3 Sintaxe e semântica

3.1 Introdução

A lógica temporal é um dos formalismos vulgarmente aceites para especificação e verificação de sistemas concorrentes e reactivos. Assim, a lógica apresentada em [Sernadas et al 92e] e refinada em [Sernadas e Sernadas 94] para especificação e verificação de sistema concorrentes, orientados a objectos (OSL - *Object Specification Logic*) apresenta-se como um bom candidato para tentar definir uma axiomatização para o fragmento da linguagem GNOME considerado.

A noção de especialização surge em OSL através de uma ordenação ao nível das classes. Recorde-se o exemplo apresentado no capítulo anterior. Para esssa especificação, diz-se que a classe *Drelógio* é uma subclasse de *Relógio*, ou então, que a classe *Relógio* é uma superclasse de *Drelógio*.

Existe também em OSL a noção de classe agregação. Por exemplo, e ainda relativamente ao exemplo anterior, *Relógio⊗ Utilizador* é uma classe agregação, cujas instâncias são compostas por pares de *relógios* e *utilizadores*. É nestas classes que se descreve a interacção entre *utilizadores* e *relógios*.

Para definir uma axiomatização para a linguagem GNOME é, no entanto, necessário estender a noção de assinatura de classe, fazendo uma distinção explícita entre acções (de iniciativa interna) e serviços (de iniciativa externa) . Esta modificação leva a algumas alterações ao nível da semântica, passando a ser necessário, em

particular, encontrar um mecanismo que, de alguma forma, represente o *ambiente*. É neste contexto que surge a *Lógica de Certificação GNOME* (GCL)[1].

Este capítulo divide-se em duas partes. Na primeira parte apresenta-se a sintaxe e semântica da *Lógica de Certificação GNOME*. Ainda na primeira parte, introduz-se uma instituição para esta lógica baseada numa semântica de ciclos de vida. Na segunda parte apresenta-se uma axiomatização sobre GCL para o fragmento da linguagem GNOME considerado. Esta axiomatização define uma semântica formal para a linguagem.

3.2 Sintaxe

Nesta secção define-se a sintaxe da GCL, começando pela noção de assinatura de classe e seguindo-se as noções de termo e fórmula da lógica.

3.2.1 Assinatura de classe

De agora em diante supõe-se definida uma assinatura de tipos de dados $\Sigma_{TD}=<G_{TD},OP>$ ([Ehrig e Mahr 85]). Em particular, supõe-se que nat,bool,phs $\in G_{TD}$, $OP_{\varepsilon,bool}=\{true,false\}$ e $OP_{\varepsilon,phs}=\{PN,AN,PM\}$. [2]

Tal como se referiu atrás se referiu, e tendo em vista o problema da especialização, é necessário estabelecer uma ordenação sobre as classes.

Definição 3.1 *Rede de herança*

Uma *rede de herança* é uma ordem parcial finita $N=<CP,\leq>$. No caso da ordem parcial ser linear à direita[3], a rede de herança diz-se uma *rede de herança linear*. □

Sobre uma rede de herança linear define-se, para cada $c \in CP$, o elemento $c^{\top}=máx\{c': c\leq c'\}$, que se designa por *classe topo* de c, e o conjunto $CP^{\top}=\{c^{\top}: c\in CP\}$ de todas as *classes topo* (note-se que o facto de a ordem parcial ser linear à direita e finita garante que $c^{\top}$ está sempre bem definida).

---

1 Do inglês, *GNOME Certification Logic*.

2 O género *phs* representa as três possíveis fases de um objecto (*phase status*): *pre-natal* (PN); animação (AN); *post-mortem* (PM).

3 Uma ordem parcial $(N,\leq)$ diz-se *linear à direita* sse $x\leq y_1$ e $x\leq y_2$ implica $y_1\leq y_2$ ou $y_2\leq y_1$, quaisquer que sejam $x,y_1,y_2\in N$.

Assume-se, de agora em diante, que qualquer rede de herança considerada é linear. Isto porque o mecanismo de herança em vista (o da linguagem GNOME) é linear, tal como se referiu atrás.

Denota-se por $\mathcal{C}_N$ a categoria cartesiana livremente gerada por N. Cada elemento de CP diz-se uma *classe primária* e cada elemento de $|\mathcal{C}_N|$ diz-se uma *classe*. Uma classe não primária diz-se *composta* (ou *agregação*). Para cada par $c,c' \in CP$ tal que $c \leq c'$ denota-se por $\iota_{c,c'}:c \to c'$ o morfismo correspondente em $\mathcal{C}_N$, a que se chama *morfismo de inclusão* (de classe). O produto de classes denota-se por $c_1 \otimes \ldots \otimes c_m$, com projecções $\pi_i:c_1 \otimes \ldots \otimes c_m \to c_i$, para i=1,…,m. A classe terminal denota-se por $\mathbb{1}$.

Definição 3.2 *Conjunto de géneros*

O conjunto de géneros induzido por N é o conjunto $\overline{G}_N = G_N \cup G_N^\tau$ em que $G_N = G_{TD} \cup |\mathcal{C}_N|$ e $G_N^\tau = \{\tau_c : c \in CP\}$. □

Os elementos de G_N dizem-se *géneros próprios* e os elementos de G_N^τ dizem-se *géneros acção*.

Definição 3.3 *Assinatura de classe*

Uma assinatura de classe sobre N é um quádruplo <c,ACC,ACI,ATR> tal que:

- $c \in |\mathcal{C}_N|$;
- ACC é uma família de conjuntos, indexada por $G_N^* \otimes G_N^\tau$, tal que se $c \in CP$ então:
  - $\perp \in ACC_{\varepsilon,\tau_c}$;
  - $ACC_{\omega,\tau_{c'}} = \emptyset$ se $c \not\leq c'$;
  - $ACC_{\omega,\tau_{c'}} \subseteq ACC_{\omega,\tau_c}$ se $c \leq c'$;
- $ACI \subseteq ACC_{\varepsilon,\tau_c}$, e é tal que $\perp \in ACI$;
- ATR é uma família de conjuntos, indexada por $\overline{G}_N^* \otimes \overline{G}_N$, tal que se $c \in CP$ então:
  - $ATR_{\tau_c,bool} = \{\diamond\}$;[1]
  - $ATR_{\varepsilon,\tau_c} = \{\nabla\}$;[2]
  - $ATR_{\varepsilon,phs} = \{ph\}$;[3]

Assume-se ainda que os conjuntos das famílias ACC e ATR são disjuntos dois a dois. □

---

1 O atributo $\diamond$ indica, para cada acção, se esta está ou não activada.
2 O atributo ∇ indica qual a acção que vai ocorrer.
3 O atributo *ph* indica qual a fase (*phase*) do objecto

Dada uma assinatura <c,ACC,ACI,ATR>, c diz-se a *classe subjacente* . Cada elemento z∈ $ACC_{\omega,g}$ diz-se um *símbolo de acção* com géneros argumento ω e género resultante g. Cada elemento de elemento de ACI diz-se um *símbolo de acção interna*. Cada elemento a∈ $ATR_{\omega,g}$ diz-se um *símbolo de atributo* com géneros argumento ω e género resultante g. Um atributo em $ATR_{\varepsilon,g}$, com g ∈ G_N diz-se um *símbolo de atributo simples* . Define-se ainda a família ACE, indexada por $G_N^* \otimes G_N^\tau$, tal que $ACE_{\varepsilon,g}=ACC_{\varepsilon,g} \setminus ACI$ e $ACE_{\omega,g}=ACC_{\omega,g}$, para ω≠ε. Cada elemento de $ACE_{\omega,g}$ diz-se um *símbolo de acção externa* (ou *serviço*) com géneros argumento ω e género resultante g.

