo),
        concatena([ListaSuj],[ListaPred],L1),
        concatena(L1,[ListaAadvo],ListaConceitos),
        gera(ListaConceitos,UNL).

% S26, S78, S104
r_Acao(frase(periodo(periodo_independente([predicado(PRED),aadvo(AADVO)]))),UNL) :-
        verificaPredAcao(PRED,RestSuj,ListaPred),
        verificaAadvo(AADVO,ListaAadvo),
        gera([ListaPred,ListaAadvo],UNL).

% S112
r_Acao(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED),aadvo(AADVO)]))),UNL) :-
        verificaPredAcao(PRED,RestSuj,ListaPred),
        verificaSujAcao(SUJ,RestSuj,ListaSuj),
        verificaAadvo(AADVO,ListaAadvo),
        concatena([ListaSuj],[ListaPred],L1),
        concatena([L1],[ListaAadvo],ListaConceitos),
        gera(ListaConceitos,UNL).

% S5, S8, S10, S12, S17, S15, S20, S22, S27, S31, S36, S37, S40, S41, S47, S48, S51, S57, S60, S66, S67, S72, S73,
% S76, S77, S90, S93, S100, S101, S110, S111
r_Estado(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED)]))),UNL) :-
        verificaPredEstado(PRED,RestSuj,ListaPred),
        verificaSujEstado(SUJ,RestSuj,ListaSuj),
        gera([ListaSuj,ListaPred],UNL).

% S25, S32, S35, S75, S86, S122, S96, S118, S119, S120
r_Estado(frase(periodo(periodo_independente(predicado(PRED)))),UNL) :-
        verificaPredEstado(PRED,RestSuj,ListaPred),
        gera(ListaPred,UNL).

% S28
r_Estado(frase(periodo(periodo_independente([predicado(PRED),aadvo(AADVO),suj(SUJ)]))),UNL) :-
        verificaPredEstado(PRED,RestSuj,ListaPred),
        verificaAadvo(AADVO,ListaAadvo),
        verificaSujEstado(SUJ,RestSuj,ListaSuj),
        concatena([ListaAadvo],[ListaPred],Lista1),
        concatena([ListaSuj],Lista1,ListaConceitos),
        gera(ListaConceitos,UNL).

% S39
r_Estado(frase(periodo(periodo_independente([predicado(PRED),aadvo(AADVO)]))),UNL) :-
        verificaPredEstado(PRED,RestSuj,ListaPred),
        verificaAadvo(AADVO,ListaAadvo),
        gera([ListaPred,ListaAadvo],UNL).

% S53, S85
r_Estado(frase(periodo(periodo_independente([aadvo(AADVO),predicado(PRED)]))),UNL) :-
        verificaPredEstado(PRED,RestSuj,ListaPred),
```

```
                    verificaAadvo(AADVO,ListaAadvo),
                    gera([ListaAadvo,ListaPred],UNL).

% S55, S106, S117
r_Estado(frase(periodo(periodo_independente([aadvo(AADVO),suj(SUJ),predicado(PRED)]))),UNL) :-
                    verificaPredEstado(PRED,RestSuj,ListaPred),
                    verificaAadvo(AADVO,ListaAadvo),
                    verificaSujEstado(SUJ,RestSuj,ListaSuj),
                    concatena([ListaPred],[ListaAadvo],Lista1),
                    gera([ListaSuj,Lista1],UNL).

% S92, S98, S99
r_Estado(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED),aadvo(AADVO)]))),UNL) :-
                    verificaPredEstado(PRED,RestSuj,ListaPred),
                    verificaAadvo(AADVO,ListaAadvo),
                    verificaSujEstado(SUJ,RestSuj,ListaSuj),
                    concatena([ListaPred],[ListaAadvo],Lista1),
                    concatena([ListaSuj],Lista1,ListaConceitos),
                    gera(ListaConceitos,UNL).

% S6, S9, S30, S45
r_Processo(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED),aadvo(AADVO)]))),UNL) :-
                    verificaPredProcesso(PRED,RestSuj,ListaPred),
                    verificaSujProcesso(SUJ,RestSuj,ListaSuj),
                    verificaAadvo(AADVO,ListaAadvo),
                    gera([ListaSuj,ListaPred,ListaAadvo],UNL).

% S13, S16, S19, S29, S38, S81, S88, S94
r_Processo(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED)]))),UNL) :-
                    verificaPredProcesso(PRED,RestSuj,ListaPred),
                    verificaSujProcesso(SUJ,RestSuj,ListaSuj),
                    gera([ListaSuj,ListaPred],UNL).

% S70, S105
r_Processo(frase(periodo(periodo_independente([aadvo(AADVO1),suj(SUJ),predicado(PRED),aadvo(AADVO2)]))),UNL) :-
                    verificaPredProcesso(PRED,RestSuj,ListaPred),
                    verificaSujProcesso(SUJ,RestSuj,ListaSuj),
                    verificaAadvo(AADVO1,ListaAadvo1),
                    verificaAadvo(AADVO2,ListaAadvo2),
                    concatena([ListaSuj],[ListaPred],Lista1),
                    concatena([Lista1],[ListaAadvo1],Lista2),
                    concatena([Lista2],[ListaAadvo2],ListaConceitos),
                    gera(ListaConceitos,UNL).

% S87, S89
r_Processo(frase(periodo(periodo_independente([suj(SUJ),predicado(PRED),aadvo(AADVO)]))),UNL) :-
                    verificaPredProcesso(PRED,RestSuj,ListaPred),
                    verificaSujProcesso(SUJ,RestSuj,ListaSuj),
                    verificaAadvo(AADVO,ListaAadvo),
                    concatena([ListaPred],[ListaAadvo],Lista1),
                    concatena([ListaSuj],Lista1,ListaConceitos),
                    gera(ListaConceitos,UNL).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                         Regras de projecao para constituintes - sintaxe geral                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% nome_regra(estrutura_sintatica,estrutura_conceitual_intermediaria_parcial) :-
%                   regras_para_subconstituintes ou acesso_ao_lexico...

% ou

% nome_regra(estrutura_sintatica,restricoes_selecao,estrutura_conceitual_intermediaria_parcial) :-
%                   regras_para_subconstituintes ou acesso_ao_lexico...

% as regras para subconstituintes podem ser tanto intermediárias (apenas decompoem a estrutura sintática do constituinte
% e diparam outras regras) ou terminais (acessam o lexico enriquecido). Essas regras retornam um ou mais conceitos, em uma
% lista, representando parte da estrutura conceitual intermediária
```

```
% as restricoes podem ser representadas por mais de um argumento, por exemplo, um para restricoes do sujeito,
% e outro para restricoes do objeto direto


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                          Regras de projecao para predicados                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S1, S2, S3, S4, S11, S18, S21, S23, S26, S33, S34, S43, S44, S46, S49, S52, S54, S56, S59, S61, S62, S63, S64,
% S65, S68, S69, S71, S74, S78, S80, S83, S84, S91, S95, S97, S102, S103, S104, S107, S108, S109, S112, S113,
% S115, S116, S121
verificaPredAcao(predv(PREDV),RestSuj,Lista) :-
                    verificaPredvAcao(PREDV,RestSuj,Lista).

% S82
verificaPredAcao(predvn(PREDVN),RestSuj,Lista) :-
                    verificaPredvnAcao(PREDVN,RestSuj,Lista).

% S1, S2, S3, S4, S11, S18, S21, S23, S26, S33, S34, S43, S44, S46, S49, S52, S54, S56, S59, S61, S62, S63, S64,
% S65, S68, S69, S71, S74, S78, S80, S84, S95, S97, S102, S103, S107, S108, S109, S113, S115, S116
verificaPredvAcao([svtd(SVTD),od(OD)],RestSuj,[Evento,ListaObj]) :-
                    verificaSvtdAcao(SVTD,RestSuj,RestObj,Evento),
                    verificaOd(OD,RestObj,ListaObj).

% S7
verificaPredvAcao([svtdi(SVTDI),oi(OI),od(OD)],RestSuj,Lista) :-
                    verificaSvtdiAcao(SVTD,RestSuj,RestObjD,RestObjI,Evento),
                    verificaOd(OD,RestObjD,ListaObjD),
                    verificaOi(OI,RestObjI,ListaObjI),
                    concatena([Evento],[ListaObjI],L),
                    concatena([ListaObjD],L,Lista).

% S14, S24, S42, S58, S79, S114, S121
verificaPredvAcao([svti(SVTI),oi(OI)],RestSuj,[Evento,ListaObj]) :-
                    verificaSvtiAcao(SVTI,RestSuj,RestObj,Evento),
                    verificaOi(OI,RestObj,ListaObj).

% S82
verificaPredvnAcao([svtd(SVTD),od(OD),pobj(sadj(SADJ))],RestSuj,Lista) :-
                    verificaSvtdAcao(SVTD,RestSuj,RestObj,RestComp,Evento),
                    verificaOd(OD,RestObj,ListaObj),
                    verificaSadjMeta(SADJ,RestComp,ListaPobj),
                    concatena([ListaObj],[ListaPobj],L1),
                    concatena([Evento],L1,Lista).

% S83, S91
verificaPredvAcao([svtdi(SVTDI),od(OD),oi(OI)],RestSuj,Lista) :-
                    verificaSvtdiAcao(SVTDI,RestSuj,RestObjD,RestObjI,Evento),
                    verificaOd(OD,RestObjD,ListaObjD),
                    verificaOi(OI,RestObjI,ListaObjI),
                    concatena([Evento],[ListaObjD],L),
                    concatena([L],[ListaObjI],Lista).

% S104, S112
verificaPredvAcao(svi(SVI),RestSuj,Lista) :-
                    verificaSviAcao(SVI,RestSuj,Lista).

% S5, S15, S20, S28, S32, S35, S39, S47, S51, S66, S75, S85, S90, S92, S100, S111, S117, S118
verificaPredEstado(predv(PREDV),RestSuj,Lista) :-
                    verificaPredvEstado(PREDV,RestSuj,Lista).

% S8, S10, S12, S17, S22, S25, S27, S31, S36, S37, S40, S41, S48, S53, S55, S57, S60, S122, S96, S67, S72, S73,
% S76, S77, S86, S93, S98, S99, S101, S106, S110, S119, S120
verificaPredEstado(predn(PREDN),RestSuj,Lista) :-
                    verificaPrednEstado(PREDN,RestSuj,Lista).

% S5, S15, S20, S32, S35, S47, S51, S85, S117
verificaPredvEstado([svtd(SVTD),od(OD)],RestSuj,[Evento,ListaObj]) :-
                    verificaSvtdEstado(SVTD,RestSuj,RestObj,Evento),
                    verificaOd(OD,RestObj,ListaObj).
```

```
% S40, S48, S55, S67
verificaPrednEstado([svl(SVL),psuj(PSUJ)],RestSuj,Lista) :-
        verificaSvlEstado(SVL,T,Aux),
        verificaPsujAtributo(PSUJ,T,Aux,Lista).

% S25, S86, S122, S96, S106, S119, S120
verificaPrednEstado([svl(SVL),psuj(PSUJ)],RestSuj,[Evento,ListaPsuj]) :-
        verificaSvlEstadoE(SVL,Evento),
        verificaPsujMeta(PSUJ,ListaPsuj).

% S8, S10, S12, S17, S22, S27, S31, S37, S41, S53, S57, S60, S72, S73, S76, S77, S93, S98, S99, S101, S110
verificaPrednEstado([svl(SVL),psuj(PSUJ)],RestSuj,Lista) :-
        verificaSvlEstado(SVL,T),
        verificaPsujAtributo(PSUJ,T,Lista).

% S28, S92
verificaPredvEstado(svi(SVI),RestSuj,Lista) :-
        verificaSviEstado(SVI,RestSuj,Lista).

% S36
verificaPrednEstado([svl(SVL),psuj(PSUJ)],RestSuj,ListaPsuj) :-
        verificaSvlEstado(SVL,T,Adv),
        verificaPsujAtributo(PSUJ,T,Adv,ListaPsuj).

% S39, S66, S75, S90, S100, S111, S118
verificaPredvEstado([svti(SVTI),oi(OI)],RestSuj,[Evento,ListaObj]) :-
        verificaSvtiEstado(SVTI,RestSuj,RestObj,Evento),
        verificaOi(OI,RestObj,ListaObj).

% S13
verificaPredProcesso(predvn(PREDVN),RestSuj,Lista) :-
        verificaPredvnProcesso(PREDVN,RestSuj,Lista).

% S6, S9, S16, S19, S29, S30, S38, S45, S70, S81, S87, S88, S89, S94, S105
verificaPredProcesso(predv(PREDV),RestSuj,Lista) :-
        verificaPredvProcesso(PREDV,RestSuj,Lista).

% S6, S9, S16, S19, S29, S30, S45, S70, S81, S87, S88, S89, S94, S105
verificaPredvProcesso(svi(SVI),RestSuj,Lista) :-
        verificaSviProcesso(SVI,RestSuj,Lista).

% S38
verificaPredvProcesso([svti(SVTI),oi(OI)],RestSuj,[Evento,ListaObj]) :-
        verificaSvtiProcesso(SVTI,RestSuj,RestObj,Evento),
        verificaOi(OI,RestObj,ListaObj).

% S13
verificaPredvnProcesso([oda(lexico(se)),svtd(SVTD),psuj(PSUJ)],RestSuj,[Evento,ListaPsuj]) :-
        verificaSvtdProcesso(SVTD,RestSuj,Evento),
        verificaPsujMeta(PSUJ,ListaPsuj).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                    Regras de projecao para verbos                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S1, S2, S3, S4, S11, S18, S21, S23, S26, S33, S49, S52, S43, S54, S59, S61, S65, S68, S69, S71, S74, S78,
% S97, S102, S103, S107, S109, S113
verificaSvtdAcao(verbo(V),RestSuj,RestObj,[evento:Evento]) :-
        v(sin(T),can(CanV),[V|_],[]),
        v(sin(_,td),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObj)]),[CanV|_],[]),
        (AP = acao; AP = acao_proc),
        string_atom(StrUnlV,UnlV),
        string_atom(StrT,T),
        strcat(StrUnlV,$.@entry.@$,StrE0),
        strcat(StrE0,StrT,StrE),
        string_atom(StrE,Evento).
```

```
% S7
verificaSvtdiAcao(verbo(V),RestSuj,RestObjD,RestObjI,[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,bi),unl(UnlV),cl(AP),rest([suj(RestSuj),comp(RestObjI),obj(RestObjD)]),[CanV|_],[]),
                        (AP = acao; AP = acao_proc),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento).

% S14, S24, S42, S58, S79, S114
verificaSvtiAcao(verbo(V),RestSuj,RestObj,[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,ti),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObj)]),[CanV|_],[]),
                        (AP = acao; AP = acao_proc),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento).
% S34, S80
verificaSvtdAcao([aadvl((aadvl_simples(sadv(adv(ADV)))))),verbo(V)],RestSuj,RestObj,[evento:Evento]) :-
                        adv(sin(neg),unl(UnlAdv),[ADV|_],[]),
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,td),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObj)]),[CanV|_],[]),
                        (AP = acao; AP = acao_proc),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        string_atom(StrUnlAdv,UnlAdv),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE1),
                        strcat(StrE1,$.@$,StrE2),
                        strcat(StrE2,StrUnlAdv,StrE),
                        string_atom(StrE,Evento).

% S44
verificaSvtdAcao([aadvl((aadvl_simples(sadv(adv(ADV)))))),verbo(V)],RestSuj,RestObj,Lista) :-
                        adv(sin(cir_mod),unl(UnlAdv),[ADV|_],[]),
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,td),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObj)]),[CanV|_],[]),
                        (AP = acao; AP = acao_proc),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento),
                        concatena([maneir:UnlAdv],[evento:Evento],Lista).

% S46, S56, S62, S63, S64, S84, S95, S108, S115, S116
verificaSvtdAcao([verboaux(VAUX),verbo(V)],RestSuj,RestObj,[evento:Evento]) :-
                        v(sin(T),can(CanVaux),[VAUX|_],[]),
                        v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                        v(sin(_,td),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObj)]),[V|_],[]),
                        (AP = acao; AP = acao_proc),
                        string_atom(StrUnlVaux,UnlVaux),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE1),
                        strcat(StrE1,$.@$,StrE2),
                        strcat(StrE2,StrUnlVaux,StrE),
                        string_atom(StrE,Evento).

% S82
verificaSvtdAcao(verbo(V),RestSuj,RestObj,RestComp,[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,td),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObj),comp(RestComp)]),[CanV|_],[]),
                        (AP = acao; AP = acao_proc),
                        string_atom(StrUnlV,UnlV),
```

```
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).
% S83
verificaSvtdiAcao([verboaux(VAUX),verbo(V)],RestSuj,RestObjD,RestObjI,[evento:Evento]) :-
                    v(sin(T),can(CanVaux),[VAUX|_],[]),
                    v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                    v(sin(_,bi),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObjD),comp(RestObjI)]),[V|_],[]),
                    (AP = acao; AP = acao_proc),
                    string_atom(StrUnlVaux,UnlVaux),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE1),
                    strcat(StrE1,$.@$,StrE2),
                    strcat(StrE2,StrUnlVaux,StrE),
                    string_atom(StrE,Evento).

% S91
verificaSvtdiAcao(verbo(V),RestSuj,RestObjD,RestObjI,[evento:Evento]) :-
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,bi),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObjD),comp(RestObjI)]),[CanV|_],[]),
                    (AP = acao; AP = acao_proc),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).

% S104, S112
verificaSviAcao(verbo(V),RestSuj,[evento:Evento]) :-
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,int),unl(UnlV),cl(AP),rest([suj(RestSuj)]),[CanV|_],[]),
                    (AP = acao; AP = acao_proc),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).

% 121
verificaSvtiAcao([verboaux(VAUX),verbo(V)],RestSuj,RestObj,[evento:Evento]) :-
                    v(sin(T),can(CanVaux),[VAUX|_],[]),
                    v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                    v(sin(_,ti),unl(UnlV),cl(AP),rest([suj(RestSuj),obj(RestObjD)]),[V|_],[]),
                    (AP = acao; AP = acao_proc),
                    string_atom(StrUnlVaux,UnlVaux),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE1),
                    strcat(StrE1,$.@$,StrE2),
                    strcat(StrE2,StrUnlVaux,StrE),
                    string_atom(StrE,Evento).

% S5, S15, S20, S32, S47, S51, S117
verificaSvtdEstado(verbo(V),RestSuj,RestObj,[evento:Evento]) :-
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,td),unl(UnlV),cl(estado),rest([suj(RestSuj),obj(RestObj)]),[CanV|_],[]),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).

% S8, S10, S12, S17, S22, S27, S31, S37, S41, S53, S57, S60, S72, S73, S76, S77, S93, S98, S99, S101, S110
verificaSvlEstado(verbo(V),T) :-
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,lig),unl(_),cl(estado),rest([suj(RestSuj),pred(RestPsuj)]),[CanV|_],[]).
```

```
% S25, S86, S122, S96, S119, S120
verificaSvlEstadoE(verbo(V),[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        (T = imper_afirm; T = pres_subj),
                        v(sin(_,lig),unl(UnlV),cl(estado),rest([suj(RestSuj),pred(RestPsuj)]),[CanV|_],[]),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento).

% S28
verificaSviEstado(verbo(V),RestSuj,[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,int),unl(UnlV),cl(estado),rest([suj(RestSuj)]),[CanV|_],[]),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento).

% S35, S85
verificaSvtdEstado([aadvl((aadvl_simples(sadv(adv(ADV)))))),verbo(V)],RestSuj,RestObj,[evento:Evento]) :-
                        adv(sin(neg),unl(UnlAdv),[ADV|_],[]),
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,td),unl(UnlV),cl(estado),rest([obj(RestObj)]),[CanV|_],[]),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        string_atom(StrUnlAdv,UnlAdv),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE1),
                        strcat(StrE1,$.@$,StrE2),
                        strcat(StrE2,StrUnlAdv,StrE),
                        string_atom(StrE,Evento).

% S36
verificaSvlEstado([aadvl((aadvl_simples(sadv(adv(ADV)))))),verbo(V)],T,UnlAdv) :-
                        adv(sin(neg),unl(UnlAdv),[ADV|_],[]),
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,lig),unl(_),cl(estado),rest([suj(RestSuj),pred(RestPsuj)]),[CanV|_],[]).

% S39
verificaSvtiEstado(verbo(V),RestSuj,RestObj,[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,ti),unl(UnlV),cl(estado),rest([obj(RestObj)]),[CanV|_],[]),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento).

% S40, S48, S55, S67
verificaSvlEstado([verboaux(VAUX),verbo(V)],T,UnlVaux) :-
                        v(sin(T),can(CanVaux),[VAUX|_],[]),
                        v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                        v(sin(_,lig),unl(_),cl(estado),rest([suj(RestSuj),pred(RestPsuj)]),[V|_],[]).

% S66, S111
verificaSvtiEstado(verbo(V),RestSuj,RestObj,[evento:Evento]) :-
                        v(sin(T),can(CanV),[V|_],[]),
                        v(sin(_,ti),unl(UnlV),cl(estado),rest([suj(RestSuj),pred(RestObj)]),[CanV|_],[]),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE),
                        string_atom(StrE,Evento).

% S75, S90, S100, S118
verificaSvtiEstado(verbo(V),RestSuj,RestObj,[evento:Evento]) :-
```

```
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,ti),unl(UnlV),cl(estado),rest([suj(RestSuj),obj(RestObj)]),[CanV|_],[]),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).

% S92
verificaSviEstado([aadvl(aadvl_simples(sadv(adv(ADV)))),verbo(V)],RestSuj,[evento:Evento]) :-
                    adv(sin(neg),unl(UnlAdv),[ADV|_],[]),
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,int),unl(UnlV),cl(estado),rest([suj(RestSuj)]),[CanV|_],[]),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    string_atom(StrUnlAdv,UnlAdv),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE1),
                    strcat(StrE1,$.@$,StrE2),
                    strcat(StrE2,StrUnlAdv,StrE),
                    string_atom(StrE,Evento).