Definição 3.4 *Morfismo de assinaturas de classe*

Sejam <c,ACC,ACI,ATR> e <c',ACC',ACI',ATR'> duas assinaturas de classe sobre N. Então um *morfismo de assinaturas de classe* sobre N

$$\sigma:<c,ACC,ACI,ATR> \rightarrow <c',ACC',ACI',ATR'>$$

é um triplo < $\sigma_{id},\sigma_{ACC},\sigma_{ATR}$> tal que:

- $\sigma_{id}:c' \rightarrow c$ é um morfismo em $\mathcal{C}_N$;
- σ_{ACC} é uma família de aplicações $\sigma_{ACC_{\omega,g}}:ACC_{\omega,g} \rightarrow ACC'_{\omega,g}$, indexada por $G_N^* \otimes G_N^\tau$, verificando
  - $\sigma_{ACC_{\varepsilon,g}}(ACI) \subseteq ACI'$;
  - $\sigma_{ACC_{\omega,g}}(ACE_{\omega,g}) \subseteq ACE'_{\omega,g}$;
- σ_{ATR} é uma família de aplicações $\sigma_{ATR_{\omega,g}}:ATR_{\omega,g} \rightarrow ATR'_{\omega,g}$, indexada por $\overline{G}_N^* \otimes \overline{G}_N$. □

As duas condições sobre as aplicações σ_{ACC} servem para garantir que as acções internas (resp. externas) de c correspondem a acções internas (resp. externas) de c'.

As assinaturas de classe e respectivos morfismos constituem uma categoria $\mathcal{A}ss_N$. Um morfismo de assinaturas de classe σ diz-se uma *inclusão* sse σ_{id} for uma inclusão de classes e as aplicações $\sigma_{ATR_{\omega,g}}$ e $\sigma_{ACC_{\omega,g}}$ forem inclusões. No caso de σ_{id} ser uma projecção de classe e as aplicações $\sigma_{ATR_{\omega,g}}$ e $\sigma_{ACC_{\omega,g}}$ serem injecções, o morfismo de assinaturas de classe diz-se uma *injecção* .

Definição 3.5 *Assinatura de comunidade de objectos*

Uma *assinatura de comunidade de objectos* é um par Σ=<N, $\mathcal{S}g$> em que:

- N=<CP,≤> é uma rede de herança linear;
- $\mathcal{S}g:\mathcal{C}_N^{op} \rightarrow \mathcal{A}ss_N$ é um functor que preserva op-inclusões, coprodutos finitos e tal que:
  - $\mathcal{S}g(c)_1$=c, para toda a classe c∈ $|\mathcal{C}_N|$;

- $\mathcal{Sg}(\sigma)_{id}=\sigma$, para todo o morfismo σ em $\mathcal{C}_N$. □

Desta definição resulta que a especialização de classes corresponde a inclusão de assinaturas, i.e., se $c \leq c'$ então a assinatura de c' deverá *estar contida* na assinatura de c. Por outro lado, a assinatura de uma agregação de classes, $c_1 \otimes \ldots \otimes c_n$, é dada pelo coproduto das assinaturas das classes que compõem essa agregação, $c_1,\ldots,c_n$ ([Sernadas e Sernadas 94], [Sernadas et al 92e]). A assinatura da classe terminal é $<\mathbb{1},\emptyset,\emptyset,\emptyset>$.

De agora em diante assume-se fixada uma assinatura de comunidade (de objectos) $\Sigma=<N, \mathcal{Sg}>$ e denota-se $\mathcal{Sg}(c)$ por $\Sigma^c=<c,ACC^c,ACI^c,ATR^c>$.

Exemplo 3.6 *Relógios*

Recorde-se o problema dos relógios e seus utilizadores, introduzido no capítulo anterior. Apresenta-se a seguir uma assinatura para a classe *Relógio*:

- $ACC_{\varepsilon,\tau_{Relógio}}=\{\perp,novo,obs,tic\}$;
- $ACI=\{\perp,tic\}$;
- $ATR_{\varepsilon,phs}=\{ph\}$;
- $ATR_{\varepsilon,nat}=\{tempo\}$;
- $ATR_{\tau_{Relógio},bool}=\{\diamond\}$;
- $ATR_{\varepsilon,\tau_{Relógio}}=\{\nabla\}$;
- $ACE_{\varepsilon,\tau_{Relógio}}=\{novo,obs\}$. □

Exemplo 3.7 *Utilizadores*

Relativamente ao mesmo problema, apresenta-se agora uma assinatura para a classe *Utilizador*:

- $ACC_{\varepsilon,\tau_{Utilizador}}=\{\perp,novo,consultar\}$;
- $ACC_{Relógio,\ \tau_{Utilizador}}=\{compra\}$;
- $ACI=\{\perp,consultar\}$;
- $ATR_{\varepsilon,phs}=\{ph\}$;
- $ATR_{\varepsilon,nat}=\{ta,tp\}$;
- $ATR_{\varepsilon,Relógio}=\{rl\}$;
- $ATR_{\varepsilon,situação}=\{sit\}$;[1]

[1] Assume-se que situação $\in G_{TD}$ e $OP_{\varepsilon,situação}=\{pobre,rico\}$.

- $ATR_{\tau_{Utilizador},bool}=\{\Diamond\}$;
- $ATR_{\varepsilon,\tau_{Utilizador}}=\{\nabla\}$;
- $ACE_{\varepsilon,\tau_{Utilizador}}=\{novo\}$;
- $ACE_{Relógio,\ \tau_{Utilizador}}=\{compra\}$. □

Exemplo 3.8 *Relógios com defeito*

Ainda para o mesmo problema, apresenta-se agora uma assinatura para a classe *Drelógio*:

- $ACC_{\varepsilon,\tau_{Drelógio}}=\{\bot,novo,obs,tic,tac\}$;
- $ACC_{\varepsilon,\tau_{Relógio}}=\{\bot,novo,obs,tic\}$;
- $ACI=\{\bot,tic,tac\}$;
- $ATR_{\varepsilon,phs}=\{ph\}$;
- $ATR_{\varepsilon,nat}=\{tempo,x\}$;
- $ATR_{\tau_{Drelógio},bool}=\{\Diamond\}$;
- $ATR_{\varepsilon,\tau_{Drelógio}}=\{\nabla\}$.
- $ACE_{\varepsilon,\tau_{Drelógio}}=\{novo,obs\}$;
- $ACE_{\varepsilon,\tau_{Relógio}}=\{novo,obs\}$. □

Exemplo 3.9 *Utilizadores e Relógios*

Pretende-se agora definir uma assinatura para a classe *Utilizador*$\otimes$ *Relógio* (com morfismos de classe π_u:*Utilizador*$\otimes$ *Relógio*$\rightarrow$ *Utilizador* e π_r:*Utilizador*$\otimes$ *Relógio*$\rightarrow$ *Relógio*). A assinatura desta classe é dada pelo coproduto (em $\mathcal{A}ss_N$[1]) das assinaturas das classes *Utilizador* e *Relógio* . Convencionou-se que

- $\nabla_u=\mathcal{S}g(\pi_u)_{ATR_{\varepsilon,\tau_{Utilizador}}}(\nabla)$;
- $\nabla_r=\mathcal{S}g(\pi_r)_{ATR_{\varepsilon,\tau_{Relógio}}}(\nabla)$;
- $novo_u=\mathcal{S}g(\pi_u)_{ACC_{\varepsilon,\tau_{Utilizador}}}(novo)$;
- $novo_r=\mathcal{S}g(\pi_r)_{ACC_{\varepsilon,\tau_{Relógio}}}(novo)$;

etc…

ou seja, convencionou-se que os símbolos referentes à assinatura da classe *Utilizador* surjam indexados pela letra u e os símbolos referentes à assinatura da classe *Relógio* surjam indexados pela letra r. Assim,

---

1 Supõe-se N=<{Relógio,Utilizador},{(Relógio,Relógio),(Utilizador,Utilizador)}>.

- $ACC_{\varepsilon,\tau_{Relógio}}=\{\perp_r,novo_r,obs_r,tic_r\}$;
- $ACC_{\varepsilon,\tau_{Utilizador}}=\{\perp_u,novo_u,consultar_u\}$;
- $ACC_{Relógio,\ \tau_{Utilizador}}=\{compra_u\}$;
- $ACI=\{\perp_u,\perp_r,consultar_u,tic_r\}$;
- $ATR_{\varepsilon,phs}=\{ph_u,ph_r\}$;
- $ATR_{\varepsilon,nat}=\{ta_u,tp_u,tempo_r\}$;
- $ATR_{\varepsilon,Relógio}=\{rl_u\}$;
- $ATR_{\varepsilon,situação}=\{sit_u\}$;
- $ATR_{\tau_{Utilizador},bool}=\{\Diamond_u\}$;
- $ATR_{\varepsilon,\tau_{Utilizador}}=\{\nabla_u\}$;
- $ATR_{\tau_{Relógio},bool}=\{\Diamond_r\}$;
- $ATR_{\varepsilon,\tau_{Relógio}}=\{\nabla_r\}$. □

3.2.2 Termos e fórmulas

Assume-se fixada, de agora em diante, uma família X de conjuntos de variáveis infinitos, disjuntos dois a dois, indexada por G_N, e munida de uma família de aplicações (ditas *funções de escolha*), $\delta_g:2^{X_g}\rightarrow X_g$, indexada por G_N, tal que $\delta_g(Y)\in Y$, para cada $g\in G_N$.