% S6, S9, S19, S29, S30, S45, S70, S81, S94
verificaSviProcesso(verbo(V),RestSuj,[evento:Evento]) :-
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,int),unl(UnlV),cl(processo),rest([suj(RestSuj)]),[CanV|_],[]),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).

% S13
verificaSvtdProcesso(verbo(V),RestSuj,[evento:Evento]) :-
                    v(sin(T),can(CanV),[V|_],[]),
                    v(sin(_,td),unl(UnlV),cl(processo),rest([suj(RestSuj),pred(RestObj)]),[CanV|_],[]),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE),
                    string_atom(StrE,Evento).

% S16, S29, S88, S89, S105
verificaSviProcesso([verboaux(VAUX),verbo(V)],RestSuj,[evento:Evento]) :-
                    v(sin(T),can(CanVaux),[VAUX|_],[]),
                    v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                    v(sin(inf_pess,int),unl(UnlV),cl(processo),rest([suj(RestSuj)]),[V|_],[]),
                    string_atom(StrUnlVaux,UnlVaux),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE1),
                    strcat(StrE1,$.@$,StrE2),
                    strcat(StrE2,StrUnlVaux,StrE),
                    string_atom(StrE,Evento).

% S87
verificaSviProcesso([verboaux(VAUX),verbo(V)],RestSuj,[evento:Evento]) :-
                    v(sin(T),can(CanVaux),[VAUX|_],[]),
                    v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                    v(sin(gerun),can(CanV),[V|_],[]),
                    v(sin(inf_pess,int),unl(UnlV),cl(processo),rest([suj(RestSuj)]),[CanV|_],[]),
                    string_atom(StrUnlV,UnlV),
                    string_atom(StrT,T),
                    strcat(StrUnlV,$.@entry.@$,StrE0),
                    strcat(StrE0,StrT,StrE1),
                    strcat(StrE1,$.@$,StrE2),
                    strcat(StrE2,$progress$,StrE),
                    string_atom(StrE,Evento).
```

```prolog
% S38
verificaSvtiProcesso([verboaux(VAUX),verbo(V)],RestSuj,RestObj,[evento:Evento]) :-
                        v(sin(T),can(CanVaux),[VAUX|_],[]),
                        v(sin(_,aux),unl(UnlVaux),cl(modal),_,[CanVaux|_],[]),
                        v(sin(_,ti),unl(UnlV),cl(processo),rest([suj(RestSuj),obj(RestObj)]),[V|_],[]),
                        string_atom(StrUnlVaux,UnlVaux),
                        string_atom(StrUnlV,UnlV),
                        string_atom(StrT,T),
                        strcat(StrUnlV,$.@entry.@$,StrE0),
                        strcat(StrE0,StrT,StrE1),
                        strcat(StrE1,$.@$,StrE2),
                        strcat(StrE2,StrUnlVaux,StrE),
                        string_atom(StrE,Evento).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Regras de projecao para sujeitos                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S11, S14, S18, S23, S33, S44, S46, S49, S52, S54, S58, S63, S68, S69, S71, S83, S95, S97, S102, S108, S115, S121
verificaSujAcao(suj_simples(sn(SN)),Rest,Lista) :-
                        verificaSnAgente(SN,Rest,Lista).

% S56, S65
verificaSujAcao(suj_composto([suj_simples(sn(SN1)),coordenador(C),suj_simples(sn(SN2))]),Rest,[Agente1,Agente2]) :-
                        verificaSnAgente1(SN1,Rest,Agente1),
                        conj(sin(coord,adit),_,[C|_],[]),
                        verificaSnAgente2(SN2,Rest,Agente2).

% S74
verificaSujAcao(osss(oss(ori(periodo(periodo_independente(PERIODO))))),Rest,Lista) :-
                        verificaPeriodoAgente(PERIODO,Lista).

% S112
verificaSujAcao(suj_simples([sn(SN),virgula(,),aposto(sn(APOSTO)),virgula(,)]),Rest,[Agente,Conteudo]) :-
                        verificaSnAgente(SN,Rest,Agente),
                        verificaSnAposto(APOSTO,Conteudo).

% S116
verificaSujAcao(suj_simples(lexico(SN)),Rest,Lista) :-
                        verificaSnAgente(SN,Rest,Lista).

% S74
verificaPeriodoAgente([predicado(predv(svi(verbo(V)))),aadvo(AADVO)],[agenteE:UnlV,ListaAadvo]) :-
                        v(sin(inf_pess,int),unl(UnlV),cl(_),rest([suj(RestSuj)]),[V|_],[]),
                        verificaAadvo(AADVO,ListaAadvo).

% S5
verificaSujEstado(suj_simples(sn(SN)),Rest,Lista) :-
                        verificaSnExperimentador(SN,Rest,Lista).

% S8, S10, S12, S15, S17, S27, S28, S31, S36, S40, S41, S47, S48, S51, S55, S57, S60, S66, S67, S76, S77,
% S90, S92, S93, S98, S99, S100, S111, S117
verificaSujEstado(suj_simples(sn(SN)),Rest,Lista) :-
                        verificaSnInativo(SN,Rest,Lista).

% S20, S22, S37, S110
verificaSujEstado(suj_composto([suj_simples(sn(SN1)),coordenador(C),suj_simples(sn(SN2))]),Rest,[Inativo1,Inativo2]) :-
                        verificaSnInativo1(SN1,Rest,Inativo1),
                        conj(sin(coord,adit),_,[C|_],[]),
                        verificaSnInativo2(SN2,Rest,Inativo2).

% S72, S101
verificaSujEstado(osss(oss(ori(periodo(periodo_independente(PERIODO))))),Rest,Lista) :-
                        verificaPeriodoInativo(PERIODO,Lista).

% S72, S101
verificaPeriodoInativo(predicado(predv([svtd(verbo(V)),od(OD)])),[inativoE:UnlV,ListaObj]) :-
                        v(sin(inf_pess,td),unl(UnlV),cl(_),rest([suj(RestSuj),obj(RestObj)]),[V|_],[]),
                        verificaOd(OD,RestObj,ListaObj).
```

```
% S6, S9, S13, S16, S19, S29, S30, S38, S45, S70, S81, S88, S94, S105, S106
verificaSujProcesso(suj_simples(sn(SN)),Rest,Lista) :-
                    verificaSnPaciente(SN,Rest,Lista).

% S87
verificaSujProcesso(suj_simples(SUJ),Rest,Lista) :-
                    verificaSnPaciente(SUJ,Rest,Lista).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                    Regras de projecao para sintagmas nominais sujeitos                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S5                 --> experimentador
verificaSnExperimentador(pron_subst(PS),Rest,[experimentador:UnlPro]) :-
                    pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                    subconjunto(Tracos,Rest).

% S6, S9             --> paciente
verificaSnPaciente([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,[paciente:P,Modo]) :-
                    art(sin(D),_,[A|_],[]),
                    verificaAadndModo(AADND,Modo),
                    ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                              subconjunto(Tracos,Rest),
                              string_atom(StrUnlS,UnlS),
                              string_atom(StrAr,D),
                              strcat(StrUnlS,$.@$,StrP0),
                              strcat(StrP0,StrAr,StrP1),
                              string_atom(StrP1,P));

                    (s(sin(N),can(CanS),[S|_],[]),
                              s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                              subconjunto(Tracos,Rest),
                              string_atom(StrUnlS,UnlS),
                              string_atom(StrAr,D),
                              string_atom(StrN,N),
                              strcat(StrUnlS,$.@$,StrP0),
                              strcat(StrP0,StrAr,StrP1),
                              strcat(StrP1,$.@$,StrP2),
                              strcat(StrP2,StrN,StrP3),
                              string_atom(StrP3,P))).

% S13, S38, S108
verificaSnPaciente(pron_subst(PS),Rest,[paciente:UnlPro]) :-
                    pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                    subconjunto(Tracos,Rest).

% S16
verificaSnPaciente([subst(S),aadnd(aadnd_simples(AADND))],Rest,[paciente:P,Modo]) :-
                    verificaAadndModo(AADND,Modo),
                    ((s(_,unl(P),sem(Tracos),[S|_],[]),
                              subconjunto(Tracos,Rest));

                    (s(sin(N),can(CanS),[S|_],[]),
                              s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                              subconjunto(Tracos,Rest),
                              string_atom(StrUnlS,UnlS),
                              string_atom(StrN,N),
                              strcat(StrUnlS,$.@$,StrP0),
                              strcat(StrP0,StrN,StrP1),
                              string_atom(StrP1,P))).

% S19
verificaSnPaciente([aadne(AADNE),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                    verificaAadnePosse(AADNE,Posse),
                    verificaAadndModo(AADND,Modo),
                    ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                              subconjunto(Tracos,Rest),
                              concatena([paciente:UnlS],[Modo],L2));
```

```
                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrN,StrP1),
                        string_atom(StrP1,P),
                        concatena([paciente:P],[Modo],L2))),
                concatena(Posse,L2,Lista).

% S29
verificaSnPaciente([aadne(AADNE),subst(S)],Rest,[Modo,paciente:P]) :-
                verificaAadneModo(AADNE,Modo),
                ((s(_,unl(P),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrN,StrP1),
                        string_atom(StrP1,P))).

% S30
verificaSnPaciente(pron_subst(PS),Rest,[paciente:UnlPro]) :-
                pron(sin(dem),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).
% S45
verificaSnPaciente([aadne([sdet([canonica(P),artigo(ART)]),ADJ]),subst(S)],Rest,Lista) :-
                pron(sin(inde),unl(UnlP),[P|_],[]),
                verificaAadnePosse(ADJ,Posse),
                concatena(Posse,[[modo:UnlP]],L2),
                art(sin(D),_,[ART|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrAr,StrP1),
                        string_atom(StrP1,Pa),
                        concatena([paciente:Pa],L2,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrAr,StrP1),
                        strcat(StrP1,$.@$,StrP2),
                        strcat(StrP2,StrN,StrP3),
                        string_atom(StrP3,Pa),
                        concatena([paciente:Pa],L2,Lista))).

% S70, S81, S94
verificaSnPaciente([aadne(sdet(artigo(A))),subst(S)],Rest,[paciente:P]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrAr,StrP1),
                        string_atom(StrP1,P));
                        (s(sin(N),can(CanS),[S|_],[]),
```

```
                    s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                            subconjunto(Tracos,Rest),
                            string_atom(StrUnlS,UnlS),
                            string_atom(StrAr,D),
                            string_atom(StrN,N),
                            strcat(StrUnlS,$.@$,StrP0),
                            strcat(StrP0,StrAr,StrP1),
                            strcat(StrP1,$.@$,StrP2),
                            strcat(StrP2,StrN,StrP3),
                            string_atom(StrP3,P))).

% S87
verificaSnPaciente(lexico(PS),Rest,[paciente:UnlPro]) :-
                    pron(sin(ret),unl(UnlPro),[PS|_],[]).

% S88, S89
verificaSnPaciente(pron_subst(PS),Rest,[paciente:UnlPro]) :-
                    pron(sin(inde),unl(UnlPro),[PS|_],[]).

% S105
verificaSnPaciente([aadne([sdet(artigo(ART)),ADJ]),subst(S)],Rest,[paciente:Pa,Posse]) :-
                    verificaAadnePosse(ADJ,Posse),
                    art(sin(D),_,[ART|_],[]),
                    ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                            subconjunto(Tracos,Rest),
                            string_atom(StrUnlS,UnlS),
                            string_atom(StrAr,D),
                            strcat(StrUnlS,$.@$,StrP0),
                            strcat(StrP0,StrAr,StrP1),
                            string_atom(StrP1,Pa));

                    (s(sin(N),can(CanS),[S|_],[]),
                            s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                            subconjunto(Tracos,Rest),
                            string_atom(StrUnlS,UnlS),
                            string_atom(StrAr,D),
                            string_atom(StrN,N),
                            strcat(StrUnlS,$.@$,StrP0),
                            strcat(StrP0,StrAr,StrP1),
                            strcat(StrP1,$.@$,StrP2),
                            strcat(StrP2,StrN,StrP3),
                            string_atom(StrP3,Pa))).


% S8                        --> inativo
verificaSnInativo([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,[inativo:I,Modo]) :-
                    art(sin(D),_,[A|_],[]),
                    verificaCnModo(CN,Modo),
                    ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                            subconjunto(Tracos,Rest),
                            string_atom(StrUnlS,UnlS),
                            string_atom(StrAr,D),
                            strcat(StrUnlS,$.@$,StrI0),
                            strcat(StrI0,StrAr,StrI1),
                            string_atom(StrI1,I));

                    (s(sin(N),can(CanS),[S|_],[]),
                            s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                            subconjunto(Tracos,Rest),
                            string_atom(StrUnlS,UnlS),
                            string_atom(StrAr,D),
                            string_atom(StrN,N),
                            strcat(StrUnlS,$.@$,StrI0),
                            strcat(StrI0,StrAr,StrI1),
                            strcat(StrI1,$.@$,StrI2),
                            strcat(StrI2,StrN,StrI3),
                            string_atom(StrI3,I))).
```

```
% S10, S17, S41, S57
verificaSnInativo(subst(S),Rest,[inativo:I]) :-
                ((s(_,unl(I),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrN,StrI1),
                        string_atom(StrI,I))).

% S12, S55, S76, S77
verificaSnInativo(pron_subst(PS),Rest,[inativo:UnlPro]) :-
                pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).

% S15, S67, S73
verificaSnInativo(pron_subst(PS),Rest,[inativo:UnlPro]) :-
                pron(sin(dem),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).


% S20, S90
verificaSnInativo1([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,[inativo1:I,Modo]) :-
                art(sin(D),_,[A|_],[]),
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@entry.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        string_atom(StrI1,I));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        strcat(StrI1,$.@$,StrI2),
                        strcat(StrI2,StrN,StrI3),
                        string_atom(StrI3,I))).

verificaSnInativo2([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,[inativo2:I,Modo]) :-
                art(sin(D),_,[A|_],[]),
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        string_atom(StrI1,I));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        strcat(StrI1,$.@$,StrI2),
                        strcat(StrI2,StrN,StrI3),
```

```
                        string_atom(StrI3,I))).

% S22, S37
verificaSnInativo1(subst(S),Rest,[inativo1:I]) :-
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        strcat(StrUnlS,$.@entry.@$,StrI0),
                        string_atom(StrI0,I));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,StrI0),
                        strcat(StrI0,StrN,StrI),
                        string_atom(StrI,I))).

verificaSnInativo2(subst(S),Rest,[inativo2:I]) :-
                ((s(_,unl(I),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrN,StrI),
                        string_atom(StrI,I))).

% S27, S31
verificaSnInativo([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,Lista) :-
                art(sin(D),_,[A|_],[]),
                verificaCnParceiro(CN,Parceiro),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        string_atom(StrI1,I),
                        concatena([inativo:I],Parceiro,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        strcat(StrI1,$.@$,StrI2),
                        strcat(StrI2,StrN,StrI3),
                        string_atom(StrI3,I),
                        concatena([inativo:I],Parceiro,Lista))).

% S28
verificaSnInativo([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,Lista) :-
                art(sin(D),_,[A|_],[]),
                verificaCnRazao(CN,Razao),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        string_atom(StrI1,I),
                        concatena([inativo:I],Razao,Lista));
```

```
                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        strcat(StrI1,$.@$,StrI2),
                        strcat(StrI2,StrN,StrI3),
                        string_atom(StrI3,I),
                        concatena([inativo:I],Razao,Lista))).

% S36
verificaSnInativo([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,Lista) :-
                art(sin(D),_,[A|_],[]),
                verificaCnModo(CN,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        string_atom(StrI1,I),
                        concatena([inativo:I],Modo,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        strcat(StrI1,$.@$,StrI2),
                        strcat(StrI2,StrN,StrI3),
                        string_atom(StrI3,I),
                        concatena([inativo:I],Modo,Lista))).

% S40
verificaSnInativo(pron_subst(PS),Rest,[inativo:UnlPro]) :-
                pron(sin(inde),unl(UnlPro),[PS|_],[]).

% S47, S76, S117
verificaSnInativo(pron_subst(PS),Rest,[inativo:UnlPro]) :-
                pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).

% S48, S92, S111
verificaSnInativo([aadne(sdet(artigo(A))),subst(S)],Rest,[inativo:I]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        string_atom(StrI1,I));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrI0),
                        strcat(StrI0,StrAr,StrI1),
                        strcat(StrI1,$.@$,StrI2),
                        strcat(StrI2,StrN,StrI3),
```

```
                        string_atom(StrI3,I))).

% S51, S99, S100
verificaSnInativo([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                        art(sin(D),_,[A|_],[]),
                        verificaAadndModo(AADND,Modo),
                        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                                subconjunto(Tracos,Rest),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrAr,D),
                                strcat(StrUnlS,$.@$,StrI0),
                                strcat(StrI0,StrAr,StrI1),
                                string_atom(StrI1,I),
                                concatena([inativo:I],Modo,Lista));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                subconjunto(Tracos,Rest),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrAr,D),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrI0),
                                strcat(StrI0,StrAr,StrI1),
                                strcat(StrI1,$.@$,StrI2),
                                strcat(StrI2,StrN,StrI3),
                                string_atom(StrI3,I),
                                concatena([inativo:I],Modo,Lista))).

% S60
verificaSnInativo([aadne(AADNE),subst(S)],Rest,Lista) :-
                        verificaAadnePosse(AADNE,Posse),
                        ((s(_,unl(I),sem(Tracos),[S|_],[]),
                                subconjunto(Tracos,Rest));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                subconjunto(Tracos,Rest),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrI0),
                                strcat(StrI0,StrN,StrI1),
                                string_atom(StrI1,I))),
                                concatena(Posse,[inativo:I],Lista).

% S66
verificaSnInativo([subst(S),aadnd(aadnd_simples(AADND))],Rest,[inativo:I,Modo]) :-
                        verificaAadndModo(AADND,Modo),
                        ((s(_,unl(I),sem(Tracos),[S|_],[]),
                                subconjunto(Tracos,Rest));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                subconjunto(Tracos,Rest),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@entry.@$,StrI0),
                                strcat(StrI0,StrN,StrI1),
                                string_atom(StrI1,I))).

% S93
verificaSnInativo([aadne(AADNE),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                        verificaAadnePosse(AADNE,Posse),
                        verificaAadndModo(AADND,Modo),
                        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                                subconjunto(Tracos,Rest),
                                concatena([inativo:UnlS],[Modo],L2));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                subconjunto(Tracos,Rest),
```

```
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,Strl0),
                        strcat(Strl0,StrN,Strl1),
                        string_atom(Strl1,I),
                        concatena([inativo:I],[Modo],L2))),
                concatena(Posse,L2,Lista).

% S110
verificaSnInativo1([aadne(sdet(artigo(A))),subst(S)],Rest,[inativo1:I]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@entry.@$,Strl0),
                        strcat(Strl0,StrAr,Strl1),
                        string_atom(Strl1,I));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,Strl0),
                        strcat(Strl0,StrAr,Strl1),
                        strcat(Strl1,$.@$,Strl2),
                        strcat(Strl2,StrN,Strl3),
                        string_atom(Strl3,I))).

verificaSnInativo2([aadne(sdet(artigo(A))),subst(S)],Rest,[inativo2:I]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,Strl0),
                        strcat(Strl0,StrAr,Strl1),
                        string_atom(Strl1,I));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,Strl0),
                        strcat(Strl0,StrAr,Strl1),
                        strcat(Strl1,$.@$,Strl2),
                        strcat(Strl2,StrN,Strl3),
                        string_atom(Strl3,I))).


% S11, S14, S52, S68, S83, S102, S108                   --> agente
verificaSnAgente([aadne(sdet(artigo(A))),subst(S)],Rest,[agente:Ag]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrAr,StrAg1),
                        string_atom(StrAg1,Ag));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
```

```
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrAr,StrAg1),
                        strcat(StrAg1,$.@$,StrAg2),
                        strcat(StrAg2,StrN,StrAg3),
                        string_atom(StrAg3,Ag))).

% S18, S23
verificaSnAgente([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,Lista) :-
                verificaCnLugar(CN,Lugar),
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrAr,StrAg1),
                        string_atom(StrAg1,Ag),
                        concatena([agente:Ag],Lugar,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrAr,StrAg1),
                        strcat(StrAg1,$.@$,StrAg2),
                        strcat(StrAg2,StrN,StrAg3),
                        string_atom(StrAg3,Ag),
                        concatena([agente:Ag],Lugar,Lista))).

% Variacao sem artigo de S18, S23
verificaSnAgente([subst(S),cn(CN)],Rest,Lista) :-
                verificaCnLugar(CN,Lugar),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([agente:UnlS],Lugar,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrN,StrAg1),
                        string_atom(StrAg1,Ag),
                        concatena([agente:Ag],Lugar,Lista))).

% S33
verificaSnAgente([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                verificaAadndModo(AADND,Modo),
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrAr,StrAg1),
                        string_atom(StrAg1,Ag),
                        concatena([agente:Ag],Modo,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAg0),
```

```
                        strcat(StrAg0,StrAr,StrAg1),
                        strcat(StrAg1,$.@$,StrAg2),
                        strcat(StrAg2,StrN,StrAg3),
                        string_atom(StrAg3,Ag),
                        concatena([agente:Ag],Modo,Lista))).

% S44, S84, S95
verificaSnAgente(pron_subst(PS),Rest,[agente:UnlPro]) :-
                pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).

% S46, S54
verificaSnAgente(pron_subst(PS),Rest,[agente:UnlPro]) :-
                pron(sin(dem),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).

% S56, S65
verificaSnAgente1(subst(S),Rest,[agente:Ag1]) :-
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        strcat(StrUnlS,$.@entry$,StrAg0),
                        string_atom(StrAg0,Ag));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,StrAg0),
                        strcat(StrAg0,StrN,StrAg1),
                        string_atom(StrAg1,Ag))).

% S56
verificaSnAgente2([subst(S),aadnd(aadnd_simples(AADND))],Rest,[agente:Ag2,Modo]) :-
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(Ag),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrN,StrAg1),
                        string_atom(StrAg1,Ag))).

% S65
verificaSnAgente2(subst(S),Rest,[agente:Ag2]) :-
                ((s(_,unl(Ag),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAg0),
                        strcat(StrAg0,StrN,StrAg1),
                        string_atom(StrAg1,Ag))).

% S58, S115
verificaSnAgente([subst(S),aadnd(aadnd_simples(AADND))],Rest,[agente:Ag,Modo]) :-
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(Ag),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
```

```
                                        string_atom(StrUnlS,UnlS),
                                        string_atom(StrN,N),
                                        strcat(StrUnlS,$.@$,StrAg0),
                                        strcat(StrAg0,StrN,StrAg1),
                                        string_atom(StrAg1,Ag))).

% S62, S82
verificaSnAgente(pron_subst(PS),Rest,[agente:UnlPro]) :-
                    pron(sin(inde),unl(UnlPro),[PS|_],[]).

% S69, S97, S121
verificaSnAgente([aadne(AADNE),subst(S)],Rest,[Posse,agente:Ag]) :-
                    verificaAadnePosse(AADNE,Posse),
                    ((s(_,unl(Ag),sem(Tracos),[S|_],[]),
                                        subconjunto(Tracos,Rest));

                    (s(sin(N),can(CanS),[S|_],[]),
                                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                        subconjunto(Tracos,Rest),
                                        string_atom(StrUnlS,UnlS),
                                        string_atom(StrN,N),
                                        strcat(StrUnlS,$.@$,StrAg0),
                                        strcat(StrAg0,StrN,StrAg1),
                                        string_atom(StrAg1,Ag))).

% S112
verificaSnAgente(nome_proprio(S),Rest,[agente:Ag]) :-
                    ((s(_,unl(Ag),sem(Tracos),[S|_],[]),
                                        subconjunto(Tracos,Rest));

                    (s(sin(N),can(CanS),[S|_],[]),
                                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                        subconjunto(Tracos,Rest),
                                        string_atom(StrUnlS,UnlS),
                                        string_atom(StrN,N),
                                        strcat(StrUnlS,$.@$,StrAg0),
                                        strcat(StrAg0,StrN,StrAg1),
                                        string_atom(StrAg1,Ag))).

% S116
verificaSnAgente(PS,Rest,[agente:UnlPro]) :-
                                            pron(sin(ret),unl(UnlPro),[PS|_],[]).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      Regras de projecao para adjuntos adnominais à esquerda                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S1, S2, S3, S4, S5, S12, S18, S19, S20, S23, S33, S34, S59, S105
verificaAadnePosse(adj(A),[posse:UnlP]) :-
                    pron(sin(poss),can(CanP),[A|_],[]),
                    pron(_,unl(UnlP),_,[CanP|_],[]).

% S6, S7, S11, S24, S29, S30, S32
verificaAadneModo(adj(A),[modo:UnlA]) :-
                    ((adj(can(CanA),[A|_],[]), adj(unl(UnlA),[CanA|_],[]));
                    adj(unl(UnlA),[A|_],[])).

% S58
verificaAadneModo([sdet(artigo(A)),adj(ADJ)],D,[modo:UnlAdj]) :-
                    art(sin(D),_,[A|_],[]),
                    ((adj(can(CanAdj),[ADJ|_],[]), adj(unl(UnlAdj),[CanAdj|_],[]));
                    adj(unl(UnlAdj),[ADJ|_],[])).

% S87, S116, S118, S91
verificaAadnePosse([sdet(artigo(ART)),adj(ADJ)],[posse:UnlP]) :-
                    pron(sin(poss),can(CanP),[ADJ|_],[]),
                    pron(_,unl(UnlP),_,[CanP|_],[]).

% S121
```

```
verificaAadneQuantidade(adj(A),[quantidade:UnlA]) :-
                    pron(sin(inde),unl(UnlA),[A|_],[]).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                 Regras de projecao para adjuntos adnominais à direita                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S1, S2, S3, S6, S7, S8, S9, S10, S12, S16, S17, S19, S20, S26, S36, S55, S59, S97, S100, S101, S103
verificaAadndModo(sadj(adj(A)),[modo:UnlA]) :-
                    ((adj(can(CanA),[A|_],[]), adj(unl(UnlA),[CanA|_],[]));
                    adj(unl(UnlA),[A|_],[])).

% S33
verificaAadndModo(sadj([adj(A),cn(CN)]),Lista) :-
                    verificaCnLugar(CN,Lugar),
                    ((adj(can(CanA),[A|_],[]),
                    adj(unl(UnlA),[CanA|_],[]),
                    concatena([modo:UnlA],Lugar,Lista));

                    (adj(unl(UnlA),[A|_],[]),
                    concatena([modo:UnlA],Lugar,Lista))).

% S51
verificaAadndModo(sp([preposicao(P),sadv(SADV)]),Lista) :-
                    verificaPrepDe(P,_),
                    verificaSadvTempo(SADV,Lista).

% S52, S56, S66, S80, S115
verificaAadndModo(sp([preposicao(P),sn(SN)]),Lista) :-
                    verificaPrepDe(P,_),
                    verificaSnModo(SN,Lista).

% S76
verificaAadndModo(sadj([aadvl(aadvl_simples(sadv(adv(ADV)))),adj(ADJ)]),[maneir:UnlAdv,modo:UnlAdj]) :-
                    adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                    ((adj(can(CanA),[ADJ|_],[]), adj(unl(UnlAdj),[CanA|_],[]));
                    adj(unl(UnlAdj),[ADJ|_],[])).

% S15
verificaAadndBeneficiario(sp([preposicao(P),sn(SN)]),Lista) :-
                    verificaPrepPara(P,_),
                    verificaSnBeneficiario(SN,_,Lista).

% S20
verificaAadndOrigem(sp([preposicao(P),sn(SN)]),Lista) :-
                    verificaPrepDe(P,_),
                    verificaSnOrigem(SN,Lista).

% S53
verificaAadndProposito(sp([preposicao(P),sn(SN)]),Lista) :-
                    verificaPrepPara(P,_),
                    verificaSnProposito(SN,Lista).

% S109
verificaAadndModo(sadj([adj(A),cn(CN)]),Lista) :-
                    verificaCnCoCoisa(CN,CoCoisa),
                    ((adj(can(CanA),[A|_],[]),
                    adj(unl(UnlA),[CanA|_],[]),
                    concatena([modo:UnlA],CoCoisa,Lista));
                    (adj(unl(UnlA),[A|_],[]),
                    concatena([modo:UnlA],CoCoisa,Lista))).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                 Regras de projecao para complementos nominais                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S3, S4, S5, S8
verificaCnModo(sp([preposicao(P),sn(SN)]),Lista) :-
```

```
                verificaPrepDe(P,_),
                verificaSnModo(SN,Lista).

% S43
verificaCnModo(sp([preposicao(P),sadv(SADV)]),Lista) :-
                verificaPrepDe(P,_),
                verificaSadvTempoM(SADV,Lista).

% S76, S104
verificaCnModo(osscn([pcn(P),oss(ori(periodo(periodo_independente(PERIODO))))]),Lista) :-
                verificaPrepDe(P,_),
                verificaPeriodoModo(PERIODO,Lista).

% S76
verificaPeriodoModo(predicado(predv([svtd(verbo(V)),od(OD)])),[modoE:UnlV,ListaObj]) :-
                v(sin(inf_pess,td),unl(UnlV),cl(_),rest([suj(RestSuj),obj(RestObj)]),[V|_],[]),
                verificaOd(OD,RestObj,ListaObj).

% S104
verificaPeriodoModo(predicado(predv(svi(verbo(V)))),[modoE:UnlV]) :-
                v(sin(inf_pess,int),unl(UnlV),cl(_),rest([suj(RestSuj)]),[V|_],[]).

% S95
verificaCnModo(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepEm(P,_),
                verificaSnModo(SN,Lista).

% S106
verificaCnModo(sp([preposicao(P),sn(SN)]),[modoEm:Prep,Objeto]) :-
                verificaPrepEm(P,Prep),
                verificaSnObjeto(SN,RestObj,Objeto).

% S95
verificaCnModo(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepEm(P,_),
                verificaSnModo(SN,Lista).

% S8
verificaCnLugarD(sp([preposicao(P),sn(SN)]),Lista) :-
                (verificaPrepA(P,_); verificaPrepPara(P,_);verificaPrepAo(P,_)),
                verificaSnLugarD(SN,Lista).

% S12, S105
verificaCnCoCoisa(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepCom(P,_),
                verificaSnCoCoisa(SN,Lista).

% S18, S23, S32, S33
verificaCnLugar(sp([preposicao(P),sn(SN)]),Lista) :-
                (verificaPrepEm(P,_); verificaPrepNo(P,_)),
                verificaSnLugar(SN,Lista).

% S27, S31
verificaCnParceiro(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepCom(P,_),
                verificaSnParceiro(SN,Lista).

% S28
verificaCnRazao(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepPor(P,_),
                verificaSnRazao(SN,Lista).

% S33, S86, S119, S120
verificaCnDestino(sp([preposicao(P),sn(SN)]),Lista) :-
                (verificaPrepA(P,_); verificaPrepAo(P,_); verificaPrepPara(P,_)),
                verificaSnDestino(SN,Lista).

% S48, S54
verificaCnBeneficiario(sp([preposicao(P),sn(SN)]),Lista) :-
                (verificaPrepPara(P,_);verificaPrepSobre(P,_)),
```

```
                        verificaSnBeneficiario(SN,_,Lista).

% S55
verificaCnCena(sp([preposicao(P),sn(SN)]),Lista) :-
                        (verificaPrepEm(P,_); verificaPrepNo(P,_)),
                        verificaSnCena(SN,Lista).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                              Regras de projecao para preposições                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S3, S4, S5, S7, S8, S20, S36
verificaPrepDe(P,UnlP) :-
                        ((prep(unl(UnlP),[P|_],[]), P = de);
                        (prep(can(de),[P|_],[]),
                        prep(unl(UnlP),[de|_],[]))).

% S6, S14, S18, S23, S32, S33
verificaPrepEm(em,UnlP) :-
                        prep(unl(UnlP),[em|_],[]).

% S8, S33
verificaPrepA(a,UnlP) :-
                        prep(unl(UnlP),[a|_],[]).

% S12, S26, S27, S31
verificaPrepCom(P,UnlP) :-
                        prep(can(com),[P|_],[]);
                        (prep(unl(UnlP),[P|_],[]), P = com).

% S8, S15, S33
verificaPrepPara(P,UnlP) :-
                        prep(can(para),[P|_],[]);
                        (prep(unl(UnlP),[P|_],[]), P = para).

% S14, S18, S23, S24, S32, S33
verificaPrepNo(P,UnlP) :-
                        ((prep(unl(UnlP),[P|_],[]), P = no);
                        (prep(can(no),[P|_],[]),
                        prep(unl(UnlP),[no|_],[]))).

% S28
verificaPrepPor(por,UnlP) :-
                        prep(unl(UnlP),[por|_],[]).

% S8, S33
verificaPrepAo(P,UnlP) :-
                        ((prep(unl(UnlP),[P|_],[]), P = ao);
                        (prep(can(ao),[P|_],[]),
                        prep(unl(UnlP),[ao|_],[]))).

% S54
verificaPrepSobre(sobre,UnlP) :- prep(unl(UnlP),[sobre|_],[]).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                              Regras de projecao para advérbios                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S6, S117
verificaAadvo(aadvo_composto([aadvo_simples(AADVO1),coordenador(C),aadvo_simples(AADVO2)]),[Maneira1,Maneira2]) :-
                        verificaAadvo_simples1(AADVO1,Maneira1),
                        conj(sin(coord,adit),_,[C|_],[]),
                        verificaAadvo_simples2(AADVO2,Maneira2).

% S9, S30, S23, S39, S44, S45, S53, S55, S70, S78, S85, S87, S89, S92, S98, S99, S105, S106, S112
verificaAadvo(aadvo_simples(AADVO),Lista) :-
            verificaAadvo_simples(AADVO,Lista).
```

```
% S6
verificaAadvo_simples1(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepEm(P,_),
                verificaSnManeira1(SN,Lista).

verificaAadvo_simples2(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepEm(P,_),
                verificaSnManeira2(SN,Lista).

% S117
verificaAadvo_simples1(sadv(SADV),Lista) :-
                verificaSadvTempo1(SADV,Lista).

verificaAadvo_simples2(sadv(SADV),Lista) :-
                verificaSadvTempo2(SADV,Lista).

% S9, S23, S53, S85, S98, S99, S105
verificaAadvo_simples(sadv(SADV),Lista)  :-
                verificaSadvTempo(SADV,Lista).

% S26, S30, S45, S105
verificaAadvo_simples(sp([preposicao(P),sn(SN)]),Lista) :-
                verificaPrepCom(P,_),
                verificaSnManeira(SN,Lista).

% S28, S44, S70, S92, S106
verificaAadvo_simples(sadv(SADV),Lista)  :-
                verificaSadvManeira(SADV,Lista).

% S39
verificaAadvo_simples(sadv([adv(ADV),aadvl(aadvl_simples(sadv(SADV)))]),Lista)  :-
                verificaPrepPor(ADV,_),
                verificaSadvDuracao(SADV,Lista).

% S78
verificaAadvo_simples(sadv(SADV),Lista)  :-
                verificaSadvMeta(SADV,Lista).

% S70
verificaAadvo_simples(sadv(SADV),Lista)  :-
                verificaSadvLugar(SADV,Lista).

% S87
verificaAadvo_simples(sp([preposicao(P),sn(SN)]),Lista)  :-
                verificaPrepA(P,_),
                verificaSnLugar(SN,Lista).

% S89
verificaAadvo_simples(sp([preposicao(P),sn(SN)]),Lista)  :-
                verificaPrepPara(P,_),
                verificaSnProposito(SN,Lista).

% S104
verificaAadvo_simples(sadv([adv(ADV),aadvl(aadvl_simples(sadv(SADV)))]),[maneir:UnlAdv,Tempo])  :-
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                verificaSadvTempo(SADV,Tempo).

% S112
verificaAadvo_simples(sp([preposicao(P),sn(SN)]),Lista)  :-
                verificaPrepPara(P,_),
                verificaSnBeneficiario(SN,_,Lista).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                    Regras de projecao para sintagmas adverbiais                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S104
verificaSadvTempo([adv(ADV),cn(CN)],Lista) :-
                adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]),
```

```
                    verificaCnModo(CN,Modo),
                    concatena([tempo:UnlAdv],[Modo],Lista).

% S28
verificaSadvManeira(adv(ADV),[maneir:UnlAdv]) :-
                    adv(unl(UnlAdv),[ADV|_],[]).

% S9, S23, S51, S53, S85, S98, S99, S105
verificaSadvTempo(adv(ADV),[tempo:UnlAdv]) :-
                    adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]).

% S39
verificaSadvDuracao(adv(ADV),[duracao:UnlAdv]) :-
                    adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]).

% S43
verificaSadvTempoM(adv(ADV),[modo:UnlAdv]) :-
                     adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]).

% S44
verificaSadvManeira([adv(ADV),aadvl(aadvl_simples(sadv(SADV)))],Lista) :-
                    pron(sin(inde),unl(UnlPro),[ADV|_],[]),
                    verificaSadvManeira(SADV,Maneira),
                    concatena([quantidade:UnlPro],Maneira,Lista).

% S44
verificaSadvManeira(adv(ADV),[maneir:UnlAdv]) :-
                    adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]).

% S55, S106
verificaSadvManeira(adv(ADV),[maneir:UnlAdv]) :-
                    adv(sin(duv),unl(UnlAdv),[ADV|_],[]).

% S70
verificaSadvLugar(adv(ADV),[lugar:UnlAdv]) :-
                    adv(sin(cir_lug),unl(UnlAdv),[ADV|_],[]).

% S78
verificaSadvMeta(adv(ADV),[meta:UnlAdv]) :-
                    adv(sin(cir_lug),unl(UnlAdv),[ADV|_],[]),
                    adj(unl(_),[ADV|_],[]).

% S70, S74, S92
verificaSadvManeira(adv(ADV),[maneir:UnlAdv]) :-
                    adv(sin(int),unl(UnlAdv),[ADV|_],[]).

% 117
verificaSadvTempo1(adv(ADV),[tempo1:A]) :-
                    adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]),
                    string_atom(StrUnlA,UnlAdv),
                    strcat(StrUnlA,$.@entry$,StrA0),
                    string_atom(StrA0,A).

verificaSadvTempo2(adv(ADV),[tempo2:UnlAdv]) :-
                    adv(sin(cir_temp),unl(UnlAdv),[ADV|_],[]).

% S72-testes
verificaSadvManeira(adv(ADV),[maneir:UnlAdv]) :-
                    adv(sin(cir_mod),unl(UnlAdv),[ADV|_],[]).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                Regras de projecao para predicativos do sujeito           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S8, S12, S22, S27, S31, S37, S72, S73, S76, S77, S93, S98, S99, S101, S110
verificaPsujAtributo(psuj_simples(sadj(SADJ)),T,Lista) :-
                    verificaSadjAtributo(SADJ,T,Lista).

% S10, S17, S41, S53, S57, S60
```

```
verificaPsujAtributo(psuj_simples(sn(SN)),T,Lista) :-
                    verificaSnAtributo(SN,T,Lista).

% S36
verificaPsujAtributo(psuj_simples(sn(SN)),T,Adv,Lista) :-
                    verificaSnAtributo(SN,T,Adv,Lista).

% S40, S48, S55, S67
verificaPsujAtributo(psuj_simples(sadj(SADJ)),T,Aux,Lista) :-
                    verificaSadjAtributo(SADJ,T,Aux,Lista).

% S13, S25, S86, S122, S96, S106, S119, S120
verificaPsujMeta(psuj_simples(sadj(SADJ)),Lista) :-
                    verificaSadjMeta(SADJ,Lista).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      Regras de projecao para sintagmas adjetivais                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S8, S73, S110
verificaSadjAtributo([aadvl(aadvl_simples(sadv(adv(ADV)))),adj(ADJ)],T,[maneir:UnlAdv,atributo:At]) :-
                    adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                    ((adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[]));
                              adj(unl(UnlAdj),[ADJ|_],[])),
                    string_atom(StrT,T),
                    string_atom(StrUnlA,UnlAdj),
                    strcat(StrUnlA,$.@entry.@$,StrAt0),
                    strcat(StrAt0,StrT,StrAt1),
                    string_atom(StrAt1,At).

% S12
verificaSadjAtributo([aadvl(aadvl_simples(sadv(adv(ADV)))),adj(ADJ),cn(CN)],T,[maneir:UnlAdv,L2]) :-
                    verificaCnCoCoisa(CN,ListaCoCoisa),
                    adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                    ((adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[]));
                              adj(unl(UnlAdj),[ADJ|_],[])),
                    string_atom(StrT,T),
                    string_atom(StrUnlA,UnlAdj),
                    strcat(StrUnlA,$.@entry.@$,StrAt0),
                    strcat(StrAt0,StrT,StrAt1),
                    strcat(StrAt1,$.@$,StrAt2),
                    string_atom(StrAt1,At),
                    concatena([atributo:At],[ListaCoCoisa],L2).

% S13, S25
verificaSadjMeta([aadvl(aadvl_simples(sadv(adv(ADV)))),adj(ADJ)],[maneir:UnlAdv,meta:UnlAdj]) :-
                    adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                    ((adj(unl(UnlAdj),[ADJ|_],[]));
                              adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[])).

% S22, S27, S31, S37, S72, S77, S93, S98, S99, S101
verificaSadjAtributo(adj(ADJ),T,[atributo:At]) :-
                    ((adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[]));
                              adj(unl(UnlAdj),[ADJ|_],[])),
                    string_atom(StrT,T),
                    string_atom(StrUnlA,UnlAdj),
                    strcat(StrUnlA,$.@entry.@$,StrAt0),
                    strcat(StrAt0,StrT,StrAt1),
                    string_atom(StrAt1,At).

% S40
verificaSadjAtributo([aadvl(aadvl_simples(sadv(adv(ADV)))),adj(ADJ)],T,Aux,[maneir:UnlAdv,atributo:At]) :-
                    adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                    ((adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[]));
```

```
                        adj(unl(UnlAdj),[ADJ|_],[])),
                string_atom(StrT,T),
                string_atom(StrAux,Aux),
                string_atom(StrUnlA,UnlAdj),
                strcat(StrUnlA,$.@entry.@$,StrAt0),
                strcat(StrAt0,StrT,StrAt1),
                strcat(StrAt1,$.@$,StrAt2),
                strcat(StrAt2,StrAux,StrAt3),
                string_atom(StrAt3,At).

% S48
verificaSadjAtributo([adj(ADJ),cn(CN)],T,Aux,[atributo:At,Beneficiario]) :-
                verificaCnBeneficiario(CN,Beneficiario),
                ((adj(can(CanA),[ADJ|_],[]),
                        adj(unl(UnlAdj),[CanA|_],[]));
                        adj(unl(UnlAdj),[ADJ|_],[])),
                string_atom(StrT,T),
                string_atom(StrAux,Aux),
                string_atom(StrUnlA,UnlAdj),
                strcat(StrUnlA,$.@entry.@$,StrAt0),
                strcat(StrAt0,StrT,StrAt1),
                strcat(StrAt1,$.@$,StrAt2),
                strcat(StrAt2,StrAux,StrAt3),
                string_atom(StrAt3,At).