Definição 3.10 *Termos*

Seja c uma classe. Então, T^c é uma família, indexada por G_N, de conjuntos de *termos* definidos indutivamente como se segue:

- $y\in T^c_g$, desde que $y\in X_g$;
- $f(t_1,\ldots,t_n)\in T^c_g$, desde que $f\in OP_{g_1\ldots g_n,g}$ e $t_i\in T^c_{g_i}$, para i=1,…,n;
- $z(t_1,\ldots,t_n)\in T^c_g$, desde que $z\in ACC^c_{g_1\ldots g_n,g}$ e $t_i\in T^c_{g_i}$, para i=1,…,n;
- $a(t_1,\ldots,t_n)\in T^c_g$, desde que $a\in ATR^c_{g_1\ldots g_n,g}$ e $t_i\in T^c_{g_i}$, para i=1,…,n;
- $\mu(t)\in T^c_{g'}$, desde que $\mu:g\rightarrow g'$ seja um morfismo de classes em $\mathcal{C}_N$ e $t\in T^c_g$. □

Cada um dos elementos de T^c_g diz-se um termo de género g na classe c. Um termo envolvendo apenas símbolos de Σ_{TD} e variáveis de género em G_{TD} diz-se um *termo de dados*.

Definição 3.11 *Fórmulas esquema*

Seja $c=c_1 \otimes \ldots \otimes c_m$ uma classe tal que $c_i \in$ CP, para i=1,…,m. O conjunto Q^c das *fórmulas esquema* sobre a classe c define-se indutivamente do seguinte modo:

- $t \in Q^c$, desde que $t \in T^c_{bool}$;
- $(t=t') \in Q^c$, desde que $t,t' \in T^c_g$;
- $(t' \equiv t'') \in Q^c$, desde que $t' \in T^c_{c'}$ e $t'' \in T^c_{c''}$;
- $\divideontimes \in Q^c$;
- $(\neg q) \in Q^c$, desde que $q \in Q^c$;
- $(q \Rightarrow q')$, $(q \mathbf{U} q')$, $(q \mathbf{S} q') \in Q^c$, desde que $q,q' \in Q^c$;[1]
- $(\forall y\ q) \in Q^c$, desde que $q \in Q^c$ e $y \in X_g$ para algum género g;
- $\alpha_z(t_1,\ldots,t_n) \in Q^c$, desde que $z \in ACE^c_{g_1\ldots g_n, \tau_{c_i}}$ e $t_j \in T^c_{g_j}$, para j=1,…,n, e i=1,…,m;
- $\beta_{c',z}(t',t_1,\ldots,t_n) \in Q^c$, desde que $c' \in$ CP, $z \in ACE^{c'}_{g_1\ldots g_n, \tau_{c'}}$, $t' \in T^c_{c'}$, $t_i \in T^c_{g_i}$, para i=1,…,n, e $ACC^c_{g_1\ldots g_n, \tau_c} \neq \emptyset$ para alguns $g_1,\ldots,g_n \in G_N^*$. □

A relação $\equiv$, definida sobre termos de género classe, é utilizada para denotar igualdade entre objectos. O seu significado será explicado com mais detalhe quando se definir a semântica das fórmulas.

A fórmula $\divideontimes$ representa o instante inicial do sistema (*bigbang*).

Os símbolos (de predicado) α e β são utilizados para representar a comunicação entre uma instância de uma classe (não necessariamente primária) e o resto da comunidade (ambiente). Assim, a fórmula $\alpha_z(t_1,\ldots,t_n)$ será utilizada para representar chamadas *do* ambiente, i.e., para determinar se a acção externa z foi chamada a partir do ambiente, com parâmetros instanciados com valores denotados por $t_1,\ldots,t_n$. A fórmula $\beta_{c',z}(t',t_1,\ldots,t_n)$ será utilizada para representar chamadas *para* o ambiente, i.e., será utilizada para determinar se a acção externa z da instância denotada por t' da classe c' está activada neste instante, se os seus parâmetros forem instanciados com valores denotados pelos termos $t_1,\ldots,t_n$. No caso de uma classe c não ter acções a sua linguagem esquema Q^c não inclui fórmulas envolvendo os predicados α e β. A não inclusão das fórmulas envolvendo predicados β prende-se com o facto de não ser possível a partir dessa classe, fazer chamadas para o ambiente. Em particular, $Q^{\mathbb{1}}$ não contém nenhuma destas fórmulas.

No apêndice B encontram-se definidos outros conectivos lógicos e operadores

[1] Os símbolos **U** e **S** representam os operadores temporais usuais *until* e *since*, respectivamente.

temporais definidos como abreviaturas, a partir dos conectivos e operadores agora introduzidos.

Uma fórmula envolvendo apenas termos de dados, conectivos lógicos não temporais, quantificadores e a operação de igualdade diz-se uma *fórmula esquema rígida de dados* .

Definição 3.12 *Conjunto de fórmulas numa classe*

Seja c uma classe. Então, o conjunto de fórmulas sobre c é o conjunto:

$L^c=\{x.q: x \in X_c, q \in Q^c\}$. □

Uma fórmula sem operadores temporais, sem símbolos de predicado α e β, e sem símbolos de atributo para além dos *atributos simples* diz-se um *constrangimento de estado*. Denota-se por CE^c o conjunto dos constrangimentos de estado sobre c.

Em algumas situações torna-se conveniente escrever uma fórmula x.*q* , substituindo em *q* cada ocorrência de cada símbolo de atributo *a* por x.*a* .

Exemplo 3.13 *Relógios*

Sobre a assinatura da classe Relógio, definida na secção anterior, admitindo que $r \in X_{Relógio}$, são exemplos de fórmulas:

- $(r. \Diamond(novo) \Leftrightarrow ((r.\ ph=PN) \wedge r.\ \alpha_{novo}))$; *(1)*
- $(((r. \nabla=novo) \wedge (A[^{r.tempo}_{0}]\ [^{r.ph}_{AN}])) \Rightarrow (\mathbf{X}A))$, qualquer que seja $A \in CE^{Relógio}$; *(2)*
- $(r. \Diamond(obs) \Leftrightarrow ((r.ph=AN) \wedge r.\ \alpha_{obs}))$; *(3)*
- $(((r. \nabla=obs) \wedge A) \Rightarrow (\mathbf{X}A))$, qualquer que seja $A \in CE^{Relógio}$; *(4)*
- $(r. \Diamond(tic) \Leftrightarrow (r.\ ph=AN))$; *(5)*
- $(((r. \nabla=tic) \wedge (A[^{r.tempo}_{r.tempo+1}])) \Rightarrow (\mathbf{X}A))$, qualquer que seja $A \in CE^{Relógio}$; *(6)*
- $(r. \Diamond(\bot) \Leftrightarrow true)$ *(7)*
- $(((r. \nabla=\bot) \wedge A) \Rightarrow (\mathbf{X}A))$, qualquer que seja $A \in CE^{Relógio}$; *(8)*
- $(\forall y\ ((x.\nabla=y) \Rightarrow r.\Diamond(y)))$. *(9)*

□

A asserção *(1)* caracteriza as condições de activação da acção externa *novo* de uma instância r da classe Relógio. Assim, esta acção só estará activada quando o seu atributo *ph* tiver valor *PN* e essa acção tiver sido chamada do ambiente. A asserção *(2)* traduz os efeitos desta acção num constrangimento de estado *A*. Assim, quando esta *acontecer*, o atributo *tempo* passará a ter o valor *0*, o atributo *ph* passará a ter o valor

AN, e qualquer outro atributo deverá manter o seu valor. As asserções *(3)*, *(4)*, *(5)*, *(6)*, *(7)* e *(8)* caracterizam as condições de activação e os efeitos das acções *obs* , *tic* e $\perp$, respectivamente. Finalmenta, a asserção *(9)* traduz o facto de uma acção só poder *acontecer* quando estiver activada.