% S55
verificaSadjAtributo([aadvl(aadvl_simples(sadv(adv(ADV)))),adj(ADJ),cn(CN)],T,Aux,Lista) :-
                verificaCnCena(CN,Cena),
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                ((adj(can(CanA),[ADJ|_],[]),
                        adj(unl(UnlAdj),[CanA|_],[]));
                        adj(unl(UnlAdj),[ADJ|_],[])),
                string_atom(StrT,T),
                string_atom(StrAux,Aux),
                string_atom(StrUnlA,UnlAdj),
                strcat(StrUnlA,$.@entry.@$,StrAt0),
                strcat(StrAt0,StrT,StrAt1),
                strcat(StrAt1,$.@$,StrAt2),
                strcat(StrAt2,StrAux,StrAt3),
                string_atom(StrAt3,At),
                concatena([atributo:At],[Cena],L1),
                concatena([maneir:UnlAdv],L1,Lista).

% S122, S96
verificaSadjMeta(adj(ADJ),[meta:UnlAdj]) :-
                ((adj(unl(UnlAdj),[ADJ|_],[]));

                adj(can(CanA),[ADJ|_],[]),
                adj(unl(UnlAdj),[CanA|_],[])).

% S67
verificaSadjAtributo(adj(ADJ),T,Aux,[atributo:At]) :-
                ((adj(can(CanA),[ADJ|_],[]),
                        adj(unl(UnlAdj),[CanA|_],[]));
                        adj(unl(UnlAdj),[ADJ|_],[])),
                string_atom(StrT,T),
                string_atom(StrAux,Aux),
                string_atom(StrUnlA,UnlAdj),
                strcat(StrUnlA,$.@entry.@$,StrAt0),
                strcat(StrAt0,StrT,StrAt1),
                strcat(StrAt1,$.@$,StrAt2),
                strcat(StrAt2,StrAux,StrAt3),
                string_atom(StrAt3,At).

% S76
verificaSadjAtributo([adj(ADJ),cn(CN)],T,[atributo:At,Modo]) :-
                verificaCnModo(CN,Modo),
                ((adj(can(CanA),[ADJ|_],[]),
                        adj(unl(UnlAdj),[CanA|_],[]));
                        adj(unl(UnlAdj),[ADJ|_],[])),
```

```
                    string_atom(StrT,T),
                    string_atom(StrUnlA,UnlAdj),
                    strcat(StrUnlA,$.@entry.@$,StrAt0),
                    strcat(StrAt0,StrT,StrAt1),
                    string_atom(StrAt1,At).

% S82
verificaSadjMeta(adj(ADJ),Rest,[meta:UnlAdj]) :-
                    ((adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[]));
                              adj(unl(UnlAdj),[ADJ|_],[])).

% S86, S119, S120
verificaSadjMeta([adj(ADJ),cn(CN)],[meta:UnlAdj,Destino]) :-
                    verificaCnDestino(CN,Destino),
                    ((adj(can(CanAdj),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanAdj|_],[]));
                              adj(unl(UnlAdj),[ADJ|_],[])).

% S106
verificaSadjMeta([adj(ADJ),cn(CN)],[meta:UnlAdj,Modo]) :-
                    verificaCnModo(CN,Modo),
                    ((adj(can(CanA),[ADJ|_],[]),
                              adj(unl(UnlAdj),[CanA|_],[]));
                              adj(unl(UnlAdj),[ADJ|_],[])).



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      Regras de projecao para sintagmas nominais diversos                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S3
verificaSnModo([subst(S),aadnd(aadnd_simples(AADND))],[modo:M,Modo]) :-
                    verificaAadndModo(AADND,Modo),
                    ((s(_,unl(M),_,[S|_],[]));

                    (s(sin(N),can(CanS),[S|_],[]),
                              s(_,unl(UnlS),_,[CanS|_],[]),
                              string_atom(StrUnlS,UnlS),
                              string_atom(StrN,N),
                              strcat(StrUnlS,$.@$,StrM0),
                              strcat(StrM0,StrN,StrM1),
                              string_atom(StrM1,M))).

% S4, S56, S80, S95, S115
verificaSnModo(subst(S),[modo:M]) :-
                    ((s(_,unl(M),sem(_),[S|_],[]));

                    (s(sin(N),can(CanS),[S|_],[]),
                              s(_,unl(UnlS),sem(_),[CanS|_],[]),
                              string_atom(StrUnlS,UnlS),
                              string_atom(StrN,N),
                              strcat(StrUnlS,$.@$,StrM0),
                              strcat(StrM0,StrN,StrM1),
                              string_atom(StrM1,M))).

% S5
verificaSnModo([aadne(AADNE),subst(S)],Lista) :-
                    verificaAadnePosse(AADNE,Posse),
                    ((s(_,unl(M),_,[S|_],[]));

                    (s(sin(N),can(CanS),[S|_],[]),
                              s(_,unl(UnlS),_,[CanS|_],[]),
                              string_atom(StrUnlS,UnlS),
                              string_atom(StrN,N),
                              strcat(StrUnlS,$.@$,StrM0),
                              strcat(StrM0,StrN,StrM1),
                              string_atom(StrM1,M))),
                    concatena(Posse,[modo:M],Lista).
```

```
% S6
verificaSnManeira1([aadne(AADNE),subst(S)],[Modo,maneira1:M]) :-
                        verificaAadneModo(AADNE,Modo),
                        ((s(_,unl(UnlS),_,[S|_],[]),
                                string_atom(StrUnlS,UnlS),
                                strcat(StrUnlS,$.@entry$,StrM0),
                                string_atom(StrM0,M));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),_,[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@entry$,StrM0),
                                strcat(StrM0,StrN,StrM1),
                                string_atom(StrM1,M))).

verificaSnManeira2([aadne(AADNE),subst(S)],[Modo,maneira2:M]) :-
                        verificaAadneModo(AADNE,Modo),
                        ((s(_,unl(M),_,[S|_],[]));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),_,[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrM0),
                                strcat(StrM0,StrN,StrM1),
                                string_atom(StrM1,M))).

% S7
verificaSnLugarO([aadne(sdet(artigo(A))),subst(S)],Rest,[lugarO:L]) :-
                        art(sin(D),_,[A|_],[]),
                        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                                subconjunto(Tracos,Rest),
                                subconjunto(Tracos,[lugar]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrAr,D),
                                strcat(StrUnlS,$.@$,StrL0),
                                strcat(StrL0,StrAr,StrL1),
                                string_atom(StrL1,L));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                                subconjunto(Tracos,Rest),
                                subconjunto(Rest,[lugar]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrAr,D),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrL0),
                                strcat(StrL0,StrAr,StrL1),
                                strcat(StrL1,$.@$,StrL2),
                                strcat(StrL2,StrN,StrL3),
                                string_atom(StrL3,L))).

% S8
verificaSnModo([subst(S),cn(CN)],[modo:M],[Lugar]) :-
                        verificaCnLugarD(CN,Lugar),
                        ((s(_,unl(M),sem(_),[S|_],[]));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrM0),
                                strcat(StrM0,StrN,StrM1),
                                string_atom(StrM1,M))).

verificaSnLugarD([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],[lugarD:L,Modo]) :-
                        art(sin(D),_,[A|_],[]),
                        verificaAadndModo(AADND,Modo),
```

```
		((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
			subconjunto(Tracos,[lugar]),
			string_atom(StrUnlS,UnlS),
			string_atom(StrAr,D),
			strcat(StrUnlS,$.@$,StrL0),
			strcat(StrL0,StrAr,StrL1),
			string_atom(StrL1,L));

		(s(sin(N),can(CanS),[S|_],[]),
			s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
			subconjunto(Tracos,[lugar]),
			string_atom(StrUnlS,UnlS),
			string_atom(StrAr,D),
			string_atom(StrN,N),
			strcat(StrUnlS,$.@$,StrL0),
			strcat(StrL0,StrAr,StrL1),
			strcat(StrL1,$.@$,StrL2),
			strcat(StrL2,StrN,StrL3),
			string_atom(StrL3,L))).

% S10, S17
verificaSnAtributo([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],T,[atributo:At,Modo]) :-
		art(sin(D),_,[A|_],[]),
		verificaAadndModo(AADND,Modo),
		((s(_,unl(UnlS),sem(_),[S|_],[]),
			string_atom(StrUnlS,UnlS),
			string_atom(StrT,T),
			string_atom(StrAr,D),
			strcat(StrUnlS,$.@entry.@$,StrAt0),
			strcat(StrAt0,StrT,StrAt1),
			strcat(StrAt1,$.@$,StrAt2),
			strcat(StrAt2,StrAr,StrAt3),
			string_atom(StrAt3,At));

		(s(sin(N),can(CanS),[S|_],[]),
			s(_,unl(UnlS),sem(_),[CanS|_],[]),
			string_atom(StrUnlS,UnlS),
			string_atom(StrT,T),
			string_atom(StrAr,D),
			string_atom(StrN,N),
			strcat(StrUnlS,$.@entry.@$,StrA0),
			strcat(StrAt0,StrT,StrAt1),
			strcat(StrAt1,$.@$,StrAt2),
			strcat(StrAt2,StrAr,StrAt3),
			strcat(StrAt3,$.@$,StrAt4),
			strcat(StrAt4,StrN,StrAt5),
			string_atom(StrAt5,At))).

% S12
verificaSnCoCoisa([aadne(AADNE),subst(S),aadnd(aadnd_simples(AADND))],Lista) :-
		verificaAadnePosse(AADNE,Posse),
		verificaAadndModo(AADND,Modo),
		((s(_,unl(C),sem(_),[S|_],[]),
			concatena([co_coisa:C],[Modo],L2));

		(s(sin(N),can(CanS),[S|_],[]),
			s(_,unl(UnlS),sem(_),[CanS|_],[]),
			string_atom(StrUnlS,UnlS),
			string_atom(StrN,N),
			strcat(StrUnlS,$.@$,StrC0),
			strcat(StrC0,StrN,StrC1),
			string_atom(StrC1,C),
			concatena([co_coisa:C],[Modo],L2))),
		concatena(Posse,L2,Lista).

% S14, S33
verificaSnLugar(subst(S),Rest,[lugar:L]) :-
		((s(_,unl(L),sem(Tracos),[S|_],[]),
			subconjunto(Tracos,Rest),
			subconjunto(Tracos,[lugar]));
```

```
                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        subconjunto(Tracos,[lugar]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrL0),
                        strcat(StrL0,StrN,StrL1),
                        string_atom(StrL1,L))).

% S15
verificaSnBeneficiario([aadne(AADNE),subst(S)],Rest,Lista) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(B),_,[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),_,[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrB0),
                        strcat(StrB0,StrN,StrB1),
                        string_atom(StrB1,B),
                        concatena([beneficiario:B],[],L1))),
                concatena(Posse,[beneficiario:B],Lista).

% S18, S23,
verificaSnLugar(subst(S),[lugar:L]) :-
                ((s(_,unl(L),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,[lugar]),
                        subconjunto(Tracos,[lugar]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Rest,[lugar]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrL0),
                        strcat(StrL0,StrN,StrL1),
                        string_atom(StrL1,L))).
% S20
verificaSnOrigem([aadne(AADNE),subst(S)],Lista) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(O),_,[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),_,[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))),
                concatena(Posse,[origem:O],Lista).

% S26
verificaSnManeira([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],[Modo,maneir:M]) :-
                art(sin(D),_,[A|_],[]),
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrAr,StrM1),
                        string_atom(StrM1,M));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
```

```
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrAr,StrM1),
                        strcat(StrM1,$.@$,StrM2),
                        strcat(StrM2,StrN,StrM3),
                        string_atom(StrM3,M))).

% S27
verificaSnParceiro([aadne(sdet(artigo(A))),subst(S)],[parceiro:P]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrAr,StrP1),
                        string_atom(StrP1,P));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrAr,StrP1),
                        strcat(StrP1,$.@$,StrP2),
                        strcat(StrP2,StrN,StrP3),
                        string_atom(StrP3,P))).

% S28
verificaSnRazao(subst(S),[razao:R]) :-
                ((s(_,unl(R),sem(_),[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrR0),
                        strcat(StrR0,StrN,StrR1),
                        string_atom(StrR1,R))).


% S30
verificaSnManeira([aadne(AADNE),subst(S)],[Modo,maneir:M]) :-
                verificaAadneModo(AADNE,Modo),
                ((s(_,unl(M),_,[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),_,[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrN,StrM1),
                        string_atom(StrM1,M))).

% S32
verificaSnLugar([aadne(AADNE),subst(S)],[Modo,lugar:L]) :-
                verificaAadneModo(AADNE,Modo),
                ((s(_,unl(L),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,[lugar]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,[lugar]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrL0),
                        strcat(StrL0,StrN,StrL1),
                        string_atom(StrL1,L))).
```

```
% S33
verificaSnDestino([aadne(AADNE),subst(S)],Lista) :-
                        verificaAadnePosse(AADNE,Posse),
                        ((s(_,unl(D),sem(_),[S|_],[]));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrD0),
                                strcat(StrD0,StrN,StrD1),
                                string_atom(StrD1,D))),
                        concatena(Posse,[destino:D],Lista).

% S36
verificaSnAtributo([subst(S),aadnd(aadnd_simples(AADND))],T,Adv,[atributo:At,Modo]) :-
                        verificaAadndModo(AADND,Modo),
                        ((s(_,unl(UnlS),sem(_),[S|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrT,T),
                                string_atom(StrAdv,Adv),
                                strcat(StrUnlS,$.@entry.@$,StrAt0),
                                strcat(StrAt0,StrT,StrAt1),
                                strcat(StrAt1,$.@$,StrAt2),
                                strcat(StrAt2,StrAdv,StrAt3),
                                string_atom(StrAt3,At));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrT,T),
                                string_atom(StrAdv,Adv),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@entry.@$,StrAt0),
                                strcat(StrAt0,StrT,StrAt1),
                                strcat(StrAt1,$.@$,StrAt2),
                                strcat(StrAt2,StrAdv,StrAt3),
                                strcat(StrAt3,$.@$,StrAt4),
                                strcat(StrAt4,StrN,StrAt),
                                string_atom(StrAt,At))).

% S41, S57
verificaSnAtributo([aadne(sdet(artigo(A))),subst(S)],T,[atributo:At]) :-
                        art(sin(D),_,[A|_],[]),
                        ((s(_,unl(UnlS),sem(_),[S|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrT,T),
                                string_atom(StrAr,D),
                                strcat(StrUnlS,$.@entry.@$,StrAt0),
                                strcat(StrAt0,StrT,StrAt1),
                                strcat(StrAt1,$.@$,StrAt2),
                                strcat(StrAt2,StrAr,StrAt3),
                                string_atom(StrAt3,At));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrT,T),
                                string_atom(StrAr,D),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@entry.@$,StrA0),
                                strcat(StrAt0,StrT,StrAt1),
                                strcat(StrAt1,$.@$,StrAt2),
                                strcat(StrAt2,StrAr,StrAt3),
                                strcat(StrAt3,$.@$,StrAt4),
                                strcat(StrAt4,StrN,StrAt5),
                                string_atom(StrAt5,At))).
```

```
% S45
verificaSnManeira(subst(S),[maneir:M]) :-
                        ((s(_,unl(M),_,[S|_],[]));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),_,[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrM0),
                                strcat(StrM0,StrN,StrM1),
                                string_atom(StrM1,M))).

% S53
verificaSnAtributo([subst(S),aadnd(aadnd_simples(AADND))],T,Lista) :-
                        verificaAadndProposito(AADND,Proposito),
                        ((s(_,unl(UnlS),sem(_),[S|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrT,T),
                                strcat(StrUnlS,$.@entry.@$,StrAt0),
                                strcat(StrAt0,StrT,StrAt1),
                                string_atom(StrAt1,At),
                                concatena([atributo:At],Proposito,Lista));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrT,T),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@entry.@$,StrAt0),
                                strcat(StrAt0,StrT,StrAt1),
                                strcat(StrAt1,$.@$,StrAt2),
                                strcat(StrAt2,StrN,StrAt),
                                string_atom(StrAt,At),
                                concatena([atributo:At],Proposito,Lista))).

% S53
verificaSnProposito(subst(S),[proposito:P]) :-
                        ((s(_,unl(P),sem(_),[S|_],[]));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrP0),
                                strcat(StrP0,StrN,StrP1),
                                string_atom(StrP1,P))).

% S55
verificaSnCena([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],[cena:C,Modo]) :-
                        art(sin(D),_,[A|_],[]),
                        verificaAadndModo(AADND,Modo),
                        ((s(_,unl(UnlS),sem(_),[S|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrAr,D),
                                strcat(StrUnlS,$.@$,StrC0),
                                strcat(StrC0,StrAr,StrC1),
                                string_atom(StrC1,C));

                        (s(sin(N),can(CanS),[S|_],[]),
                                s(_,unl(UnlS),sem(_),[CanS|_],[]),
                                string_atom(StrUnlS,UnlS),
                                string_atom(StrAr,D),
                                string_atom(StrN,N),
                                strcat(StrUnlS,$.@$,StrC0),
                                strcat(StrC0,StrAr,StrC1),
                                strcat(StrC1,$.@$,StrC2),
                                strcat(StrC2,StrN,StrC3),
                                string_atom(StrC3,C))).
```

```
% S60
verificaSnAtributo([aadne(sdet(artigo(A))),subst(S),cn(CN)],T,[atributo:At,CoCoisa]) :-
                art(sin(D),_,[A|_],[]),
                verificaCnCoCoisa(CN,CoCoisa),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrT,T),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@entry.@$,StrAt0),
                        strcat(StrAt0,StrT,StrAt1),
                        strcat(StrAt1,$.@$,StrAt2),
                        strcat(StrAt2,StrAr,StrAt3),
                        string_atom(StrAt3,At));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrT,T),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,StrA0),
                        strcat(StrAt0,StrT,StrAt1),
                        strcat(StrAt1,$.@$,StrAt2),
                        strcat(StrAt2,StrAr,StrAt3),
                        strcat(StrAt3,$.@$,StrAt4),
                        strcat(StrAt4,StrN,StrAt5),
                        string_atom(StrAt5,At))).

% S60
verificaSnCoCoisa([subst(S),aadnd(aadnd_simples(AADND))],[co_coisa:C,Modo]) :-
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(C),sem(_),[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrC0),
                        strcat(StrC0,StrN,StrC1),
                        string_atom(StrC1,C))).

% S66, S111
verificaSnLugar([aadne(sdet(artigo(A))),subst(S)],Rest,[lugar:L]) :-
                ((s(_,unl(L),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        subconjunto(Tracos,[lugar]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        subconjunto(Tracos,[lugar]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrL0),
                        strcat(StrL0,StrN,StrL1),
                        string_atom(StrL1,L))).

% S66
verificaSnModo([subst(S),cn(CN)],[modo:M,Cena]) :-
                verificaCnCena(CN,Cena),
                ((s(_,unl(M),sem(_),[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrN,StrM1),
                        string_atom(StrM1,M))).
```

```
% S71
verificaSnModo([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Lista) :-
                art(sin(D),_,[A|_],[]),
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrAr,StrM1),
                        string_atom(StrM1,M),
                        concatena([modo:M],Modo,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrAr,StrM1),
                        strcat(StrM1,$.@$,StrM2),
                        strcat(StrM2,StrN,StrM3),
                        string_atom(StrM3,M),
                        concatena([modo:M],Modo,Lista))).

% S83
verificaSnDestino([aadne(sdet(artigo(A))),subst(S)],Rest,[destino:De]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrDe0),
                        strcat(StrDe0,StrAr,StrDe1),
                        string_atom(StrDe1,De));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrDe0),
                        strcat(StrDe0,StrAr,StrDe1),
                        strcat(StrDe1,$.@$,StrDe2),
                        strcat(StrDe2,StrN,StrDe3),
                        string_atom(StrDe3,De))).

% S86
verificaSnDestino([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],[destino:De,Modo]) :-
                art(sin(D),_,[A|_],[]),
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrDe0),
                        strcat(StrDe0,StrAr,StrDe1),
                        string_atom(StrDe1,De));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrDe0),
                        strcat(StrDe0,StrAr,StrDe1),
                        strcat(StrDe1,$.@$,StrDe2),
                        strcat(StrDe2,StrN,StrDe3),
                        string_atom(StrDe3,De))).
```

```
% S86
verificaSnModo([aadne(sdet(artigo(A))),subst(S)],[modo:M]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrAr,StrM1),
                        string_atom(StrM1,M));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrM0),
                        strcat(StrM0,StrAr,StrM1),
                        strcat(StrM1,$.@$,StrM2),
                        strcat(StrM2,StrN,StrM3),
                        string_atom(StrM3,M))).

% S87
verificaSnLugar([aadne(AADNE),subst(S)],[Posse,lugar:L]) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(L),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,[lugar]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Rest,[lugar]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrL0),
                        strcat(StrL0,StrN,StrL1),
                        string_atom(StrL1,L))).

% S89
verificaSnProposito([aadne(AADNE),subst(S)],Lista) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(P),_,[S|_],[]));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),_,[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrP0),
                        strcat(StrP0,StrN,StrP1),
                        string_atom(StrP1,P))),
                concatena(Posse,[proposito:P],Lista).

% S91
verificaSnProposito(predicado(predv([svti(verbo(V)),oi(OI)])),Rest,[propositoE:UnlV,ListaObj]) :-
                v(sin(inf_pess,ti),unl(UnlV),cl(_),rest([suj(RestSuj),obj(RestObj)]),[V|_],[]),
                verificaOi(OI,RestObj,ListaObj).

% S105
verificaSnManeira([aadvl(aadvl_simples(sadv(adv(ADV)))),subst(S)],[maneir:UnlAdv,maneir:O]) :-
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).
```

```
% S109
verificaSnCoCoisa([aadne(sdet(artigo(A))),subst(S)],[co_coisa:C]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrC0),
                        strcat(StrC0,StrAr,StrC1),
                        string_atom(StrC1,C));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrC0),
                        strcat(StrC0,StrAr,StrC1),
                        strcat(StrC1,$.@$,StrC2),
                        strcat(StrC2,StrN,StrC3),
                        string_atom(StrC3,C))).

% S112
verificaSnBeneficiario(pron_subst(PS),Rest,[beneficiario:UnlPro]) :-
                pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).

% S112
verificaSnAposto([aadne(sdet(artigo(A))),nome_proprio(S),aadnd(aadnd_simples(AADND))],Lista) :-
                verificaAadndModo(AADND,Modo),
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrAp0),
                        strcat(StrAp0,StrAr,StrAp1),
                        string_atom(StrAp1,Ap),
                        concatena([conteudo:Ap],Modo,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(_),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrAp0),
                        strcat(StrAp0,StrAr,StrAp1),
                        strcat(StrAp1,$.@$,StrAp2),
                        strcat(StrAp2,StrN,StrAp3),
                        string_atom(StrAp3,Ag),
                        concatena([conteudo:Ap],Modo,Lista))).

% S119, S120
verificaSnDestino([aadne(sdet(artigo(A))),subst(S)],[destino:De]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(_),[S|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrDe0),
                        strcat(StrDe0,StrAr,StrDe1),
                        string_atom(StrDe1,De));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrDe0),
                        strcat(StrDe0,StrAr,StrDe1),
                        strcat(StrDe1,$.@$,StrDe2),
                        strcat(StrDe2,StrN,StrDe3),
                        string_atom(StrDe3,De))).
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                         Regras de projecao para objetos                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S1, S2, S3, S4, S5, S7, S11, S15, S18, S20, S23, S26, S32, S33, S34, S35, S43, S44, S46, S47, S49, S52,
% S54, S56, S59, S61, S63, S64, S65, S68, S69, S71, S72, S74, S76, S78, S80, S82, S83, S85, S91, S95, S97,
% S101, S102, S103, S107, S108, S109, S115, S116, S117
verificaOd(od_simples(sn(SN)),RestObj,Lista) :-
            verificaSnObjeto(SN,RestObj,Lista).

% S21, S62, S113
verificaOd(od_composto([od_simples(sn(SN1)),coordenador(C),od_simples(sn(SN2))]),RestObj,[Objeto1,Objeto2]) :-
            verificaSnObjeto1(SN1,RestObj,Objeto1),
            conj(sin(coord,adit),_,[C|_],[]),
            verificaSnObjeto2(SN2,RestObj,Objeto2).

% S84
verificaOd(od_composto([od_simples(sn(SN1)),coordenador(C),od_simples(sn(SN2))]),RestObj,[Objeto3,Objeto4]) :-
            verificaSnObjeto3(SN1,RestObj,Objeto3),
            conj(sin(coord,alter),_,[C|_],[]),
            verificaSnObjeto4(SN2,RestObj,Objeto4).

% S7
verificaOi(oi_simples([poi(P),sn(SN)]),RestObjI,Lista) :-
            verificaPrepDe(P,_),
            verificaSnLugarO(SN,RestObjI,Lista).

% S14, S66, S111
verificaOi(oi_simples([poi(P),sn(SN)]),RestObjI,Lista) :-
            (verificaPrepEm(P,_); verificaPrepNo(P,_)),
            verificaSnLugar(SN,RestObjI,Lista).

% S24
verificaOi(oi_simples([poi(P),sn(SN)]),RestObj,[objeto:Prep,Objeto]) :-
            verificaPrepNo(P,Prep),
            verificaSnObjeto(SN,RestObj,Objeto).

% S38
verificaOi(oi_simples([poi(P),sn(SN)]),RestObj,[meta:Prep,Objeto]) :-
            verificaPrepEm(P,Prep),
            verificaSnObjeto(SN,RestObj,Objeto).

% S39, S90
verificaOi(oi_composto([oi_simples([poi(P),sn(SN1)]),coordenador(C),oi_simples([poi(P),sn(SN2)])]),RestObj,[Objeto1,Objeto2])
:-
            verificaPrepDe(P,Prep),
            verificaSnObjeto1(SN1,RestObj,Objeto1),
            conj(sin(coord,adit),_,[C|_],[]),
            verificaSnObjeto2(SN2,RestObj,Objeto2).

% S42, S75, S79, S91, S100, S121
verificaOi(oi_simples([poi(P),sn(SN)]),RestObj,Lista) :-
            verificaPrepDe(P,_),
            verificaSnObjeto(SN,RestObj,Lista).

% S58
verificaOi(oi_simples([poi(P),sn(SN)]),RestObjI,Lista) :-
            verificaPrepCom(P,_),
            verificaSnObjeto(SN,RestObjI,Lista).

% S83
verificaOi(oi_simples(sp([preposicao(P),sn(SN)])),RestObjI,Lista) :-
            verificaPrepPara(P,_),
            verificaSnDestino(SN,RestObjI,Lista).

% S91
verificaOi(ossoi([poi(P),oss(ori(periodo(periodo_independente(PERIODO))))]),RestObjI,Lista) :-
            verificaPrepPara(P,_),
            verificaSnProposito(PERIODO,RestObjI,Lista).
```

```
% S114
verificaOi(oi_simples([poi(P),sn(SN)]),RestObj,[objeto:Prep,Objeto]) :-
        verificaPrepEm(P,Prep),
        verificaSnObjeto(SN,RestObj,Objeto).

% S118
verificaOi(oi_composto([oi_simples([poi(P),sn(SN1)]),coordenador(C),oi_simples([poi(P),sn(SN2)])]),RestObj,[objeto:Prep,L1]) :-
        verificaPrepEm(P,Prep),
        verificaSnObjeto1(SN1,RestObj,Objeto1),
        conj(sin(coord,adit),_,[C|_],[]),
        verificaSnObjeto2(SN2,RestObj,Objeto2),
        concatena([Objeto1],[Objeto2],L1).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      Regras de projecao para sintagmas nominais objetos                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% S121
verificaSnObjeto([aadne(AADNE),subst(S)],Rest,[Quantidade,objeto:O]) :-
        verificaAadneQuantidade(AADNE,Quantidade),
        ((s(_,unl(O),sem(Tracos),[S|_],[]),
                subconjunto(Tracos,Rest));

        (s(sin(N),can(CanS),[S|_],[]),
                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrN,N),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrN,StrO1),
                string_atom(StrO1,O))).

% S1, S2, S59
verificaSnObjeto([aadne(AADNE),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
        verificaAadnePosse(AADNE,Posse),
        verificaAadndModo(AADND,Modo),
        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                subconjunto(Tracos,Rest),
                concatena([objeto:UnlS],[Modo],L2));

        (s(sin(N),can(CanS),[S|_],[]),
                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrN,N),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrN,StrO1),
                string_atom(StrO1,O),
                concatena([objeto:O],[Modo],L2))),
        concatena(Posse,L2,Lista).

% S3, S4
verificaSnObjeto([aadne(AADNE),subst(S),cn(CN)],Rest,Lista) :-
        verificaAadnePosse(AADNE,Posse),
        verificaCnModo(CN,Modo),
        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                subconjunto(Tracos,Rest),
                concatena([objeto:UnlS],[Modo],L2));

        (s(sin(N),can(CanS),[S|_],[]),
                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrN,N),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrN,StrO1),
                string_atom(StrO1,O),
                concatena([objeto:O],[Modo],L2))),
        concatena(Posse,L2,Lista).
```

```
% S5, S95
verificaSnObjeto([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,[objeto:O,Modo]) :-
                art(sin(D),_,[A|_],[]),
                verificaCnModo(CN,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        strcat(StrO1,$.@$,StrO2),
                        strcat(StrO2,StrN,StrO3),
                        string_atom(StrO3,O))).

% S7
verificaSnObjeto([aadne(AADNE),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                verificaAadneModo(AADNE,Modo1),
                verificaAadndModo(AADND,Modo2),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([objeto:UnlS],[Modo2],L2));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],[Modo2],L2))),
                concatena([Modo1],L2,Lista).


% S11, S24, S46, S72, S85, S121
verificaSnObjeto([aadne(AADNE),subst(S)],Rest,[Modo,objeto:O]) :-
                verificaAadneModo(AADNE,Modo),
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

% S15
verificaSnObjeto([aadvl(aadvl_simples(sadv(adv(ADV)))),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                verificaAadndBeneficiario(AADND,Beneficiario),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([objeto:UnlS],[Beneficiario],L2));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
```

```
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],[Beneficiario],L2))),
                concatena([maneir:UnlAdv],L2,Lista).

% S18, S23, S34, S56, S91, S116
verificaSnObjeto([aadne(AADNE),subst(S)],Rest,[Posse,objeto:O]) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

% S20
verificaSnObjeto([aadvl(aadvl_simples(sadv(adv(ADV)))),subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                verificaAadndOrigem(AADND,Origem),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([objeto:UnlS],[Origem],L2));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],[Origem],L2))),
                concatena([maneir:UnlAdv],L2,Lista).


% S21, S90, S113
verificaSnObjeto1(subst(S),Rest,[objeto1:O]) :-
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        strcat(StrUnlS,$.@entry$,StrO0),
                        string_atom(StrO0,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

verificaSnObjeto2(subst(S),Rest,[objeto2:O]) :-
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
```

```
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

% S26, S49, S60, S69, S74, S75, S78, S79, S91, S107, S108, S114, S115
verificaSnObjeto([aadne(sdet(artigo(A))),subst(S)],Rest,[objeto:O]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        strcat(StrO1,$.@$,StrO2),
                        strcat(StrO2,StrN,StrO3),
                        string_atom(StrO3,O))).

% S32
verificaSnObjeto([subst(S),cn(CN)],Rest,[objeto:O,Lugar]) :-
                verificaCnLugar(CN,Lugar),
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

% S33
verificaSnObjeto([aadvl(aadvl_simples(sadv(adv(ADV)))),subst(S),cn(CN)],Rest,Lista) :-
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                verificaCnDestino(CN,Destino),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([objeto:UnlS],[Destino],L2));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],[Destino],L2))),
                concatena([maneir:UnlAdv],L2,Lista).

% S35, S44, S68, S100, S102
verificaSnObjeto(subst(S),Rest,[objeto:O]) :-
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
```

```
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

% S38, S76, S80, S101, S103, S109, S117
verificaSnObjeto([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,[objeto:O,Modo]) :-
                art(sin(D),_,[A|_],[]),
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        strcat(StrO1,$.@$,StrO2),
                        strcat(StrO2,StrN,StrO3),
                        string_atom(StrO3,O))).

% S47, S51
verificaSnObjeto([aadvl(aadvl_simples(sadv(adv(ADV)))),subst(S)],Rest,[objeto:O,maneir:UnlAdv]) :-
                adv(sin(int),unl(UnlAdv),[ADV|_],[]),
                ((s(_,unl(O),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O))).

% S54
verificaSnObjeto([aadne(AADNE),subst(S),cn(CN)],Rest,Lista) :-
                verificaAadneModo(AADNE,Modo),
                verificaCnBeneficiario(CN,Beneficiario),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([objeto:UnlS],[Beneficiario],L1));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],[Beneficiario],L1))),
                concatena([Modo],L1,Lista).

% S58
verificaSnObjeto([aadne(AADNE),subst(S)],Rest,Lista) :-
                verificaAadneModo(AADNE,D,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrO0),
```

```
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],Modo,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        strcat(StrO1,$.@$,StrO2),
                        strcat(StrO2,StrN,StrO3),
                        string_atom(StrO3,O),
                        concatena([objeto:O],Modo,Lista))).


% S62, S118
verificaSnObjeto1([aadne(AADNE),subst(S)],Rest,Lista) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        strcat(StrUnlS,$.@entry$,StrO0),
                        string_atom(StrO0,O),
                        concatena(Posse,[objeto1:O],Lista));

        (s(sin(N),can(CanS),[S|_],[]),
                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrN,N),
                strcat(StrUnlS,$.@entry.@$,StrO0),
                strcat(StrO0,StrN,StrO1),
                string_atom(StrO1,O),
                concatena(Posse,[objeto1:O],Lista))).

verificaSnObjeto2([aadne(AADNE),subst(S)],Rest,Lista) :-
                verificaAadnePosse(AADNE,Posse),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena(Posse,[objeto2:UnlS],Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena(Posse,[objeto2:O],Lista))).

% S63
verificaSnObjeto([subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
                verificaAadndModo(AADND,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        concatena([objeto:UnlS],Modo,Lista));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrN,StrO1),
                        string_atom(StrO1,O),
                        concatena([objeto:O],Modo,Lista))).
```

```
% S64, S65
verificaSnObjeto([aadne(sdet(artigo(A))),subst(S)],Rest,[objeto:O]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        strcat(StrO1,$.@$,StrO2),
                        strcat(StrO2,StrN,StrO3),
                        string_atom(StrO3,O))).

% S71
verificaSnObjeto([aadne(sdet(artigo(A))),subst(S),cn(CN)],Rest,[objeto:O,Modo]) :-
                art(sin(D),_,[A|_],[]),
                verificaCnModo(CN,Modo),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        strcat(StrO1,$.@$,StrO2),
                        strcat(StrO2,StrN,StrO3),
                        string_atom(StrO3,O))).

% S82, S83
verificaSnObjeto(pron_subst(PS),Rest,[objeto:UnlPro]) :-
                pron(sin(trat),unl(UnlPro),sem(Tracos),[PS|_],[]),
                subconjunto(Tracos,Rest).

% S84
verificaSnObjeto3([aadne(sdet(artigo(A))),subst(S)],Rest,[objeto3:O]) :-
                art(sin(D),_,[A|_],[]),
                ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        strcat(StrUnlS,$.@entry.@$,StrO0),
                        strcat(StrO0,StrAr,StrO1),
                        string_atom(StrO1,O));

                (s(sin(N),can(CanS),[S|_],[]),
                        s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                        subconjunto(Tracos,Rest),
                        string_atom(StrUnlS,UnlS),
                        string_atom(StrAr,D),
                        string_atom(StrN,N),
                        strcat(StrUnlS,$.@entry.@$,StrO0),
```

```
                    strcat(StrO0,StrAr,StrO1),
                    strcat(StrO1,$.@$,StrO2),
                    strcat(StrO2,StrN,StrO3),
                    string_atom(StrO3,O))).

% S84
verificaSnObjeto4([aadne(sdet(artigo(A))),subst(S),aadnd(aadnd_simples(AADND))],Rest,[objeto4:O,Modo]) :-
        art(sin(D),_,[A|_],[]),
        verificaAadndModo(AADND,Modo),
        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrAr,D),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrAr,StrO1),
                string_atom(StrO1,O));

        (s(sin(N),can(CanS),[S|_],[]),
                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrAr,D),
                string_atom(StrN,N),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrAr,StrO1),
                strcat(StrO1,$.@$,StrO2),
                strcat(StrO2,StrN,StrO3),
                string_atom(StrO3,O))).

% S97
verificaSnObjeto([pron_subst(S),aadnd(aadnd_simples(AADND))],Rest,Lista) :-
        verificaAadndModo(AADND,Modo),
        pron(sin(inde),unl(UnlS),[S|_],[]),
        concatena([objeto:UnlS],Modo,Lista).

% S106
verificaSnObjeto([aadne(sdet(artigo(A))),subst(S)],[objeto:O]) :-
        art(sin(D),_,[A|_],[]),
        ((s(_,unl(UnlS),sem(Tracos),[S|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrAr,D),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrAr,StrO1),
                string_atom(StrO1,O));

        (s(sin(N),can(CanS),[S|_],[]),
                s(_,unl(UnlS),sem(Tracos),[CanS|_],[]),
                subconjunto(Tracos,Rest),
                string_atom(StrUnlS,UnlS),
                string_atom(StrAr,D),
                string_atom(StrN,N),
                strcat(StrUnlS,$.@$,StrO0),
                strcat(StrO0,StrAr,StrO1),
                strcat(StrO1,$.@$,StrO2),
                strcat(StrO2,StrN,StrO3),
                string_atom(StrO3,O))).
```

4.2 *Templates* de Relacionamento

Os templates são disparados pelas regras de projeção mais genéricas, para sentenças, por meio do predicado "gera/2". Esse predicado assume como argumento de entrada a estrutura conceitual intermediária e retorna como saída a estrutura conceitual final, em UNL. Ele dispara um predicado auxiliar, "simplifica/4", que realiza uma série de procedimentos sobre a estrutura conceitual intermediária para identificar cada um dos pares de conceitos que podem ser relacionados. A partir dessa identificação é que são aplicados os templates de relacionamento, propriamente ditos, um para cada par de conceitos,

com base nos seus papéis semânticos. Esses procedimentos são descritos na Seção 6. A seguir, somente o código dos templates é apresentado, ilustrando os **103** *templates* distintos.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                      Templates de relacionamento                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rel('posse-objeto', P, O, pos(O, P), 2).
rel('objeto-modo', O, M, md(O, M), 1).
rel('evento-objeto', E, O, obj(E, O), 1).
rel('modo-modo', M1, M2, md(M1, M2), 1).
rel('posse-modo', P, M, pos(M, P), 2).
rel('experimentador-evento', Ex, Ev, aoj(Ev, Ex), 2).
rel('modo-maneira1', Mo, Ma, md(Ma, Mo), 2).
rel('modo-maneira2', Mo, Ma, md(Ma, Mo), 2).
rel('maneira1-maneira2', M1, M2, and01(M1, M2), 1).
rel('evento-maneira1', E, M, man(E, a01), 1).
rel('paciente-evento', P, E, obj(E, P), 2).
rel('paciente-modo', P, M, md(P, M), 1).
rel('evento-lugarO', E, L, plf(E, L), 1).
rel('modo-objeto', M, O, md(O, M), 2).
rel('objeto-evento', O, E, obj(E, O), 2).
rel('lugarD-modo', L, M, md(L, M), 1).
rel('modo-lugarD', M, L, plt(M, L), 1).
rel('inativo-modo', I, M, md(I, M), 1).
rel('maneir-atributo', M, A, man(A, M), 2).
rel('inativo-atributo', I, A, aoj(A, I), 2).
rel('evento-tempo', E, T, tim(E, T), 1).
rel('atributo-modo', A, M, md(A, M), 1).
rel('agente-evento', A, E, agt(E, A), 2).
rel('posse-co_coisa', P, C, pos(C, P), 2).
rel('atributo-co_coisa', A, C, cao(A, C), 1).
rel('co_coisa-modo', C, M, md(C, M), 1).
rel('maneir-meta', Ma, Me, man(Me, Ma), 2).
rel('maneir-atributo', M, A, man(A, M), 2).
rel('evento-meta', E, M, gol(E, M), 1).
rel('experimentador-atributo', E, A, aoj(A, E), 2).
rel('evento-lugar', E, L, plc(E, L), 1).
rel('objeto-beneficiario', O, B, ben(O, B), 1).
rel('maneir-objeto', M, O, man(O, M), 2).
rel('inativo-evento', I, E, aoj(E, I), 2).
rel('agente-lugar', A, L, plc(A, L), 1).
rel('posse-paciente', Po, Pa, pos(Pa, Po), 2).
rel('inativo1-modo', I, M, md(I, M), 1).
rel('inativo2-modo', I, M, md(I, M), 1).
rel('inativo1-inativo2', I1, I2, and01(I1, I2), 1).
rel('inativo1-evento', I, E, aoj(E, a01), 2).
rel('posse-origem', P, O, pos(O, P), 2).
rel('objeto-origem', Ob, Or, frm(Ob, Or), 1).
rel('objeto1-objeto2', O1, O2, and01(O1, O2), 1).
rel('evento-objeto1', E, O, obj(E, a01), 1).
rel('inativo1-atributo', I, A, aoj(A, a01), 2).
rel('tempo-evento', T, E, tim(E, T), 2).
rel('objeto-objeto', O1, O2, obj(O1, O2), 1).
rel('evento-maneir', E, M, man(E, M), 1).
rel('modo-maneir', Mo, Ma, md(Ma, Mo), 2).
rel('inativo-parceiro', I, P, ptn(I, P), 1).
rel('inativo-razao', I, R, rsn(I, R), 1).
rel('maneir-evento', M, E, man(E, M), 2).
rel('modo-paciente', M, P, md(P, M), 2).
rel('modo-lugar', M, L, md(L, M), 2).
rel('objeto-lugar', O, L, plc(O, L), 1).
rel('agente-modo', A, M, md(A, M), 1).
rel('posse-destino', P, D, pos(D, P), 2).
rel('objeto-destino', O, D, to(O, D), 1).
rel('meta-objeto', M, O, obj(M, O), 1).
rel('evento-duracao', E, D, dur(E, D), 1).
rel('quantidade-maneir', Q, M, qua(M, Q), 2).
rel('paciente-posse', Pa, Po, pos(Pa, Po), 1).
```

```
rel('objeto-maneir', O, M, man(O, M), 1).
rel('atributo-beneficiario', A, B, ben(A, B), 1).
rel('inativo-tempo', I, T, md(I, T), 1).
rel('atributo-proposito', A, P, pur(A, P), 1).
rel('tempo-atributo', T, A, aoj(A, T), 2).
rel('posse-beneficiario', P, B, pos(B, P), 2).
rel('atributo-cena', A, C, scn(A, C), 1).
rel('cena-modo', C, M, md(C, M), 1).
rel('atributo-maneir', A, M, man(A, M), 1).
rel('agente1-agente2', A1, A2, and01(A1, A2), 1).
rel('agente2-modo', A, M, md(A, M), 1).
rel('agente1-evento', A, E, agt(E, a01), 2).
rel('posse-inativo', P, I, pos(I, P), 2).
rel('posse-objeto1', P, O, pos(O, P), 2).
rel('posse-objeto2', P, O, pos(O, P), 2).
rel('modo-cena', M, C, scn(M, C), 1).
rel('posse-agente', P, A, pos(A, P), 2).
rel('inativoE-objeto', I, O, obj(I, O), 1).
rel('inativoE-atributo', I, A, aoj(A, I), 1).
rel('maneir-modo', Ma, Mo, man(Mo, Ma), 2).
rel('modoE-objeto', M, O, obj(M, O), 1).
rel('atributo-modoE', A, M, md(A, M), 1).
rel('evento-destino', E, D, to(E, D), 1).
rel('objeto3-objeto4', O3, O4, or01(O3, O4), 1).
rel('evento-objeto3', E, O, obj(E, o01), 1).
rel('objeto4-modo', O, M, md(O, M), 1).
rel('meta-destino', M, D, to(M, D), 1).
rel('destino-modo', D, M, md(D, M), 1).
rel('posse-lugar', P, L, pos(L, P), 2).
rel('posse-proposito', Po, Pr, pos(Pr, Po), 2).
rel('evento-proposito', E, P, pur(E, P), 1).
rel('evento-propositoE', E, P, pur(E, P), 1).
rel('propositoE-objeto', P, O, obj(P, O), 1).
rel('tempo-modoE', T, M, md(T, M), 1).
rel('atributo-tempo', A, T, tim(A, T), 1).
rel('modo-co_coisa', M, C, cao(M, C), 1).
rel('conteudo-modo', C, M, md(C, M), 1).
rel('agente-conteudo', A, C, cnt(A, C), 1).
rel('evento-beneficiario', E, B, ben(E, B), 1).
rel('tempo1-tempo2', T1, T2, and01(T1, T2), 1).
rel('evento-tempo1', E, T, tim(E, a01), 1).
rel('objeto-objeto1', O1, O2, obj(O1, a01), 1).
rel('modoEm-objeto', M, O, obj(M, O), 1).
rel('meta-modoEm', Me, Mo, md(Me, Mo), 1).
rel('maneir-maneir', Ma1, Ma2, man(Ma2, Ma1), 2).
rel('quantidade-objeto', Q, O, qua(O, Q), 2).
rel('agenteE-maneir', A, M, man(A, M), 1).
rel('agenteE-evento', A, E, agt(E, A), 1).
```