Exemplo 3.14 *Utilizadores*

Sobre a assinatura da classe *Utilizador*, e admitindo que u$\in$ $X_{Utilizador}$, são exemplos de fórmulas:

- $(u.\ \Diamond(novo) \Leftrightarrow ((u.\ ph{=}PN) \wedge u.\ \alpha_{novo}))$; *(1)*
- $(((u.\ \nabla{=}novo) \wedge (A[^{u.ta}_{0}][^{u.tp}_{0}][^{u.sit}_{pobre}][^{u.\ ph}_{AN}])) \Rightarrow (\mathbf{X}A))$, *(2)*

  qualquer que seja A$\in$ $CE^{Utilizador}$;
- $((u.\ \Diamond(compra(r)) \Leftrightarrow ((u.rl{\neq}r) \wedge (u.\ ph{=}AN) \wedge u.\ \alpha_{compra}(r)))$, *(3)*
- $(((u.\ \nabla{=}compra(r)) \wedge (A\ [^{u.ta}_{0}][^{u.tp}_{0}][^{u.rl}_{r}][^{u.sit}_{rico}])) \Rightarrow (\mathbf{X}A))$, *(4)*

  qualquer que seja A$\in$ $CE^{Utilizador}$, r $\in T^{Utilizador}_{Relógio}$;
- $(u.\ \Diamond(consultar) \Leftrightarrow$

  $((u.sit{=}rico) \wedge (u.ph{=}AN) \wedge (u.rl{\neq}0) \wedge (u.rl \not\equiv u) \wedge u.\ \beta_{Relógio,obs}\ (u.rl)))$; *(5)*
- $(((u.\ \nabla{=}consultar) \wedge (\forall y\ A[^{u.tp}_{u.ta}][^{u.ta}_{y}])) \Rightarrow (\mathbf{X}A))$, *(6)*

  qualquer que seja A $\in CE^{Utilizador}$;
- $(u.\ \Diamond(\perp) \Leftrightarrow true)$ *(7)*
- $(((u.\ \nabla{=}\perp) \wedge A) \Rightarrow (\mathbf{X}A))$, qualquer que seja A$\in$ $CE^{Utilizador}$; *(8)*
- $(\forall y\ ((u.\nabla{=}y) \Rightarrow u.\ \Diamond(y)))$. *(9)*

□

Tal como no exemplo anterior, as asserções *(1)-(8)* caracterizam as condições de activação e os efeitos das acções de uma instância da classe *Utilizador* sobre um constrangimento de estado *A*. A asserção *(6)* , no entanto, merece especial relevo. Até agora, todas as outras asserções relativas aos efeitos de uma acção sobre um constrangimento de estado envolveram o cálculo de um valor para os atributos que dependeu apenas dos atributos dessa classe. A acção *consultar* , de acordo com a descrição do problema, corresponde a um *Utilizador* consultar o valor de um *Relógio*. O valor devolvido pelo *Relógio* não pode ser calculado recorrendo apenas aos atributos do *Utilizador*. Mais, do ponto de vista do *Utilizador* esse valor pode ser qualquer. Então, utiliza-se uma variável y quantificada universalmente para representar esse valor, i.e., $\forall$ y $A^{u.ta}_{y\ \cdots}$. Mais tarde, quando se explicitar a interacção entre um *Utilizador* e um *Relógio*, esta asserção deverá ser modificada de modo a reflectir a ligação entre o valor devolvido pelo *Relógio* e a expressão de valoração do atributo.

Exemplo 3.15 *Relógios e utilizadores*

Finalmente, sobre a assinatura da classe *Utilizador⊗Relógio*, admitindo que $v \in X_{\text{Utilizador}\otimes\text{Relógio}}$, são exemplos de fórmulas:

- $((v.rl_r = \pi_r(v)) \Rightarrow ((v.\nabla_u = \text{consultar}_u) \Rightarrow (v.\nabla_r = \text{obs}_r)))$; *(1)*
- $((v.rl_u = \pi_r(v)) \Rightarrow$

  $(((v.\nabla_u = \text{consultar}_u) \wedge (v.\nabla_r = \text{obs}_r) \wedge (A\,[^{v.tp_u}_{v.ta_u}]\,[^{v.ta_u}_{v.tempo_r}])) \Rightarrow (\mathbf{X}A)))$, *(2)*

  qualquer que seja $A \in CE^{\text{Utilizador}\otimes\text{Relógio}}$
- $((v.rl_u = \pi_r(v)) \Rightarrow (v.\beta_{\text{Relógio,obs}}\,(v.rl_u) \Leftrightarrow r.ph_r = AN))$ *(3)*

□

A asserção *(1)* traduz o facto de que se o *Relógio* do *Utilizador* (valor do seu atributo *rl*) coincidir com a instância de *Relógio* considerada ($\pi_r(v)$) então a ocorrência da acção *consultar* do *Utilizador* obriga à ocorrência da acção *obs* do *Relógio*, i.e., a acção *consultar* do *Utilizador chama* a acção *obs* do relógio. A asserção *(2)* caracteriza os efeitos dessa chamada. Compare-se esta asserção com a asserção *(6)* do exemplo anterior. Neste caso, a expressão que surge na valoração do atributo *ta* já pode ser calculada com base nos atributos da assinatura da classe. A asserção *(3)* dá uma caracterização do predicado $\beta_{\text{Relógio,obs}}$ à custa das condições de activação da acção *obs*, nas condições em pode haver uma chamada (é equivalente à condição de activação da acção *obs* sem o predicado α_{obs}).

3.3 Semântica

Pretende-se nesta secção definir estruturas de interpretação para a GCL bem como uma relação de satisfação.

3.3.1 Estruturas de interpretação

Assume-se fixada uma álgebra *td* sobre a assinatura $\Sigma_{TD}=\langle G_{TD}, OP\rangle$ tal que $td_{\text{bool}}=\{\text{true}_{td}, \text{false}_{td}\}$, $td_{\text{nat}}=\mathbb{N}_\circ$ e $td_{\text{phs}}=\{PN_{td}, AN_{td}, PM_{td}\}$.

Definição 3.16 *Functor população*

Um functor $\mathcal{P}op:\mathcal{C}_N \rightarrow \mathcal{S}et$ sobre $\Sigma=\langle N, Sg\rangle$ tal que, para cada $c \in CP$, $\mathcal{P}op(c)=\mathbb{N}_\circ$ e que preserve produtos finitos diz-se um *functor população*. □

Desta definição resulta que a população de uma classe agregação $c_1\otimes\ldots\otimes c_n$ é dada

pelo produto das populações das classes $c_1,\ldots,c_n$.

De agora em diante assume-se fixado um functor população $\mathcal{P}op$. Denota-se $\mathcal{P}op(c)$ por $\mathcal{P}op_c$, qualquer que seja a classe, e estende-se a notação de tal modo que $\mathcal{P}op_g$ denota td_g, quando $g\in G_{TD}$. Em particular, $\mathcal{P}op_{\mathbb{1}}$ é um conjunto singular, denotado por $\{\mathbb{1}\}$.

Falta ainda encontrar um suporte apropriado para cada género τ_c. Esse suporte é definido com base no seguinte conjunto.