5 Léxico Enriquecido

O Léxico Enriquecido (Specia & Rino, 2002b) é um léxico semântico que armazena informações morfossintáticas, o conceito UNL e informações semânticas sobre as palavras dos córpus base e de teste, as quais são acessadas pelas regras de projeção terminais. Sua implementação é realizada seguindo o formalismo DCG (*Definite Clause Grammar*) (Pereira & Warren, 1980). As entradas ilustradas na seqüência são divididas de acordo com a categoria sintática das palavras e representam tanto as palavras do córpus base quanto as do córpus de teste. A sintaxe geral de cada categoria é indicada em comentário. Os casos de inclusão de novas entradas as para palavras do córpus de testes, em cada categoria, também são indicados. Ao todo, são 732 entradas lexicais distintas.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                Lexico                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ADJETIVOS: adj(forma_canonica) --> [forma_analisada] OU adj(forma_unl) --> [forma_canonica]

adj(can(pessoal)) --> [pessoais].
adj(can(superior)) --> [superiores].
adj(unl(crescent)) --> [crescente].
adj(unl(important)) --> [importante].
adj(unl(inevitable)) --> [inevitável].
adj(unl(personal)) --> [pessoal].
adj(unl(previous)) --> [anterior].
adj(unl(spiritual)) --> [espiritual].
adj(unl(superior)) --> [superior].
adj(unl(versatile)) --> [versátil].
adj(can(crítico)) --> [críticas].
adj(can(delicado)) --> [delicadas].
adj(can(doméstico)) --> [domésticas].
adj(can(longínquo)) --> [longínquas].
adj(can(algum)) --> [alguma].
adj(can(cotidiano)) --> [cotidiana].
adj(can(externo)) --> [externa].
adj(can(gratuito)) --> [gratuita].
adj(can(relativo)) --> [relativa].
adj(can(bem_vindo)) --> [bem_vindos].
adj(can(conservador)) --> [conservadores].
adj(can(maluco)) --> [malucos].
adj(can(novo)) --> [novos].
adj(unl(audacious)) --> [ousado].
adj(unl(conservative)) --> [conservador].
adj(unl(crazy)) --> [maluco].
adj(unl(critical)) --> [crítico].
adj(unl(daily)) --> [diário].
adj(unl(delicate)) --> [delicado].
adj(unl(domestic)) --> [doméstico].
adj(unl(gratuitous)) --> [gratuito].
adj(unl(faraway)) --> [longínquo].
adj(unl(human)) --> [humano].
adj(unl(in_tune)) --> [afinado].
adj(unl(left)) --> [esquerdo].
adj(unl(new)) --> [novo].
adj(unl(outer)) --> [externo].
adj(unl(physical)) --> [físico].
adj(unl(quotidian)) --> [cotidiano].
adj(unl(relative)) --> [relativo].
adj(unl(some)) --> [algum].
adj(unl(that)) --> [nesse].
adj(unl(this)) --> [este].
adj(unl(that)) --> [esse].
adj(unl(those)) --> [aqueles].
adj(unl(welcome)) --> [bem_vindo].
adj(can(próximo)) --> [próximos].
adj(unl(near)) --> [próximo].
adj(unl(beautiful)) --> [lindo].
adj(unl(fair)) --> [justo].
adj(unl(every)) --> [qualquer].
adj(unl(optimum)) --> [ótimo].
adj(unl(nice)) --> [bacana].
adj(unl(present)) --> [presente].
adj(unl(familiar)) --> [familiar].
adj(can(ríspido)) --> [ríspidas].
adj(unl(harsh)) --> [ríspido].
adj(can(bom)) --> [boas].
adj(can(bom)) --> [bons].
adj(unl(good)) --> [bom].
adj(can(apropriado)) --> [apropriadas].
adj(unl(suitable)) --> [apropriado].
adj(can(melhor)) --> [melhores].
adj(unl(better)) --> [melhor].
```

```
adj(unl(indomitable)) --> [indomável].
adj(unl(irrevocable)) --> [irreversível].
adj(can(pouco)) --> [poucos].
adj(unl(few)) --> [pouco].
adj(unl(realistic)) --> [realista].
adj(unl(capable)) --> [capaz].
adj(unl(comfortable)) --> [confortável].
adj(unl(skilful)) --> [hábil].
adj(unl(impatient)) --> [impaciente].
adj(can(suspeito)) --> [suspeitíssimo].
adj(unl(suspect)) --> [suspeito].
adj(unl(half)) --> [meio].
adj(unl(open)) --> [aberto].
adj(can(aberto)) --> [aberta].
adj(can(muito)) --> [muitas].
adj(unl(many)) --> [muito].
adj(unl(limited)) --> [limitado].
adj(can(satisfeito)) --> [satisfeita].
adj(unl(satisfied)) --> [satisfeito].
adj(unl(social)) --> [social].
adj(can(controlado)) --> [controlada].
adj(unl(controlled)) --> [controlado].
adj(unl(tuned)) --> [ligado].
adj(unl(attentive)) --> [atento].
adj(can(dosado)) --> [dosada].
adj(unl(dosed)) --> [dosado].
adj(can(eficiente)) --> [eficientes].
adj(unl(effective)) --> [eficiente].
adj(unl(greek)) --> [grego].
adj(unl(special)) --> [especial].
adj(can(muito)) --> [muitos].
adj(unl(bizarre)) --> [bizarro].
adj(unl(far_away)) --> [longe].
adj(unl(eminent)) --> [eminente].

% entradas adicionais para o corpus de teste

adj(unl(sacred)) --> [sagrado].
adj(unl(brilliant)) --> [radiante].
adj(unl(bad)) --> [ruim].
adj(unl(ease)) --> [fácil].
adj(unl(fundamental)) --> [fundamental].
adj(can(claro)) --> [clara].
adj(unl(clear)) --> [claro].
adj(unl(calm)) --> [tranqüilo].
adj(unl(dramatic)) --> [melodramático].
adj(can(imediato)) --> [imediata].
adj(unl(immediate)) --> [imediato].
adj(can(expressivo)) --> [expressiva].
adj(unl(expressive)) --> [expressivo].
adj(can(criativo)) --> [criativas].
adj(unl(creative)) --> [criativo].
adj(unl(malleable)) --> [maleável].
adj(can(valioso)) --> [valiosa].
adj(unl(valuable)) --> [valioso].
adj(can(favorecido)) --> [favorecida].
adj(unl(favoured)) --> [favorecido].
adj(unl(devoted)) --> [dedicado].
adj(unl(beneficial)) --> [benéfico].
adj(can(gratificante)) --> [gratificantes].
adj(unl(gratifying)) --> [gratificante].
adj(unl(profitable)) --> [proveitoso].
adj(can(agradável)) --> [agradáveis].
adj(unl(pleasant)) --> [agradável].
adj(can(atenuado)) --> [atenuados].
adj(unl(reduced)) --> [atenuado].
adj(can(elevado)) --> [elevada].
adj(unl(elevated)) --> [elevado].
adj(can(próximo)) --> [próximas].
adj(unl(agitated)) --> [agitado].
```

```
adj(can(receptivo)) --> [receptiva].
adj(unl(receptive)) --> [receptivo].
adj(unl(mental)) --> [mental].
adj(can(introspectivo)) --> [introspectiva].
adj(unl(introspective)) --> [introspectivo].
adj(unl(difficult)) --> [difícil].
adj(unl(funny)) --> [divertido].
adj(unl(calm)) --> [calmo].
adj(unl(favourable)) --> [favorável].
adj(can(importante)) --> [importantes].
adj(unl(certain      )) --> [certo].
adj(can(novo)) --> [novas].
adj(unl(charming)) --> [charmoso].
adj(can(bom)) --> [boa].
adj(unl(that)) --> [essa].

% ADVERBIOS: adv(tracos_sintaticos,forma_unl) --> [forma_canonica] OU adv(forma_unl) --> [forma_canonica]

adv(sin(cir_temp),unl(today)) --> [hoje].
adv(sin(cir_temp),unl(tomorrow)) --> [amanhã].
adv(sin(int),unl(more)) --> [mais].
adv(sin(neg),unl(not)) --> [não].
adv(unl(also)) --> [também].
adv(sin(cir_temp),unl(always)) --> [sempre].
adv(sin(cir_temp),unl(time)) --> [vez].
adv(sin(cir_temp),unl(times)) --> [vezes].
adv(sin(cir_mod),unl(only)) --> [só].
adv(sin(cir_temp),unl(saturday)) --> [sábado].
adv(sin(duv),unl(perhaps)) --> [talvez].
adv(sin(int),unl(too_much)) --> [demais].
adv(sin(cir_lug),unl(there)) --> [ali].
adv(sin(int),unl(very)) --> [muito].
adv(sin(cir_temp),unl(before)) --> [antes].
adv(sin(cir_lug),unl(far_away)) --> [longe].

% entradas adicionais para o corpus de teste

adv(sin(cir_mod),unl(cordially)) --> [cordialmente].

% ARTIGOS: art(tracos_sintaticos,forma_canonica) --> [forma_analisada]  OU
% art(tracos_sintaticos,forma_unl) --> [forma_canonica]

art(sin(i),can(um)) --> [uma].
art(sin(de),can(o)) --> [a].
art(sin(de),can(o)) --> [as].
art(sin(de),can(o)) --> [os].
art(sin(de),unl(the)) --> [o].
art(sin(i),unl(a)) --> [um].

% CONJUNCOES: conj(tracos_sintaticos,forma_unl) --> [forma_canonica]

conj(sin(coord,adit),unl(and)) --> [e].
conj(sin(coord,alter),unl(and)) --> [ou].

% PREPOCICOES: prep(forma_canonica) --> [forma_analisada] OU prep(forma_unl) --> [forma_canonica]

prep(can(ao) --> [às].
prep(can(de)) --> [do].
prep(can(no)) --> [nos].
prep(unl(to)) --> [para].
prep(unl(for)) --> [por].
prep(unl(from)) --> [de].
prep(unl(in)) --> [em].
prep(unl(in)) --> [no].
prep(unl(of)) --> [de].
prep(unl(to)) --> [a].
prep(unl(to)) --> [ao].
prep(unl(with)) --> [com].
prep(can(no)) --> [nas].
prep(can(no)) --> [na].
```

```
prep(unl(with)) --> [com].
prep(unl(on)) --> [sobre].
prep(can(de)) --> [da].
prep(can(ao) --> [aos].
prep(can(ao) --> [à].

% PRONOMES: pron(tracos_sintaticos,forma_canonica) --> [forma_analisada] OU
% pron(tracos_sintaticos,forma_unl) --> [forma_canonica]

pron(sin(dem),unl(this),sem([abstrato,inanimado,nao_humano]) --> [isso].
pron(sin(poss),can(você)) --> [seu].
pron(sin(poss),can(você)) --> [sua].
pron(sin(poss),can(você)) --> [suas].
pron(sin(poss),can(você)) --> [seus].
pron(sin(trat),unl(you),sem([concreto,animado,humano])) --> [você].
pron(sin(trat),unl(you),sem([concreto,animado,humano])) --> [vocês].
pron(sin(inde),unl(everything)) --> [tudo].
pron(sin(inde),can(todo)) --> [toda].
pron(sin(inde),unl(all)) --> [todo].
pron(sin(inde),unl(nobody)) --> [ninguém].
pron(sin(ret),unl(they)) --> [eles].
pron(sin(ret),unl(they),sem([concreto,animado,humano])) --> [eles].
pron(sin(inde),unl(something)) --> [algo].
pron(sin(refl),unl(yourself)) --> [se].
pron(sin(inde),unl(many)) --> [muitas].
pron(sin(inde),unl(many)) --> [muitos].

% entradas adicionais para o corpus de teste

pron(sin(inde),unl(nothing)) --> [nada].

% SUBSTANTIVOS: s(tracos_sintaticos,forma_canonica) --> [forma_analisada] OU
% s(tracos_sintaticos,forma_unl,tracos_semanticos) --> [forma_canonica]

s(sin(pl),can(conservador)) --> [conservadores].
s(sin(pl),can(emoção)) --> [emoções].
s(sin(pl),can(cor)) --> [cores].
s(sin(pl),can(hora)) --> [horas].
s(sin(pl),can(mudança)) --> [mudanças].
s(sin(pl),can(perna)) --> [pernas].
s(sin(pl),can(perspectiva)) --> [perspectivas].
s(sin(pl),can(relação)) --> [relações].
s(sin(pl),can(superior)) --> [superiores].
s(sin(pl),can(viagem)) --> [viagens].
s(sin(si),unl(affectionateness),sem([abstrato,inanimado,nao_humano])) --> [afetividade].
s(sin(si),unl(attention),sem([abstrato,inanimado,nao_humano])) --> [atenção].
s(sin(si),unl(blessing),sem([abstrato,inanimado,nao_humano])) --> [bênção].
s(sin(si),unl(change),sem([abstrato,inanimado,nao_humano])) --> [mudança].
s(sin(si),unl(color),sem([abstrato,inanimado,nao_humano])) --> [cor].
s(sin(si),unl(conservative),sem([concreto,animado,humano])) --> [conservador].
s(sin(si),unl(creativity),sem([abstrato,inanimado,nao_humano])) --> [criatividade].
s(sin(si),unl(distraction),sem([abstrato,inanimado,nao_humano])) --> [distração].
s(sin(si),unl(emotion),sem([abstrato,inanimado,nao_humano])) --> [emoção].
s(sin(si),unl(freedom),sem([abstrato,inanimado,nao_humano])) --> [liberdade].
s(sin(si),unl(hour),sem([abstrato,inanimado,nao_humano,tempo])) --> [hora].
s(sin(si),unl(image),sem([abstrato,inanimado,nao_humano])) --> [imagem].
s(sin(si),unl(initiative),sem([abstrato,inanimado,nao_humano])) --> [iniciativa].
s(sin(si),unl(impatience),sem([abstrato,inanimado,nao_humano])) --> [impaciência].
s(sin(si),unl(intuition),sem([abstrato,inanimado,nao_humano])) --> [intuição].
s(sin(si),unl(leg),sem([concreto,animado,humano])) --> [perna].
s(sin(si),unl(life),sem([abstrato,inanimado,nao_humano])) --> [vida].
s(sin(si),unl(mercy),sem([abstrato,inanimado,nao_humano])) --> [misericórdia].
s(sin(si),unl(moon),sem([concreto,animado,nao_humano])) --> [lua].
s(sin(si),unl(need),sem([abstrato,inanimado,nao_humano])) --> [necessidade].
s(sin(si),unl(peace),sem([abstrato,inanimado,nao_humano])) --> [paz].
s(sin(si),unl(perspective),sem([abstrato,inanimado,nao_humano])) --> [perspectiva].
s(sin(si),unl(phase),sem([abstrato,inanimado,nao_humano])) --> [fase].
s(sin(si),unl(pressure),sem([abstrato,inanimado,nao_humano])) --> [pressão].
s(sin(si),unl(relation),sem([abstrato,inanimado,nao_humano])) --> [relação].
s(sin(si),unl(superior),sem([concreto,animado,humano])) --> [superior].
```

```
s(sin(si),unl(piece),sem([abstrato,inanimado,nao_humano])) --> [parte].
s(sin(si),unl(trip),sem([abstrato,inanimado,nao_humano])) --> [viagem].
s(sin(si),unl(truce),sem([abstrato,inanimado,nao_humano])) --> [trégua].
s(sin(pl),can(desacerto)) --> [desacertos].
s(sin(pl),can(estudo)) --> [estudos].
s(sin(pl),can(interesse)) --> [interesses].
s(sin(pl),unl(pisces),sem([abstrato,inanimado,nao_humano,lugar])) --> [peixes].
s(sin(pl),can(plano)) --> [planos].
s(sin(pl),can(relacionamento)) --> [relacionamentos].
s(sin(pl),can(talento)) --> [talentos].
s(sin(si),unl(conflict),sem([abstrato,inanimado,nao_humano])) --> [conflito].
s(sin(si),unl(control),sem([abstrato,inanimado,nao_humano])) --> [controle].
s(sin(si),unl(discomfort),sem([abstrato,inanimado,nao_humano])) --> [desconforto].
s(sin(si),unl(error),sem([abstrato,inanimado,nao_humano])) --> [desacerto].
s(sin(si),unl(foot),sem([concreto,animado,humano])) --> [pé].
s(sin(si),unl(warmth),sem([abstrato,inanimado,nao_humano])) --> [calor].
s(sin(si),unl(instinct),sem([abstrato,inanimado,nao_humano])) --> [instinto].
s(sin(si),unl(interest),sem([abstrato,inanimado,nao_humano])) --> [interesse].
s(sin(si),unl(moment),sem([abstrato,inanimado,nao_humano,tempo])) --> [momento].
s(sin(si),unl(optimism),sem([abstrato,inanimado,nao_humano])) --> [otimismo].
s(sin(si),unl(plan),sem([abstrato,inanimado,nao_humano])) --> [plano].
s(sin(si),unl(point),sem([abstrato,inanimado,nao_humano])) --> [ponto].
s(sin(si),unl(regress),sem([abstrato,inanimado,nao_humano])) --> [retorno].
s(sin(si),unl(relationship),sem([abstrato,inanimado,nao_humano])) --> [relacionamento].
s(sin(si),unl(revolt),sem([abstrato,inanimado,nao_humano])) --> [rebeldia].
s(sin(si),unl(sagittarius),sem([abstrato,inanimado,nao_humano,lugar])) --> [sagitário].
s(sin(si),unl(sector),sem([abstrato,inanimado,nao_humano,lugar])) --> [setor].
s(sin(si),unl(sign),sem([abstrato,inanimado,nao_humano,lugar])) --> [signo].
s(sin(si),unl(size),sem([abstrato,inanimado,nao_humano])) --> [tamanho].
s(sin(si),unl(state),sem([abstrato,inanimado,nao_humano,lugar])) --> [estado].
s(sin(si),unl(study),sem([abstrato,inanimado,nao_humano])) --> [estudo].
s(sin(si),unl(sun),sem([concreto,animado,nao_humano])) --> [sol].
s(sin(si),unl(talent), sem([abstrato,inanimado,nao_humano])) --> [talento].
s(sin(si),unl(that),sem([])) --> [esse].
s(sin(si),unl(that),sem([])) --> [essa].
s(sin(si),unl(this),sem([])) --> [este].
s(sin(si),unl(wardrobe),sem([concreto,inanimado,nao_humano,lugar])) --> [armário].
s(sin(si),unl(wastage),sem([abstrato,inanimado,nao_humano])) --> [desgaste].
s(sin(si),unl(work),sem([abstrato,inanimado,nao_humano])) --> [trabalho].
s(sin(si),unl(pluto),sem([concreto,animado,nao_humano,lugar])) --> [plutão].
s(sin(si),unl(principle),sem([abstrato,inanimado,nao_humano])) --> [princípio].
s(sin(si),unl(weight),sem([abstrato,inanimado,nao_humano])) --> [peso].
s(sin(si),unl(obligation),sem([abstrato,inanimado,nao_humano])) --> [obrigação].
s(sin(si),unl(atmosphere),sem([abstrato,inanimado,nao_humano])) --> [clima].
s(sin(si),unl(appearance),sem([abstrato,inanimado,nao_humano])) --> [visual].
s(sin(pl),can(crítica)) --> [críticas].
s(sin(si),unl(criticism),sem([abstrato,inanimado,nao_humano])) --> [crítica].
s(sin(si),unl(distaste),sem([abstrato,inanimado,nao_humano])) --> [desagrado].
s(sin(si),unl(generosity),sem([abstrato,inanimado,nao_humano])) --> [generosidade].
s(sin(si),unl(shine),sem([abstrato,inanimado,nao_humano])) --> [brilho].
s(sin(si),unl(program),sem([abstrato,inanimado,nao_humano])) --> [programa].
s(sin(si),unl(cheer),sem([abstrato,inanimado,nao_humano])) --> [ânimo].
s(sin(si),unl(day),sem([abstrato,inanimado,nao_humano])) --> [dia].
s(sin(si),unl(stress),sem([abstrato,inanimado,nao_humano])) --> [tensão].
s(sin(pl),can(sombra)) --> [sombras].
s(sin(si),unl(shadow),sem([concreto,inanimado,nao_humano])) --> [sombra].
s(sin(si),unl(saturday),sem([abstrato,inanimado,nao_humano])) --> [sábado].
s(sin(si),unl(agitation),sem([abstrato,inanimado,nao_humano])) --> [agitação].
s(sin(si),unl(exchange),sem([abstrato,inanimado,nao_humano])) --> [troca].
s(sin(pl),can(informação)) --> [informações].
s(sin(si),unl(information),sem([abstrato,inanimado,nao_humano])) --> [informação].
s(sin(si),unl(time),sem([abstrato,inanimado,nao_humano])) --> [tempo].
s(sin(si),unl(meditation),sem([abstrato,inanimado,nao_humano])) --> [meditação].
s(sin(si),unl(effect),sem([abstrato,inanimado,nao_humano])) --> [efeito].
s(sin(si),unl(humour),sem([abstrato,inanimado,nao_humano])) --> [humor].
s(sin(si),unl(irritation),sem([abstrato,inanimado,nao_humano])) --> [irritação].
s(sin(si),unl(variation),sem([abstrato,inanimado,nao_humano])) --> [variação].
s(sin(si),unl(truth),sem([abstrato,inanimado,nao_humano])) --> [verdade].
s(sin(pl),can(palavra)) --> [palavras].
s(sin(si),unl(word),sem([abstrato,inanimado,nao_humano])) --> [palavra].
```

s(sin(pl),can(intenção)) --> [intenções].
s(sin(si),unl(intention),sem([abstrato,inanimado,nao_humano])) --> [intenção].
s(sin(pl),can(intervenção)) --> [intervenções].
s(sin(si),unl(intervention),sem([abstrato,inanimado,nao_humano])) --> [intervenção].
s(sin(si),unl(strength),sem([abstrato,inanimado,nao_humano])) --> [força].
s(sin(si),unl(dream),sem([abstrato,inanimado,nao_humano])) --> [sonho].
s(sin(pl),can(mundo)) --> [mundos].
s(sin(si),unl(world),sem([concreto,animado,nao_humano])) --> [mundo].
s(sin(si),unl(belief),sem([abstrato,inanimado,nao_humano])) --> [crença].
s(sin(si),unl(determination),sem([abstrato,inanimado,nao_humano])) --> [determinação].
s(sin(pl),can(lição)) --> [lições].
s(sin(si),unl(lesson),sem([abstrato,inanimado,nao_humano])) --> [lição].
s(sin(si),unl(vivacity),sem([abstrato,inanimado,nao_humano])) --> [vivência].
s(sin(si),unl(mind),sem([abstrato,inanimado,humano])) --> [mente].
s(sin(si),unl(mars),sem([concreto,animado,nao_humano,lugar])) --> [marte].
s(sin(si),unl(uranus),sem([concreto,animado,nao_humano,lugar])) --> [urano].
s(sin(pl),can(cena)) --> [cenas].
s(sin(si),unl(scene),sem([abstrato,inanimado,nao_humano])) --> [cena].
s(sin(si),unl(drama),sem([abstrato,inanimado,nao_humano])) --> [drama].
s(sin(si),unl(pair),sem([abstrato,inanimado,nao_humano])) --> [casal].
s(sin(si),unl(horizon),sem([abstrato,inanimado,nao_humano,lugar])) --> [horizonte].
s(sin(si),unl(libra),sem([concreto,animado,nao_humano,lugar])) --> [libra].
s(sin(si),unl(look),sem([abstrato,inanimado,nao_humano])) --> [olhar].
s(sin(pl),can(forma)) --> [formas].
s(sin(si),unl(shape),sem([abstrato,inanimado,nao_humano])) --> [forma].
s(sin(si),unl(reflex),sem([abstrato,inanimado,nao_humano])) --> [reflexo].
s(sin(pl),can(objetivo)) --> [objetivos].
s(sin(si),unl(aim),sem([abstrato,inanimado,nao_humano])) --> [objetivo].
s(sin(si),unl(adrenalin),sem([abstrato,inanimado,nao_humano])) --> [adrenalina].
s(sin(si),unl(common),sem([abstrato,inanimado,nao_humano])) --> [comum].
s(sin(si),unl(space),sem([abstrato,inanimado,nao_humano])) --> [espaço].
s(sin(pl),can(preocupação)) --> [preocupações].
s(sin(si),unl(preoccupation),sem([abstrato,inanimado,nao_humano])) --> [preocupação].
s(sin(si),unl(tiredness),sem([abstrato,inanimado,nao_humano])) --> [cansaço].
s(sin(pl),can(sinal)) --> [sinais].
s(sin(si),unl(signal),sem([abstrato,inanimado,nao_humano])) --> [sinal].
s(sin(si),unl(sadness),sem([abstrato,inanimado,nao_humano])) --> [tristeza].
s(sin(si),unl(pressure),sem([abstrato,inanimado,nao_humano])) --> [pressão].
s(sin(si),unl(loneliness),sem([abstrato,inanimado,nao_humano])) --> [solidão].
s(sin(si),unl(desert),sem([abstrato,inanimado,nao_humano])) --> [deserto].
s(sin(si),unl(stage),sem([abstrato,inanimado,nao_humano])) --> [palco].
s(sin(si),unl(term),sem([abstrato,inanimado,nao_humano])) --> [termo].
s(sin(pl),can(vento)) --> [ventos].
s(sin(si),unl(wind),sem([abstrato,animado,nao_humano])) --> [vento].
s(sin(si),unl(environs),sem([abstrato,inanimado,nao_humano,lugar])) --> [volta].
s(sin(si),unl(happiness),sem([abstrato,inanimado,nao_humano])) --> [alegria].
s(sin(si),unl(inovation),sem([abstrato,inanimado,nao_humano])) --> [inovação].
s(sin(pl),can(coisa)) --> [coisas].
s(sin(si),unl(thing),sem([abstrato,inanimado,nao_humano])) --> [coisa].
s(sin(si),unl(rest),sem([abstrato,inanimado,nao_humano])) --> [resto].
s(sin(si),unl(power),sem([abstrato,inanimado,nao_humano])) --> [poder].
s(sin(si),unl(bargain),sem([abstrato,inanimado,nao_humano])) --> [barganha].
s(sin(si),unl(step),sem([abstrato,inanimado,nao_humano])) --> [passo].
s(sin(si),unl(false),sem([abstrato,inanimado,nao_humano])) --> [falso].
s(sin(pl),can(caminho)) --> [caminhos].
s(sin(si),unl(way),sem([concreto,inanimado,nao_humano,lugar])) --> [caminho].
s(sin(pl),can(antena)) --> [antenas].
s(sin(si),unl(antenna),sem([concreto,inanimado,nao_humano])) --> [antena].
s(sin(si),unl(curiosity),sem([abstrato,inanimado,nao_humano])) --> [curiosidade].
s(sin(si),unl(eminence),sem([abstrato,inanimado,nao_humano])) --> [destaque].
s(sin(si),unl(brother),sem([concreto,animado,humano])) --> [irmão].
s(sin(si),unl(taurus),sem([concreto,animado,nao_humano,lugar])) --> [touro].
s(sin(si),unl(instability),sem([abstrato,inanimado,nao_humano])) --> [instabilidade].
s(sin(pl),can(desejo)) --> [desejos].
s(sin(si),unl(desire),sem([abstrato,inanimado,nao_humano])) --> [desejo].
s(sin(si),unl(dissatisfaction),sem([abstrato,inanimado,nao_humano])) --> [insatisfação].
s(sin(pl),can(ouvido)) --> [ouvidos].
s(sin(si),unl(ear),sem([concreto,inanimado,humano])) --> [ouvido].
s(sin(si),unl(chance),sem([abstrato,inanimado,nao_humano])) --> [chance].
s(sin(si),unl(wilfulness),sem([abstrato,inanimado,nao_humano])) --> [teimosia].

```
s(sin(si),unl(suavity),sem([abstrato,inanimado,nao_humano])) --> [suavidade].
s(sin(si),unl(touch),sem([abstrato,inanimado,nao_humano])) --> [toque].
s(sin(si),unl(presence),sem([abstrato,inanimado,nao_humano])) --> [presença].
s(sin(si),unl(unexpected),sem([abstrato,inanimado,nao_humano])) --> [inesperado].
s(sin(si),unl(mercury),sem([concreto,animado,nao_humano,lugar])) --> [mercúrio].
s(sin(si),unl(hermes),sem([concreto,animado,humano])) --> [hermes].
s(sin(pl),can(sonho)) --> [sonhos].
s(sin(pl),can(idéia)) --> [idéias].
s(sin(si),unl(thought),sem([abstrato,inanimado,nao_humano])) --> [idéia].
s(sin(pl),can(colega)) --> [colegas].
s(sin(si),unl(colleague),sem([concreto,animado,humano])) --> [colega].
s(sin(si),unl(quotidian),sem([abstrato,inanimado,nao_humano])) --> [cotidiano].
s(sin(si),unl(gift),sem([abstrato,inanimado,nao_humano])) --> [dom].
s(sin(pl),can(notícia)) --> [notícias].
s(sin(si),unl(news),sem([abstrato,inanimado,nao_humano])) --> [notícia].
s(sin(si),unl(environment),sem([abstrato,inanimado,nao_humano])) --> [ambiente].
s(sin(pl),can(assunto)) --> [assuntos].
s(sin(si),unl(subject),sem([abstrato,inanimado,nao_humano])) --> [assunto].
s(sin(si),unl(load),sem([abstrato,inanimado,nao_humano])) --> [carga].
s(sin(si),unl(help),sem([abstrato,inanimado,nao_humano])) --> [ajuda].
s(sin(si),unl(attitude),sem([abstrato,inanimado,nao_humano])) --> [postura].
s(sin(si),unl(competence),sem([abstrato,inanimado,nao_humano])) --> [competência].
s(sin(si),unl(experience),sem([abstrato,inanimado,nao_humano])) --> [experiência].

% entradas adicionais para o corpus de teste

s(sin(si),unl(head),sem([concreto,inanimado,humano])) --> [cabeça].
s(sin(si),unl(field),sem([abstrato,inanimado,nao_humano])) --> [campo].
s(sin(si),unl(conscience),sem([abstrato,inanimado,nao_humano])) --> [consciência].
s(sin(si),unl(temple),sem([concreto,inanimado,nao_humano])) --> [templo].
s(sin(pl),can(experiência)) --> [experiências].
s(sin(si),unl(expression),sem([abstrato,inanimado,nao_humano])) --> [expressão].
s(sin(pl),can(silêncio)) --> [silêncios].
s(sin(si),unl(silence),sem([abstrato,inanimado,nao_humano])) --> [silêncio].
s(sin(pl),can(período)) --> [períodos].
s(sin(si),unl(period),sem([abstrato,inanimado,nao_humano,tempo])) --> [período].
s(sin(si),unl(inactivity),sem([abstrato,inanimado,nao_humano])) --> [inatividade].
s(sin(si),unl(ignorance),sem([abstrato,inanimado,nao_humano])) --> [ignorância].
s(sin(si),unl(anterior),sem([abstrato,inanimado,nao_humano])) --> [anterior].
s(sin(pl),can(sentimento)) --> [sentimentos].
s(sin(si),unl(feeling),sem([abstrato,inanimado,nao_humano])) --> [sentimento].
s(sin(pl),can(acontecimento)) --> [acontecimentos].
s(sin(si),unl(happening),sem([abstrato,inanimado,nao_humano])) --> [acontecimento].
s(sin(si),unl(reality),sem([abstrato,inanimado,nao_humano])) --> [realidade].
s(sin(si),unl(beauty),sem([abstrato,inanimado,nao_humano])) --> [beleza].
s(sin(pl),can(recurso)) --> [recursos].
s(sin(si),unl(resource),sem([abstrato,inanimado,nao_humano])) --> [recurso].
s(sin(pl),can(associado)) --> [associados].
s(sin(si),unl(partner),sem([abstrato,inanimado,nao_humano])) --> [associado].
s(sin(si),unl(success),sem([abstrato,inanimado,nao_humano])) --> [sucesso].
s(sin(si),unl(focus),sem([abstrato,inanimado,nao_humano])) --> [foco].
s(sin(pl),can(pesquisa)) --> [pesquisas].
s(sin(si),unl(research),sem([abstrato,inanimado,nao_humano])) --> [pesquisa].
s(sin(si),unl(communication),sem([abstrato,inanimado,nao_humano])) --> [comunicação].
s(sin(si),unl(retraction),sem([abstrato,inanimado,nao_humano])) --> [retraimento].
s(sin(si),unl(capability),sem([abstrato,inanimado,nao_humano])) --> [capacidade].
s(sin(si),unl(adaptation),sem([abstrato,inanimado,nao_humano])) --> [adaptação].
s(sin(si),unl(romanticism),sem([abstrato,inanimado,nao_humano])) --> [romantismo].
s(sin(si),unl(receptivity),sem([abstrato,inanimado,nao_humano])) --> [receptividade].
s(sin(si),unl(magnetism),sem([abstrato,inanimado,nao_humano])) --> [magnetismo].
s(sin(pl),can(atividade)) --> [atividades].
s(sin(si),unl(activity),sem([abstrato,inanimado,nao_humano])) --> [atividade].
s(sin(si),unl(nature),sem([abstrato,inanimado,nao_humano])) --> [natureza].
s(sin(pl),can(sugestão)) --> [sugestões].
s(sin(si),unl(suggestion),sem([abstrato,inanimado,nao_humano])) --> [sugestão].
s(sin(si),unl(area),sem([abstrato,inanimado,nao_humano,lugar])) --> [area].
s(sin(pl),can(circunstância)) --> [circunstâncias].
s(sin(si),unl(circumstance),sem([abstrato,inanimado,nao_humano])) --> [circunstância].
s(sin(si),unl(advance),sem([abstrato,inanimado,nao_humano])) --> [avanço].
s(sin(si),unl(profession),sem([abstrato,inanimado,nao_humano])) --> [profissão].
```

```
s(sin(si),unl(reputation),sem([abstrato,inanimado,nao_humano])) --> [reputação].
s(sin(si),unl(role),sem([abstrato,inanimado,nao_humano])) --> [papel].
s(sin(si),unl(creativeness),sem([abstrato,inanimado,nao_humano])) --> [criatividade].
s(sin(si),unl(weapon),sem([concreto,inanimado,nao_humano])) --> [arma].
s(sin(pl),can(cobrança)) --> [cobranças].
s(sin(si),unl(collection),sem([abstrato,inanimado,nao_humano])) --> [cobrança].
s(sin(si),can(amado)) --> [amada].
s(sin(si),unl(beloved),sem([concreto,animado,humano])) --> [amado].
s(sin(si),unl(seduction),sem([abstrato,inanimado,nao_humano])) --> [sedução].
s(sin(si),unl(courage),sem([abstrato,inanimado,nao_humano])) --> [coragem].
s(sin(pl),can(oscilação)) --> [oscilações].
s(sin(si),unl(oscillation),sem([abstrato,inanimado,nao_humano])) --> [oscilação].
s(sin(pl),can(contato)) --> [contatos].
s(sin(si),unl(contact),sem([abstrato,inanimado,nao_humano])) --> [contato].
s(sin(pl),can(momento)) --> [momentos].
s(sin(si),unl(advice),sem([abstrato,inanimado,nao_humano])) --> [conselho].
s(sin(pl),can(capacidade)) --> [capacidades].
s(sin(si),unl(opportunity),sem([abstrato,inanimado,nao_humano])) --> [oportunidade].
s(sin(si),unl(jealousy),sem([abstrato,inanimado,nao_humano])) --> [ciúme].
s(sin(si),unl(possession),sem([abstrato,inanimado,nao_humano])) --> [possessão].
s(sin(pl),unl(people),sem([concreto,animado,humano])) --> [pessoas].
s(sin(si),unl(outline),sem([abstrato,inanimado,nao_humano])) --> [esboço].
s(sin(si),unl(health),sem([abstrato,inanimado,nao_humano])) --> [saúde].
s(sin(si),unl(weekend),sem([abstrato,inanimado,nao_humano])) --> [fim_de_semana].
s(sin(pl),can(clamor)) --> [clamores].
s(sin(si),unl(clamour),sem([abstrato,inanimado,nao_humano])) --> [clamor].
s(sin(si),unl(news),sem([abstrato,inanimado,nao_humano])) --> [novidade].
s(sin(si),unl(passion),sem([abstrato,inanimado,nao_humano])) --> [paixão].
s(sin(pl),can(conflito)) --> [conflitos].
s(sin(pl),can(manifestação)) --> [manifestações].
s(sin(si),unl(manifestation),sem([abstrato,inanimado,nao_humano])) --> [manifestação].
s(sin(si),unl(clarity),sem([abstrato,inanimado,nao_humano])) --> [clareza].
s(sin(si),unl(friday),sem([abstrato,inanimado,nao_humano])) --> [sexta].
s(sin(pl),can(pergunta)) --> [perguntas].
s(sin(si),unl(question),sem([abstrato,inanimado,nao_humano])) --> [pergunta].
s(sin(pl),can(familiar)) --> [familiares].
s(sin(si),unl(relative),sem([concreto,animado,humano])) --> [familiar].
s(sin(si),unl(loose_control),sem([abstrato,inanimado,nao_humano])) --> [descontrole].
s(sin(pl),can(responsabilidade)) --> [responsabilidades].
s(sin(si),unl(responsibility),sem([abstrato,inanimado,nao_humano])) --> [responsabilidade].
s(sin(pl),can(alegria)) --> [alegrias].
s(sin(pl),can(oportunidade)) --> [oportunidades].
s(sin(si),unl(area),sem([abstrato,inanimado,nao_humano,lugar])) --> [área].
s(sin(si),unl(profile),sem([abstrato,inanimado,nao_humano])) --> [perfil].

% VERBOS: v(tracos_sintaticos,forma_canonica) --> [forma_analisada] OU
%  v(tracos_sintaticos,forma_unl,classe,estrutura_argumentos) --> [forma_canonica]