Definição 3.17 *Conjunto de acções*

O conjunto de acções de uma classe primária c sobre Σ relativamente a $\mathcal{P}op$ é o conjunto $acc(c)$, definido do seguinte modo:

$$\bigcup_{<g_1\ldots g_n>\in G_N^*} \{z(u_1,\ldots,u_n): z\in ACC^c_{g_1\ldots g_n,\tau_c},\ u_i\in \mathcal{P}op_{g_i},\ \text{para } i=1,\ldots,n\} \ \cup$$

$$\bigcup_{<c',g_1\ldots g_n>\in|\mathcal{C}_N|\otimes G_N^*} \{\mathcal{S}g(\sigma)_{ACC_{g_1\ldots g_n,\tau_c}}(z)(u_1,\ldots,u_n): \sigma:c'\to c \text{ é um morfismo de classes em } \mathcal{C}_N,\ z\in ACC^c_{g_1\ldots g_n,\tau_c},\ u_i\in \mathcal{P}op_{g_i},\ \text{para } i=1,\ldots,n\}. \quad \square$$

Recordem-se agora os exemplos **3.6** e **3.9**. De acordo com a definição anterior, surgem no conjunto $acc(Relógio)$, entre outros, os elementos $novo$ e $novo_r$. Estes dois elementos referem-se à mesma acção, em dois contextos diferentes. Como é evidente, quando se proceder à interpretação de termos, a *interpretação* de uma acção deverá ser independente do contexto em que ela surge (numa classe isolada ou numa agregação). Assim, estabelece-se uma relação de equivalência entre as acções.

Definição 3.18 *Equivalência de acções*

Nas condições da definição anterior, e sobre o conjunto $acc(c)$, define-se a seguinte relação binária, $\sim_c \subseteq acc(c)\otimes acc(c)$:

$$\sim_c = \bigcup_{<c',g_1\ldots g_n>\in|\mathcal{C}_N|\otimes G_N^*} \{(z(u_1,\ldots,u_n),\ \mathcal{S}g(\sigma)_{ACC_{g_1\ldots g_n,\tau_c}}(z)(u_1,\ldots,u_n)):\ \sigma:c'\to c \text{ é um morfismo de classes em } \mathcal{C}_N,\ z\in ACC^c_{g_1\ldots g_n,\tau_c},\ u_i\in \mathcal{P}op_{g_i},\ \text{para } i=1,\ldots,n\}.$$

Esta relação induz uma *relação de equivalência* $\approx_c$ *entre acções* dada pelo fecho simétrico da relação $\sim_c$. $\square$

De agora em diante assume-se que $\mathcal{P}op_{\tau_c}$ denota o conjunto $acc(c)/\approx_c$, cujos elementos são da forma $[z(u_1,\ldots,u_n)]=\{x\in acc(c): x\approx_c z(u_1,\ldots,u_n)\}$. De acordo com a definição, $novo \approx_{Relógio} novo_r$ e portanto $[novo]=[novo_r]$.

Definição 3.19 *Instância de classe*

Uma instância sobre Σ relativamente a $\mathcal{P}op$ é um par <c,u> tal que c é uma classe e

$u \in \mathcal{P}op_c$. □

No caso do exemplo dos relógios, apresentado atrás, o par <*Relógio*,1> denota a primeira instância da classe *Relógio*. O par <*Utilizador*⊗*Relógio*,(2,1)> denota o par composto pela primeira instância da classe *Relógio* e pela segunda instância da classe *Utilizador*.

Definição 3.20 *Morfismo de instâncias*

Um morfismo de instâncias sobre Σ relativamente a $\mathcal{P}op$, $\sigma:<c,u> \rightarrow <c',u'>$ é um morfismo de classes $\sigma : c \rightarrow c'$ tal que $\mathcal{P}op(\sigma)(u)=u'$. □

Com base nas noções anteriores, é agora possível introduzir a categoria das instâncias.

Proposição 3.21 *Categoria das instâncias*

As instâncias sobre Σ relativamente a $\mathcal{P}op$, com os respectivos morfismos constituem uma *categoria*, $Inst_{\Sigma,\mathcal{P}op}$. Esta categoria é cartesiana. □

A prova desta proposição para OSL encontra-se em [Sernadas e Sernadas 94]. O objecto terminal desta categoria é a instância $<\mathbb{1}, \mathbb{1}>$.

Por definição de categoria cartesiana, resulta desta proposição que existem em $Inst_{\Sigma,\mathcal{P}op}$ todas as instâncias de agregações de classes.

Pretende-se agora descrever o comportamento de uma instância numa dada comunidade de objectos. Por descrição do comportamento entende-se a descrição de uma possível evolução dos valores dos atributos dessa instância. Chama-se *vida* a cada uma dessas possíveis evoluções. Ao conjunto de valores que os atributos tomam em cada *instante* chama-se *situação*.

Definição 3.22 *Conjunto de atributos*

O *conjunto de atributos da classe c*, de género g, sobre Σ relativamente a $\mathcal{P}op$ é o conjunto $atr(c)_g$ definido do seguinte modo:

$$\bigcup_{<g_1 \ldots g_n> \in \overline{G}_N^*} \{a(u_1,\ldots,u_n): a \in ATR^c_{g_1 \ldots g_n,g},\ u_i \in \mathcal{P}op_{g_i}, \text{ para } i=1,\ldots,n\} .$$

O conjunto

$$atr(c) = \bigcup_{g \in \overline{G}_N} atr(c)_g$$

diz-se o conjunto dos atributos da classe c sobre Σ relativamente $\mathcal{P}op$. □

No contexto do exemplo **3.6**, a assinatura da classe *Relógio* dá origem aos seguintes

conjuntos de atributos: $atr(Relógio)_{nat}$={tempo}; $atr(Relógio)_{bool}$ ={◇(⊥),◇(novo), ◇(obs), ◇(tic)}; $atr(Relógio)_{phs}$={*ph*}; $atr(Relógio)_{\tau_{Relógio}}$={∇ }.

Definidos os atributos, é agora necessário encontrar um domínio semântico apropriado (i.e., definir a noção de *situação*). Uma solução consiste em definir uma situação como um tuplo com tantas componentes quantos os atributos e tal que cada componente corresponda ao valor do atributo nessa situação.

Para cada atributo b∈ $atr(c)_g$, $\mathcal{P}op_g$ diz-se o *domínio* de b, e denota-se por D_b.

Definição 3.23 *Situação*

O *conjunto das situações da classe c* sobre Σ relativamente a $\mathcal{P}op$ é o produto em $\mathcal{S}et$ dos conjuntos D_b , para b ∈$atr(c)$, cujo vértice se denota por $sit(c)$ $=\otimes_{b\in atr(c)} D_b$ e cujas projecções se denotam por $p_b:sit(c)\rightarrow D_b$. □

Em particular, o conjunto das situações da classe $\mathbb{1}$ tem que ser um conjunto singular, por exemplo { $\mathbb{1}$}.

Novamente para o exemplo da classe relógio, uma possível representação para o conjunto de situações é a seguinte:

$$sit(c) = td_{nat} \otimes td_{bool} \otimes td_{bool} \otimes td_{bool} \otimes td_{bool} \otimes td_{phs} \otimes acc(Relógio)/\approx_c$$

com projecções $p_{tempo}:td_{nat} \otimes td_{bool} \otimes td_{bool} \otimes td_{bool} \otimes td_{bool} \otimes td_{phs} \otimes acc(Relógio)/\approx_c \rightarrow td_{nat}$, etc….

Definição 3.24 *Imagem de situação*

Seja σ:c'→c um morfismo de classes e V' ∈$sit(c')$ sobre Σ para $\mathcal{P}op$. Então, a *imagem* $\sigma(V')$ de V' sob σ é a única situação V∈ $sit(c)$ tal que

$$p_{a(u_1,\ldots,u_n)}(V)=p_{\mathcal{S}g(\sigma)_{ATR_{g_1\ldots g_n,g}}(a)(u_1,\ldots,u_n)}(V')$$

para cada a∈ $ATR_{g_1\ldots g_n,g}$ e $u_i\in \mathcal{P}op_{g_i}$, para i=1,…,n. □

Definição 3.25 *Vida de uma classe*

Seja c uma classe e λ: $\mathbb{N}_\circ \rightarrow sit(c)$ uma sucessão. Então, o par <c,λ> diz-se uma *vida* sobre Σ relativamente a $\mathcal{P}op$ se, sempre que c ∈CP e k∈ $\mathbb{N}_\circ$:

- $p_{\diamond(\perp)}(\lambda_k)=true_{td}$;
- se $p_{\nabla}(\lambda_k)=\perp$ então $p_b(\lambda_{k+1})=p_b(\lambda_k)$, para cada b∈ $atr(c)$ tal que b≠∇;
- se $p_{\nabla}(\lambda_k)=z(u_1,\ldots,u_n)$ então $p_{\diamond([z(u_1,\ldots,u_n)])}(\lambda_k)=true_{td}$. □

Uma vida <c,λ > diz-se uma vida na classe c. É óbvio que só existe uma vida na classe terminal 𝟙, que se denota por <𝟙, 𝟙>.

A primeira condição exige que a acção ⊥ esteja activada em qualquer instante de uma vida. Por sua vez, ocorrência desta deve implicar que o valor dos atributos não muda, excepto possivelmente o valor do atributo ∇. Esta propriedade é expressa pela segunda condição. Finalmente, a terceira condição significa que uma acção só pode ocorrer numa situação em que esteja activada.

Definição 3.26 *Morfismo de vidas*

Sejam <c, λ> e <c', λ'> duas vidas sobre Σ relativamente a $\mathcal{P}op$. Então um *morfismo de vidas* sobre Σ relativamente a $\mathcal{P}op$, $\sigma:\langle c',\lambda'\rangle\rightarrow\langle c,\lambda\rangle$ é um morfismo de classes $\sigma:c'\rightarrow c$ tal que $\lambda_k=\sigma(\lambda'_k)$, qualquer que seja $k\in\mathbb{N}_0$. □

Proposição 3.27 *Categoria das vidas*

As vidas sobre Σ relativamente a $\mathcal{P}op$ formam uma categoria, $\mathcal{V}id_{\Sigma,\mathcal{P}op}$. Esta categoria é cartesiana. □

A prova deste resultado encontra-se em [Sernadas e Sernadas 94]. O objecto terminal desta categoria é a vida <𝟙,𝟙>.

Como corolário desta proposição, resulta que dadas vidas $\langle c_1,\lambda_1\rangle,\ldots,\langle c_n,\lambda_n\rangle$ é sempre possível construir uma vida para a agregação $c_1\otimes\ldots\otimes c_n$, a partir de $\lambda_1,\ldots,\lambda_n$.

Def/Proposição 3.28 *Vida de uma comunidade*

A vida de uma comunidade sobre Σ relativamente a $\mathcal{P}op$ é um functor

$$\Lambda: \mathcal{I}nst_{\Sigma,\mathcal{P}op}\rightarrow\mathcal{V}id_{\Sigma,\mathcal{P}op}$$

transformando cada <c,u> numa vida na classe c. Este functor preserva produtos finitos. □

Definição 3.29 *Comunidade de objectos*

Uma *comunidade de objectos* sobre Σ é um par $co=\langle\mathcal{P}op,\Lambda\rangle$ em que:

- $\mathcal{P}op:\mathcal{C}_N\rightarrow\mathcal{S}et$ é um functor população sobre Σ;
- $\Lambda: \mathcal{I}nst_{\Sigma,\mathcal{P}op}\rightarrow\mathcal{V}id_{\Sigma,\mathcal{P}op}$ é uma vida de comunidade sobre Σ relativamente a $\mathcal{P}op$. □

Assim, uma comunidade de objectos é compostas por instâncias das classes e respectivas vidas.

De agora em diante supõe-se fixada uma comunidade de objectos $co=\langle\mathcal{P}op,\Lambda\rangle$.

Para terminar a definição da semântica falta apenas considerar a avaliação dos símbolos relativos à interacção com o ambiente. Segue-se um método semelhante ao utilizado para a definição de situações e vidas.

Definição 3.30 *Conjunto ambiente*

O *conjunto ambiente* para uma classe c, com $c=c_1 \otimes \ldots \otimes c_m$, relativamente a $\mathcal{P}op$ é o conjunto $amb(c)$ definido do seguinte modo:

$(U_{<g_1 \ldots g_n> \in G_N{}^*} \{\alpha_z(u_1,\ldots,u_n)$: $z \in ACE^{c}_{g_1 \ldots g_n, \tau_{c_i}}$, i=1,…,m, j=1,…,n $\})$ U

$(U_{<c',g_1 \ldots g_n> \in CP \otimes G_N{}^*} \{\beta_{c',z}(u',u_1,\ldots,u_n)$: $z \in ACE^{c'}_{g_1 \ldots g_n, \tau_{c'}}$, $u' \in \mathcal{P}op_{c'}$, $u_i \in \mathcal{P}op_{g_i}$, i=1,…,n, desde que $ACE^{c}_{g'_1 \ldots g'_n, \tau_c} \neq \emptyset$, para $g'_1,\ldots,g'_n \in G_N\}$) □

Tal como na definição da linguagem, no caso de uma classe não ter acções o ambiente de qualquer das suas intâncias é vazio. Em particular, o ambiente da classe $\mathbb{1}$ é vazio.

Escolheu-se para domínio semântico do conjunto ambiente, o *contexto*, um tuplo com tantas componentes quantos os elementos do ambiente e tal que a cada componente do tuplo faz corresponder o valor do elemento do ambiente nesse contexto. O domínio de cada $b \in amb(c)$ denota-se por D_b e coincide com $\mathcal{P}op_{bool}$.

Definição 3.31 *Contexto*

O *conjunto dos contextos* da classe c sobre Σ relativamente a $\mathcal{P}op$ é o produto em $\mathcal{S}et$ dos conjuntos D_b, para $b \in amb(c)$, cujo vértice se denota por $cxt(c) = \otimes_{b \in amb(c)} D_b$ e cujas projecções se denotam por $p_b : cxt(c) \rightarrow D_b$. □

Alguns dos contextos definidos podem ser eliminados à partida, se se tiver em conta a semântica pretendida para a linguagem GNOME.

Definição 3.32 *Contexto admissível*

Seja <c,u> uma instância. Um contexto $ct \in cxt(c)$ diz-se *admissível* para <c,u> sse

- $p\beta_{c',z}(u',u_1,\ldots,u_n)(ct) = false_{td}$, quaisquer que sejam $\sigma: c \rightarrow c'$, $z \in ACE^{c}_{g_1 \ldots g_n, \tau_{c'}}$, $\mathcal{P}op(\sigma)(u) = u'$ e $u_i \in \mathcal{P}op_{g_i}$, para i=1,…,n;
- $p\beta_{c',z}(0,u_1,\ldots,u_n)(ct) = false_{td}$, quaisquer que sejam c', $z \in ACE^{c}_{g_1 \ldots g_n, \tau_{c'}}$, e $u_i \in \mathcal{P}op_{g_i}$, para i=1,…,n.

O conjunto dos contextos admissíveis denota-se por $cont(<c,u>)$. □

A primeira condição traduz o facto de uma instância não poder chamar um serviço de si própria. A segunda condição traduz o facto de nenhum serviço da instância 0 de qualquer classe c estar activado.

O contexto de $<\mathbb{1},\mathbb{1}>$ tem que ser um conjunto singular, que se denota por $\{\mathbb{1}\}$. Este contexto é obviamente admissível.

Definição 3.33 *Imagem de contexto*

Sejam $\sigma:c'\to c$ um morfismo de classes e $ct' \in cxt(c')$. Então, a *imagem* $\sigma(ct')$ de ct' sob σ é o único contexto $ct \in cxt(c)$ tal que:

- $p\alpha_{z(u_1,\ldots,u_n)}(ct) = p\,\alpha_{\mathcal{S}g(\sigma)ACC_{g_1\ldots g_n,\tau_c}(z)(u_1,\ldots,u_n)}(ct')$
- $p\beta_{c'',z(u'',u_1,\ldots,u_n)}(ct) = p\,\beta_{c'',z(u'',u_1,\ldots,u_n)}(ct')$. □

Proposição 3.34 *Imagem de contextos admissíveis*

Sejam $\sigma:c'\to c$ um morfismo de classes, $u'\in \mathcal{P}op_{c'}$, $ct'\in cont(<c',u'>)$ e $ct=\sigma(ct')$. Então, a imagem de ct' sob σ, ct, é um contexto admissível para $<c,\mathcal{P}op(\sigma)(u')>$.

prova:

Para mostrar que ct é um contexto admissível é necessário mostrar que se verificam as duas condições da definição, o que resulta directamente das definições anteriores e a composição de morfismos. □

Definição 3.35 *Sequência de contextos*

Seja $<c,u>$ uma instância sobre Σ relativamente a $\mathcal{P}op$. Uma *sequência de contextos* para $<c,u>$ é uma sucessão $\omega:\mathbb{N}_\circ\to cont(<c,u>)$.