v(sin(fut_pres),can(acontecer)) --> [acontecerá].
v(sin(fut_pres),can(estar)) --> [estará].
v(sin(fut_pres),can(estar)) --> [estarão].
v(sin(fut_pres),can(exigir)) --> [exigirão].
v(sin(fut_pres),can(iluminar)) --> [iluminará].
v(sin(fut_pres),can(mudar)) --> [mudará].
v(sin(fut_pres),can(passar)) --> [passarão].
v(sin(fut_pres),can(ser)) --> [será].
v(sin(fut_pres),can(tornar)) --> [tornará].
v(sin(imper_afirm),can(apostar)) --> [aposte].
v(sin(imper_afirm),can(considerar)) --> [considere].
v(sin(imper_afirm),can(esperar)) --> [espere].
v(sin(imper_afirm),can(exibir)) --> [exiba].
v(sin(imper_afirm),can(ser)) --> [seja].
v(sin(imper_afirm),can(tirar)) --> [tire].
v(sin(imper_afirm),can(começar)) --> [comece].
v(sin(inf_pess,aux),unl(can),cl(modal),rest([])) --> [poder].
v(sin(inf_pess,bi),unl(take),cl(acao_proc),rest([suj([animado]),comp([lugar]),obj([abstrato])])) --> [tirar].
v(sin(inf_pess,int),unl(change),cl(processo),rest([suj([inanimado])])) --> [mudar].
v(sin(inf_pess,int),unl(exist),cl(estado),rest([suj([inanimado])])) --> [existir].
v(sin(inf_pess,int),unl(happen),cl(processo),rest([suj([abstrato])])) --> [acontecer].
v(sin(inf_pess,int),unl(occur),cl(processo),rest([suj([])])) --> [ocorrer].
```

v(sin(inf_pess,int),unl(pass),cl(processo),rest([suj([tempo])])) --> [passar].
v(sin(inf_pess,int),unl(dawn),cl(processo),rest([suj([])])) --> [despontar].
v(sin(inf_pess,lig),unl(be),cl(estado),rest([suj([]),pred([])])) --> [estar].
v(sin(inf_pess,lig),unl(be),cl(estado),rest([suj([]),pred([])])) --> [ser].
v(sin(inf_pess,td),unl(become),cl(processo),rest([suj([]),pred([])])) --> [tornar].
v(sin(inf_pess,td),unl(consider),cl(acao),rest([suj([animado]),obj([abstrato])])) --> [considerar].
v(sin(inf_pess,td),unl(exist),cl(estado),rest([obj([])])) --> [haver].
v(sin(inf_pess,td),unl(hope),cl(estado),rest([suj([humano]),obj([abstrato])])) --> [esperar].
v(sin(inf_pess,td),unl(illuminate),cl(acao_proc),rest([suj([concreto,animado]),obj([abstrato])])) --> [iluminar].
v(sin(inf_pess,td),unl(know),cl(estado),rest([suj([animado]),obj([nao_humano])])) --> [saber].
v(sin(inf_pess,td),unl(promise),cl(estado),rest([suj([inanimado]),obj([inanimado])])) --> [prometer].
v(sin(inf_pess,td),unl(require),cl(estado),rest([suj([]),obj([inanimado])])) --> [exigir].
v(sin(inf_pess,td),unl(exhibit),cl(acao_proc),rest([suj([animado]),obj([])])) --> [exibir].
v(sin(inf_pess,td),unl(start),cl(acao_proc),rest([suj([animado]),obj([abstrato])])) --> [começar].
v(sin(inf_pess,td),unl(suggest),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [sugerir].
v(sin(inf_pess,td),unl(transit),cl(acao),rest([suj([animado]),obj([lugar])])) --> [transitar].
v(sin(inf_pess,ti),unl(bet),cl(acao),rest([suj([animado]),obj([])])) --> [apostar].
v(sin(inf_pess,ti),unl(go_in),cl(acao),rest([suj([animado]),obj([lugar])])) --> [ingressar].
v(sin(pres),can(despontar)) --> [despontam].
v(sin(pres),can(estar)) --> [estão].
v(sin(pres),can(existir)) --> [existem].
v(sin(pres),can(haver)) --> [há].
v(sin(pres),can(iluminar)) --> [ilumina].
v(sin(pres),can(ocorrer)) --> [ocorre].
v(sin(pres),can(poder)) --> [podem].
v(sin(pres),can(prometer)) --> [promete].
v(sin(pres),can(saber)) --> [sabe].
v(sin(pres),can(ser)) --> [é].
v(sin(pres),can(ser)) --> [são].
v(sin(pres),can(sugerir)) --> [sugere].
v(sin(pres),can(transitar)) --> [transita].
v(sin(pret_perf),can(ingressar)) --> [ingressou].
v(sin(pres),can(ir)) --> [vai].
v(sin(inf_pess,aux),unl(will),cl(modal),rest([])) --> [ir].
v(sin(inf_pess,ti),unl(give_way),cl(processo),rest([suj([humano]),obj([])])) --> [ceder].
v(sin(pres),can(chegar)) --> [chega].
v(sin(inf_pess,ti),unl(suffice),cl(estado),rest([obj([abstrato])])) --> [chegar].
v(sin(imper_afirm),can(cuidar)) --> [cuide].
v(sin(inf_pess,ti),unl(care),cl(acao),rest([suj([]),obj([])])) --> [cuidar].
v(sin(imper_afirm),can(desdenhar)) --> [desdenhe].
v(sin(inf_pess,td),unl(despise),cl(acao_proc),rest([suj([]),obj([])])) --> [desdenhar].
v(sin(pres),can(expressar)) --> [expressa].
v(sin(inf_pess,td),unl(express),cl(acao),rest([suj([]),obj([abstrato])])) --> [expressar].
v(sin(fut_pres),can(aparecer)) --> [aparecerá].
v(sin(inf_pess,int),unl(appear),cl(processo),rest([suj([abstrato])])) --> [aparecer].
v(sin(inf_pess,td),unl(heat),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [esquentar].
v(sin(fut_pres),can(ter)) --> [terá].
v(sin(inf_pess,td),unl(have),cl(estado),rest([suj([]),obj([abstrato])])) --> [ter].
v(sin(pres),can(diminuir)) --> [diminui].
v(sin(inf_pess,td),unl(decrease),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [diminuir].
v(sin(pres),can(espantar)) --> [espanta].
v(sin(inf_pess,td),unl(dissipate),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [espantar].
v(sin(pres),can(favorecer)) --> [favorece].
v(sin(inf_pess,td),unl(favour),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [favorecer].
v(sin(fut_pres),can(fazer)) --> [fará].
v(sin(inf_pess,td),unl(make),cl(acao),rest([suj([]),obj([abstrato])])) --> [fazer].
v(sin(pres_subj),can(precisar)) --> [precise].
v(sin(fut_pres),can(precisar)) --> [precisará].
v(sin(inf_pess,aux),unl(need),cl(modal),rest([])) --> [precisar].
v(sin(inf_pess,td),unl(damage),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [estragar].
v(sin(pres),can(acabar)) --> [acabam].
v(sin(inf_pess,ti),unl(destroy),cl(acao_proc),rest([suj([]),obj([])])) --> [acabar].
v(sin(imper_afirm),can(fazer)) --> [faça].
v(sin(inf_pess,td),unl(make),cl(acao_proc),rest([suj([humano]),obj([abstrato])])) --> [fazer].
v(sin(imper_afirm),can(manter)) --> [mantenha].
v(sin(inf_pess,td),unl(keep),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [manter].
v(sin(imper_afirm),can(continuar)) --> [continue].
v(sin(inf_pess,lig),unl(continue),cl(estado),rest([suj([]),pred([])])) --> [continuar].
v(sin(inf_pess,td),unl(rob),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [roubar].
v(sin(inf_pess,td),unl(give),cl(acao_proc),rest([suj([animado]),obj([abstrato])])) --> [dar].

```
v(sin(inf_pess,td),unl(use),cl(acao),rest([suj([]),obj([inanimado])])) --> [usar].
v(sin(pres),can(atrapalhar)) --> [atrapalham].
v(sin(inf_pess,td),unl(perturb),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [atrapalhar].
v(sin(inf_pess,ti),unl(be),cl(estado),rest([suj([]),pred([lugar])])) --> [estar].
v(sin(pres),can(atravessar)) --> [atravessa].
v(sin(inf_pess,td),unl(cross),cl(acao),rest([suj([]),obj([lugar])])) --> [atravessar].
v(sin(pres),can(refletir)) --> [reflete].
v(sin(inf_pess,td),unl(reflect),cl(acao_proc),rest([suj([inanimado]),obj([concreto])])) --> [refletir].
v(sin(pres),can(oscilar)) --> [oscilam].
v(sin(inf_pess,int),unl(oscillate),cl(processo),rest([suj([])])) --> [oscilar].
v(sin(imper_afirm),can(ver)) --> [veja].
v(sin(pres),can(ver)) --> [vê].
v(sin(inf_pess,td),unl(see),cl(acao),rest([suj([]),obj([abstrato])])) --> [ver].
v(sin(inf_pess,td),unl(determine),cl(acao_proc),rest([suj([]),obj([])])) --> [fixar].
v(sin(fut_pret),can(ser)) --> [seria].
v(sin(fut_pret),can(aumentar)) --> [aumentaria].
v(sin(inf_pess,td),unl(increase),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [aumentar].
v(sin(inf_pess,int),unl(intend),cl(estado),rest([suj([])])) --> [pretender].
v(sin(imper_afirm),can(divergir)) --> [divirja].
v(sin(inf_pess,ti),unl(diverge),cl(estado),rest([suj([]),obj([abstrato])])) --> [divergir].
v(sin(inf_pess,td),unl(conquer),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [conquistar].
v(sin(imper_afirm),can(deixar)) --> [deixe].
v(sin(inf_pess,td),unl(leave),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [deixar].
v(sin(imper_afirm),can(desprezar)) --> [despreze].
v(sin(inf_pess,td),unl(despise),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [desprezar].
v(sin(pret_perf),can(aumentar)) --> [aumentou].
v(sin(inf_pess,int),unl(rise),cl(processo),rest([suj([])])) --> [aumentar].
v(sin(pret_perf),can(deixar)) --> [deixou].
v(sin(inf_pess,td),unl(let),cl(acao_proc),rest([suj([]),obj([animado]),comp([])])) --> [deixar].
v(sin(inf_pess,bi),unl(push),cl(acao_proc),rest([suj([]),obj([]),comp([])])) --> [empurrar].
v(sin(inf_pess,td),unl(choose),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [escolher].
v(sin(pres),can(estar)) --> [estão].
v(sin(inf_pess,aux),unl(be),cl(modal),rest([])) --> [estar].
v(sin(gerun),can(soprar)) --> [soprando].
v(sin(inf_pess,int),unl(blow),cl(processo),rest([suj([])])) --> [soprar].
v(sin(pres),can(poder)) --> [pode].
v(sin(pres),can(precisar)) --> [precisa].
v(sin(inf_pess,ti),unl(need),cl(estado),rest([suj([]),obj([])])) --> [precisar].
v(sin(imper_afirm),can(dedicar)) --> [dedique].
v(sin(inf_pess,bi),unl(dedicate),cl(acao_proc),rest([suj([]),obj([]),comp([])])) --> [dedicar].
v(sin(inf_pess,ti),unl(treat),cl(acao),rest([suj([]),obj([])])) --> [tratar].
v(sin(pres),can(importar)) --> [importa].
v(sin(inf_pess,int),unl(matter),cl(estado),rest([suj([])])) --> [importar].
v(sin(pres),can(estar)) --> [está].
v(sin(pres),can(aumentar)) --> [aumenta].
v(sin(inf_pess,td),unl(give),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [dar].
v(sin(pres),can(captar)) --> [captam].
v(sin(inf_pess,td),unl(catch),cl(acao_proc),rest([suj([]),obj([])])) --> [captar].
v(sin(inf_pess,td),unl(adopt),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [adotar].
v(sin(imper_afirm),can(refletir)) --> [reflita].
v(sin(inf_pess,int),unl(think),cl(acao),rest([suj([])])) --> [refletir].
v(sin(inf_pess,int),unl(speak),cl(acao),rest([suj([humano])])) --> [falar].
v(sin(imper_afirm),can(espreitar)) --> [espreite].
v(sin(inf_pess,td),unl(peep),cl(acao),rest([suj([]),obj([concreto])])) --> [espreitar].
v(sin(imper_afirm),can(ficar)) --> [fique].
v(sin(inf_pess,lig),unl(stay),cl(estado),rest([suj([]),pred([])])) --> [ficar].
v(sin(inf_pess,td),unl(bring),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [trazer].
v(sin(fut_pres),can(ser)) --> [serão].
v(sin(pres),can(trabalhar)) --> [trabalha].
v(sin(inf_pess,int),unl(work),cl(acao),rest([suj([])])) --> [trabalhar].
v(sin(imper_afirm),can(anotar)) --> [anote].
v(sin(inf_pess,td),unl(register),cl(acao_proc),rest([suj([]),obj([])])) --> [anotar].
v(sin(imper_afirm),can(caprichar)) --> [capriche].
v(sin(inf_pess,ti),unl(excel),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [caprichar].
v(sin(inf_pess,td),unl(confuse),cl(acao_proc),rest([suj([]),obj([])])) --> [confundir].
v(sin(imper_afirm),can(confiar)) --> [confie].
v(sin(inf_pess,ti),unl(trust),cl(estado),rest([suj([]),obj([])])) --> [confiar].
v(sin(fut_pres),can(esperar)) --> [esperará].

% entradas adicionais para o corpus de teste
```

v(sin(imper_afirm),can(levantar)) --> [levante].
v(sin(inf_pess,td),unl(raise),cl(acao_proc),rest([suj([]),obj([concreto])])) --> [levantar].
v(sin(pres),can(conhecer)) --> [conhece].
v(sin(inf_pess,td),unl(know),cl(estado),rest([suj([humano]),obj([nao_humano])])) --> [conhecer].
v(sin(pres),can(crescer)) --> [crescem].
v(sin(inf_pess,int),unl(grow),cl(processo),rest([suj([abstrato])])) --> [crescer].
v(sin(imper_afirm),can(subestimar)) --> [subestime].
v(sin(inf_pess,td),unl(understimate),cl(acao),rest([suj([]),obj([abstrato])])) --> [subestimar].
v(sin(pres),can(perpetuar)) --> [perpetua].
v(sin(inf_pess,td),unl(perpetuate),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [perpetuar].
v(sin(inf_pess,td),unl(destroy),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [destruir].
v(sin(inf_pess,td),unl(hide),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [ocultar].
v(sin(imper_afirm),can(encarar)) --> [encare].
v(sin(inf_pess,td),unl(face),cl(acao),rest([suj([]),obj([abstrato])])) --> [encarar].
v(sin(fut_pres),can(transformar)) --> [transformará].
v(sin(inf_pess,td),unl(transform),cl(acao_proc),rest([suj([]),obj([])])) --> [transformar].
v(sin(imper_afirm),can(atrair)) --> [atraia].
v(sin(inf_pess,td),unl(attract),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [atrair].
v(sin(inf_pess,td),unl(change),cl(acao),rest([suj([]),obj([abstrato])])) --> [mudar].
v(sin(imper_afirm),can(aproveitar)) --> [aproveite].
v(sin(inf_pess,td),unl(utilize),cl(acao),rest([suj([]),obj([abstrato])])) --> [aproveitar].
v(sin(imper_afirm),can(realizar)) --> [realize].
v(sin(inf_pess,td),unl(accomplish),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [realizar].
v(sin(imper_afirm),can(controlar)) --> [controle].
v(sin(inf_pess,td),unl(control),cl(acao_proc),rest([suj([]),obj([])])) --> [controlar].
v(sin(imper_afirm),can(superar)) --> [supere].
v(sin(inf_pess,td),unl(overcome),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [superar].
v(sin(pres),can(encontrar)) --> [encontra].
v(sin(inf_pess,td),unl(find),cl(processo),rest([suj([]),obj([abstrato])])) --> [encontrar].
v(sin(pres),can(acentuar)) --> [acentua].
v(sin(inf_pess,td),unl(accentuate),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [acentuar].
v(sin(imper_afirm),can(privilegiar)) --> [privilegie].
v(sin(inf_pess,td),unl(privilege),cl(acao_proc),rest([suj([]),obj([inanimado])])) --> [privilegiar].
v(sin(imper_afirm),can(buscar)) --> [busque].
v(sin(inf_pess,td),unl(search),cl(acao),rest([suj([]),obj([abstrato])])) --> [buscar].
v(sin(imper_afirm),can(dar)) --> [dê].
v(sin(inf_pess,td),unl(give),cl(acao),rest([suj([]),obj([abstrato])])) --> [dar].
v(sin(pres),can(favorecer)) --> [favorecem].
v(sin(pres),can(jogar)) --> [joga].
v(sin(inf_pess,td),unl(play),cl(acao),rest([suj([]),obj([abstrato])])) --> [jogar].
v(sin(pres),can(facilitar)) --> [facilitam].
v(sin(inf_pess,td),unl(facilitate),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [facilitar].
v(sin(imper_afirm),can(evitar)) --> [evite].
v(sin(inf_pess,td),unl(avoid),cl(acao),rest([suj([]),obj([])])) --> [evitar].
v(sin(imper_afirm),can(exercer)) --> [exerça].
v(sin(inf_pess,td),unl(exert),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [exercer].
v(sin(inf_pess,td),unl(heat),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [aquecer].
v(sin(imper_afirm),can(usar)) --> [use].
v(sin(inf_pess,td),unl(damage),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [prejudicar].
v(sin(fut_pres),can(poder)) --> [poderá].
v(sin(inf_pess,td),unl(enjoy),cl(processo),rest([suj([]),obj([])])) --> [curtir].
v(sin(fut_pres),can(poder)) --> [poderão].
v(sin(imper_afirm),can(desperdiçar)) --> [desperdice].
v(sin(inf_pess,td),unl(waste),cl(acao_proc),rest([suj([]),obj([])])) --> [desperdiçar].
v(sin(fut_pres),can(apontar)) --> [apontará].
v(sin(inf_pess,td),unl(point),cl(acao),rest([suj([]),obj([])])) --> [apontar].
v(sin(fut_pres),can(ver)) --> [verá].
v(sin(imper_afirm),can(estar)) --> [esteja].
v(sin(inf_pess,int),unl(travel),cl(acao),rest([suj([])])) --> [viajar].
v(sin(fut_pres),can(favorecer)) --> [favorecerá].
v(sin(fut_pret),can(poder)) --> [poderiam].
v(sin(inf_pess,td),unl(cause),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [gerar].
v(sin(fut_pres),can(partir)) --> [partirá].
v(sin(inf_pess,int),unl(depart),cl(acao),rest([suj([animado])])) --> [partir].
v(sin(fut_pres),can(haver)) --> [haverá].
v(sin(imper_afirm),can(acreditar)) --> [acredite].
v(sin(inf_pess,ti),unl(believe),cl(estado),rest([suj([]),obj([])])) --> [acreditar].
v(sin(imper_afirm),can(responder)) --> [responda].
v(sin(inf_pess,td),unl(answer),cl(acao),rest([suj([humano]),obj([])])) --> [responder].

```
v(sin(inf_pess,int),unl(arise),cl(processo),rest([suj([abstrato])])) --> [chegar].
v(sin(fut_pres),can(mostrar)) --> [mostrará].
v(sin(inf_pess,td),unl(show),cl(acao_proc),rest([suj([]),obj([abstrato])])) --> [mostrar].
v(sin(pres),can(surgir)) --> [surgem].
v(sin(inf_pess,int),unl(arise),cl(processo),rest([suj([])])) --> [surgir].
v(sin(imper_afirm),can(mexer)) --> [mexa].
v(sin(inf_pess,ti),unl(touch),cl(acao),rest([suj([]),obj([abstrato])])) --> [mexer].
v(sin(pres),can(ter)) --> [tem].
```

6 Procedimentos auxiliares

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%          Procedimentos auxiliares para recupera palavras das sentenças de entrada          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

recupera_sentenca(ES,Sent) :-
        sentenca(ES,S),
        flatten(S,S1),
        stringlist_concat(S1, $ $, Sent).

sentenca(ES,Sent) :-
        atomic(ES),
        concatena([ES],[],Sent).

sentenca(ES,Sent) :-
        ES =.. L,
        concatena([Arg1],[Resto],L),
        isvazia(Resto),
        concatena([Arg1],[],Sent).

sentenca(ES,Sent) :-
        ES =.. L,
        concatena([Arg1],[Resto],L),
        not isvazia(Resto),
        not islist(Resto),
        sentenca(Resto,Sent).

sentenca(ES,Sent) :-
        ES =.. L,
        concatena([Arg1],[Resto],L),
        not isvazia(Resto),
        islist(Resto),
        lista(Resto,ListaSent),
        concatena(ListaSent,[],Sent).

lista(L,ListaSent) :-
        remove(Arg1,L,Resto),
        sentenca(Arg1,L1),
        lista(Resto,L2),
        concatena([L1],[L2],ListaSent).

lista([],[]) :- !.


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                Procedimentos auxiliares para as regras de projeção                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

:-op(800,xfy,:).

isvazia([]).

concatena([], X, X).
concatena([A|X], Y, [A|Z]) :- concatena(X,Y,Z).

flatten([], []).
flatten( [X|T], [X|T2] ) :-    var(X), !,
```

```
                                flatten(T, T2).

flatten([ [] | T], T2) :- !, flatten(T, T2).

flatten([ [H|T] | T2], [H1|T3]) :-      flatten([H|T], [H1|T1]), !,
                                        flatten([T1|T2], T3).

flatten([H|T], [H|T2]) :- flatten(T, T2).

remove(Elem,[Elem|Cauda],Cauda).

remove(Elem,[Elem1|Cauda],[Elem1|Cauda1]) :- remove(Elem,Cauda,Cauda1).

subconjunto([],[]).

subconjunto([Prim|Resto],[Prim|Subconj]) :- subconjunto(Resto,Subconj).

subconjunto([Prim|Resto],Subconj) :- subconjunto(Resto,Subconj).



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                Procedimentos auxiliares para os templates de relacionamento                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% chamada dos templates, pelas regras de projeção

gera(L,Resultado) :- simplifica(L, _, [], Resultado).

% programas auxiliares

simplifica(X, Y, Rs1, Rs2) :-
                functor(X, ':', 2),
                !,
                X = Y,
                Rs1 = Rs2.

simplifica([X], Y, Rs1, Rs2) :-
                !,
                simplifica(X, Y, Rs1, Rs2).

simplifica(X, Y, Rs1, Rs3) :-
                simplificaL(X, SimpX, Rs1, Rs2),
                resta1(SimpX, Y, Rs2, Rs3).

simplificaL([], [], Rs1, Rs1) :- !.

simplificaL([X| Xs], [SimpX| SimpXs], Rs1, Rs3) :-
                simplifica(X, SimpX, Rs1, Rs2),
                simplificaL(Xs, SimpXs, Rs2, Rs3).

resta1([X], Y, Rs1, Rs2) :-
                !,
                X = Y,
                Rs1 = Rs2.

resta1([X1| Xs1], Y, Rs1, Rs2) :-
                pega1(X2, Xs1, Xs2),
                arg(1, X1, P1),
                arg(1, X2, P2),
                relTag(P1, P2, RelTag),
                rel1(RelTag, X1, X2, RNova, XFica),
                resta1([XFica| Xs2], Y, [RNova| Rs1], Rs2).

relTag(P1, P2, Tag) :- atomlist_concat([P1, '-', P2], Tag).

rel1(RelTag, O1, O2, UNLRel, OFica) :-
                rel(RelTag, Arg1, Arg2, UNLRel, IFica),
                ofica(IFica, O1, O2, OFica),
```

```
                    pegaArgs(O1, O2, _:Arg1, _:Arg2).
pegaArgs(O1, O2, O1, O2).
ofica( 1, O1, _, O1) :- !.
ofica( _, _, O2, O2).
pega1(X, [X| Xs], Xs).
pega1(X, [Y| Xs], [Y| Ys]) :- pega1(X, Xs, Ys).
```

Referências Bibliográficas

Martins, R.T.; Hasegawa, R.; Nunes, M.G.V. (2002). *Curupira: um parser funcional para o português*. Série de Relatórios Técnicos do NILC, NILC-TR-02-26. São Carlos, Dezembro, 43p.

Pereira, F.C.N.; Warren, D.H.D. (1980). Definite Clause Grammars for Language Analysis – A Survey of the Formalism and a Comparison with Augmented Transition Networks. *Artificial Intelligence*, 13, pp. 231-278.

Specia, L.; Rino, L.H.M. (2002a). *ConPor: um gerador de estruturas conceituais UNL*. Série de Relatórios Técnicos do NILC, NILC-TR-02-15. São Carlos, Novembro, 40p.

Specia, L.; Rino, L.H.M. (2002b). *O desenvolvimento de um léxico para a geração de estruturas conceituais UNL*. Série de Relatórios Técnicos do NILC, NILC-TR-02-14. São Carlos, Setembro, 25p.

Specia, L.; Rino, L.H.M. (2003). *A generalização do sistema ConPor*. Série de Relatórios Técnicos do NILC, NILC-TR-03-01. São Carlos, Janeiro, 34p.

UNL (2001). *The Universal Networking Language (UNL) Specifications*. UNU/IAS/UNL Center, Tokyo. Disponível em http://www.unl.ias.unu.edu.