Nestas condições, o par $<<c,u>,\omega>$ diz-se uma *sequência de contextos* sobre Σ relativamente a $\mathcal{P}op$. □

Existe apenas uma sequência de contextos para $<\mathbb{1},\mathbb{1}>$, que se denota por $\mathbb{1}$.

Definição 3.36 *Morfismo de sequência de contextos*

Sejam $<<c,u>,\omega>$ e $<<c',u'>,\omega'>$ duas sequências sobre Σ relativamente a $\mathcal{P}op$. Então, um *morfismo de sequências de contextos* sobre Σ relativamente $\mathcal{P}op$, $\sigma:<<c',u'>,\omega'>\to<<c,u>,\omega>$ é um morfismo de instâncias $\sigma:<c,u>\to<c',u'>$ tal que $\omega_k=\sigma(\omega'_k)$. □

Proposição 3.37 *Categoria dos contextos*

As sequências de contextos sobre Σ relativamente a $\mathcal{P}op$ com os respectivos morfismos constituem uma categoria, que se designa por $\mathcal{C}ont_{\Sigma,\mathcal{P}op}$. □

Proposição 3.38

Sejam $<<c_1,u_1>,\omega_1>$ e $<<c_2,u_2>,\omega_2>$ duas sequências de contextos sobre Σ relativamente a $\mathcal{P}op$. Se, para cada $\beta_{c',z}(u',u'_1,\ldots,u'_n)\in amb(c_1)\cap amb(c_2)$, $p\beta_{c',z(u',u'_1,\ldots,u'_n)}(\omega_{1k})=p\beta_{c',z(u',u'_1,\ldots,u'_n)}(\omega_{2k})$, qualquer que seja $k\in\mathbb{N}_\circ$, então o produto

de $\langle\langle c_1, u_1\rangle, \omega_1\rangle$ e $\langle\langle c_2, u_2\rangle, \omega_2\rangle$ está definido em $\mathcal{C}ont_{\Sigma,\mathcal{P}op}$.

prova:

O produto de $\langle\langle c_1, u_1\rangle, \omega_1\rangle$ e $\langle\langle c_2, u_2\rangle, \omega_2\rangle$ é $\langle\langle c_1 \otimes c_2, (u_1, u_2)\rangle, \omega\rangle$ munido com as projecções π_1, π_2 e tal que, para i=1,2:

- $p\alpha_{Sg(\pi_i)ACC_{g_1\ldots g_n,\tau_c}(z)}(u'_1,\ldots,u'_n)(\omega_k) = p\alpha_z(u'_1,\ldots,u'_n)(\omega_{i_k})$, se $\alpha_z(u'_1,\ldots,u'_n) \in amb(c_i)$
- $p\beta_{c'',z}(u',u'_1,\ldots,u'_n)(\omega_k) = p\beta_{c'',z}(u',u'_1,\ldots,u'_n)(\omega_{i_k})$, se $\beta_{c'',z}(u',u'_1,\ldots,u'_n) \in amb(c_i)$. □

Proposição 3.39 *Contexto de uma comunidade*

O *contexto de uma comunidade* sobre Σ relativamente a $\mathcal{P}op$ é um functor

$\Omega: Inst_{\Sigma,\mathcal{P}op} \rightarrow \mathcal{C}ont_{\Sigma,\mathcal{P}op}$

transformando cada <c,u> numa sequência de contextos para <c,u>, e tal que quaisquer que sejam <c,u>,<c',u'> $\in |Inst_{\Sigma,\mathcal{P}op}|$, se $\beta_{c'',z}(u'',u_1,\ldots,u_n) \in amb(c) \cap amb(c')$ então $p\beta_{c',z}(u'',u_1,\ldots,u_n)(\Omega(\langle c,u\rangle)_{2,k}) = p\beta_{c',z}(u'',u_1,\ldots,u_n)(\Omega(\langle c',u'\rangle)_{2,k})$, para $k \in \mathbb{N}_0$.

Este functor preserva produtos finitos.

prova:

- $\Omega(\langle \mathbb{1}, \mathbb{1}\rangle) = \langle\langle \mathbb{1}, \mathbb{1}\rangle, \mathbb{1}\rangle$
- $\Omega(\langle c_1 \otimes c_2, u\rangle) = \Omega(\langle c_1, u_1\rangle) \otimes \Omega(\langle c_2, u_2\rangle)$, desde que $\pi_i(u) = u_i$, com i=1,2. □

Admite-se de agora em diante fixado um contexto Ω para a comunidade *co*.

3.3.2 Relação de satisfação

Com base na secção anterior é agora possível definir a noção de interpretação de termos e de satisfação de fórmulas.

Definição 3.40 *Atribuição*

Uma *atribuição* θ em *co* é uma família indexada por G_N de aplicações $\theta_g: X_g \rightarrow \mathcal{P}op_g$. Sendo $y \in X_g$, duas atribuições θ e θ' dizem-se *y-equivalentes* sse para todo o $g' \in G_N$ e $y' \in X_{g'}$, $\theta_{g'}(y') = \theta'_{g'}(y')$, se $y' \neq y$. □

Definição 3.41 *Interpretação de termos*

A *interpretação de termos* sobre uma classe c, para a atribuição θ, no ponto $k \in \mathbb{N}_0$ da instância $u \in \mathcal{P}op_c$ em *co* define-se indutivamente do seguinte modo:

- $[\![y]\!]^{\theta}_{k,u,co} = \theta_g(y)$, desde que $y \in X_g$;
- $[\![f(t_1,\ldots,t_n)]\!]^{\theta}_{k,u,co} = f_{td}([\![t_1]\!]^{\theta}_{k,u,co},\ldots,[\![t_n]\!]^{\theta}_{k,u,co})$;

- $[\![z(t_1,\ldots,t_n)]\!]^{\theta}_{k,u,co} = [z([\![t_1]\!]^{\theta}_{k,u,co},\ldots,[\![t_n]\!]^{\theta}_{k,u,co})]$;
- $[\![a(t_1,\ldots,t_n)]\!]^{\theta}_{k,u,co} = p_a([\![t_1]\!]^{\theta}_{k,u,co},\ldots,[\![t_n]\!]^{\theta}_{k,u,co})(\Lambda(\langle c,u\rangle)_{2,k})$;
- $[\![\mu(t)]\!]^{\theta}_{k,u,co} = \mathcal{P}op(\mu)([\![t]\!]^{\theta}_{k,u,co})$. □

A avaliação de termos é definida indutivamente, sobre a sua estrutura. Resulta desta definição que o resultado da avaliação de um termo $t \in T^c_g$ é um elemento de $\mathcal{P}op_g$. A interpretação de termos de dados é feita na álgebra *td*, e denota-se por $[\![t]\!]^{\theta}_{td}$ o valor do termo t em *td*, para atribuição θ.

Os únicos termos cuja a avaliação merece um pouco mais de atenção são os termos resultantes de atributos. Considere-se o termo $a(t_1,\ldots,t_n)$, com $t_i \in T^c_{g_i}$, para i=1,…,n. Da definição, tal como se referiu, resulta que $[\![t_i]\!]^{\theta}_{k,u,co} \in \mathcal{P}op_{g_i}$, logo $a([\![t_1]\!]^{\theta}_{k,u,co},\ldots, [\![t_n]\!]^{\theta}_{k,u,co}) \in atr(c)$. Então, o valor deste atributo é dado pela situação correspondente ao momento k da vida da instância <c,u> na comunidade *co*, i.e., pela situação $\Lambda(\langle c,u\rangle)_{2,k}$.

A definição de uma relação de satisfação para fórmulas esquema quaisquer, envolve em particular, a avaliação dos símbolos de predicado α e β. Para tal, é necessário considerar um contexto Ω para a comunidade *co*. Esta relação denota-se por $\Vdash_c$.

Definição 3.42 *Satisfação de fórmula esquema*

A *satisfação de fórmula esquema* sobre uma classe $c=c_1 \otimes \ldots \otimes c_n$, para a atribuição θ, no ponto $k \in \mathbb{N}_0$ de $u \in \mathcal{P}op_c$ relativamente a Ω em *co* define-se indutivamente do seguinte modo:

- $\theta,k,u,\Omega,co \Vdash_c t$ sse $[\![t]\!]^{\theta}_{k,u,co} = true_{td}$;
- $\theta,k,u,\Omega,co \Vdash_c (t=t')$ sse $[\![t]\!]^{\theta}_{k,u,co} = [\![t']\!]^{\theta}_{k,u,co}$;
- $\theta,k,u,\Omega,co \Vdash_c (t' \equiv t'')$ sse $c'^{\top} = c''^{\top}$ e $[\![t']\!]^{\theta}_{k,u,co} = [\![t'']\!]^{\theta}_{k,u,co}$;
- $\theta,k,u,\Omega,co \Vdash_c \divideontimes$ sse k=0;
- $\theta,k,u,\Omega,co \Vdash_c (\neg q)$ sse não $\theta,k,u,\Omega,co \Vdash_c q$;
- $\theta,k,u,\Omega,co \Vdash_c (q \Rightarrow q')$ sse se $\theta,k,u,\Omega,co \Vdash_c q$ então $\theta,k,u,\Omega,co \Vdash_c q'$;
- $\theta,k,u,\Omega,co \Vdash_c (q \mathbf{U} q')$ sse existe $k'>k$ tal que $\theta,k',u,\Omega,co \Vdash_c q'$ e, para qualquer $k<k''<k'$, $\theta,k'',u,\Omega,co \Vdash_c q$;
- $\theta,k,u,\Omega,co \Vdash_c (q \mathbf{S} q')$ sse existe $k'<k$ tal que $\theta,k',u,\Omega,co \Vdash_c q'$ e, para qualquer $k'<k''<k$, $\theta,k'',u,\Omega,co \Vdash_c q$;
- $\theta,k,u,\Omega,co \Vdash_c (\forall y\ q)$ sse $\theta',k,u,\Omega,co \Vdash_c q$, para qualquer atribuição θ', y-equivalente a θ;
- $\theta,k,u,\Omega,co \Vdash_c \alpha_z(t_1,\ldots,t_n)$ sse $p_{\alpha_z}([\![t_1]\!]^{\theta}_{k,u,co},\ldots,[\![t_n]\!]^{\theta}_{k,u,co})(\Omega(\langle c,u\rangle)_{2,k}) = true_{td}$;

- $\theta,k,u, \Omega,co \Vdash_c \beta_{c',z}(t,t_1,\ldots,t_n)$ sse $p\ \beta_{c',z}([\![t]\!]^{\theta}_{k,u,co},[\![t_1]\!]^{\theta}_{k,u,co},\ldots,[\![t_n]\!]^{\theta}_{k,u,co})(\Omega(<c,u>)_{2,k})=true_{td}$. □

A avaliação de uma fórmula esquema rígida de dados pode ser estabelecida apenas na álgebra *td*, e *td* ,$\theta \Vdash_c$ q denota a satisfação da fórmula esquema rígida de dados q na álgebra *td*, relativamente a θ .

Tal como se disse, a relação $\equiv$ denota igualdade entre objectos. Assim, diz-se que dois termos t e t', de géneros c e c' respectivamente, denotam o mesmo objecto sse denotarem a mesma instância da mesma classe topo.

A partir desta definição, é imediato estabelecer uma relação de satisfação para fórmulas da linguagem.

Definição 3.43 *Satisfação de fórmula*

Define-se *satisfação* de uma formula x.q $\in L^c$ para uma atribuição θ , no ponto $k \in \mathbb{N}_0$ de $u \in \mathcal{P}op_c$ relativamente a Ω em *co* do seguinte modo: θ ,k,u,Ω, $co \Vdash_c$ x.q sse $\theta,k,u, \Omega,co \Vdash_c$ q desde que $\theta_c(x)=u$. □

Definição 3.44 *Satisfação de fórmula em comunidade*

Uma comunidade de objectos *co* satisfaz a fórmula $\varphi \in L^c$, o que se denota por $co \Vdash_c \varphi$, sse $\theta,k,u, \Omega,co \Vdash_c \varphi$, quaisquer que sejam a atribuição θ, $k \in \mathbb{N}_0$, $u \in \mathcal{P}op_c$ e o contexto de comunidade Ω. □

3.4 Instituição

O conceito de instituição foi introduzido em [Goguen e Burstall 84], [Goguen e Burstall 92] tendo como objectivo principal fornecer um enquadramento formal para que certos dos problemas pudessem ser tratados e discutidos independentemente das particularidades de um qualquer sistema lógico.

O objectivo deste capítulo é estabelecer uma instituição para a Lógica de Certificação GNOME, com base na semântica de ciclos de vida, acabada de introduzir, seguindo a ideia apresentada [Sernadas e Sernadas 94].

Definição 3.45 *Mapas de tradução*

Seja $\sigma:c' \rightarrow c$ um morfismo de classes. Este morfismo induz os seguintes *mapas de tradução*:

- $\sigma_{T,g}:T^c_g \rightarrow T^{c'}_g$, para $g \in \overline{G}_N$, define-se indutivamente como se segue:

- $\sigma_{T,g}(y)=y$;
- $\sigma_{T,g}(f(t_1,\ldots,t_n)) = f(\sigma_{T,g_1}(t_1),\ldots,\sigma_{T,g_n}(t_n))$;
- $\sigma_{T,g}(z(t_1,\ldots,t_n)) = \mathcal{Sg}(\sigma)_{ACC_{g_1\ldots g_n,g}}(z)(\sigma_{T,g_1}(t_1),\ldots,\sigma_{T,g_n}(t_n))$;
- $\sigma_{T,g}(a(t_1,\ldots,t_n)) = \mathcal{Sg}(\sigma)_{ATR_{g_1\ldots g_n,g}}(a)(\sigma_{T,g_1}(t_1),\ldots,\sigma_{T,g_n}(t_n))$;
- $\sigma_{T,g'}(\mu(t)) = \mu(\sigma_{T,g}(t))$.

- $\sigma_Q:Q^c\rightarrow Q^{c'}$ define-se indutivamente do seguinte modo:
  - $\sigma_Q(t) = \sigma_{T,bool}(t)$;
  - $\sigma_Q((t=t')) = (\sigma_{T,g}(t)=\sigma_{T,g}(t'))$;
  - $\sigma_Q((t'\equiv t'')) = (\sigma_{T,c''}(t')\equiv\sigma_{T,c'''}(t''))$;
  - $\sigma_Q(\divideontimes)=\divideontimes$;
  - $\sigma_Q((\neg q))=(\neg\sigma_Q(q))$;
  - $\sigma_Q((q\Rightarrow q'))=(\sigma_Q(q)\Rightarrow\sigma_Q(q'))$;
  - $\sigma_Q((q\,\mathbf{U}\,q'))=(\sigma_Q(q)\,\mathbf{U}\,\sigma_Q(q'))$;
  - $\sigma_Q((q\,\mathbf{S}q'))=(\sigma_Q(q)\,\mathbf{S}\sigma_Q(q'))$;
  - $\sigma_Q((\forall y\ q))=(\forall y\ \sigma_Q(q))$;
  - $\sigma_Q(\alpha_z(t_1,\ldots,t_n))=\alpha_{\mathcal{Sg}(\sigma)_{ACC_{g_1\ldots g_n,g}}(z)(\sigma_{T,g_1}(t_1),\ldots,\sigma_{T,g_n}(t_n))}$
  - $\sigma_Q(\beta_{c',z}(t',t_1,\ldots,t_n))=\beta_{c',z}(\sigma_{T,c'}(t'),\sigma_{T,g_1}(t_1),\ldots,\sigma_{T,g_n}(t_n))$.
- $\sigma_L:L^c\rightarrow L^{c'}$ define-se do seguinte modo:
  - $\sigma_L(x.q)=x'.\ \sigma_Q(q[^x_{\sigma(x')}])$, em que x' é $\delta_{c'}(X_{c'}\backslash Var_{c'}(q))$ e $q[^x_{\sigma(x')}]$ é a fórmula esquema que se obtém de q substituindo todas as ocorrências livres de x por $\sigma(x')$.

□

Antes de introduzir a instituição propriamente dita, provam-se alguns resultados técnicos, úteis para estabelecer a condição de satisfação da instituição.

Lema 3.46

Sejam $\sigma:c'\rightarrow c$ um morfismo de classes, θ uma atribuição, $k\in\mathbb{N}_\circ$, $u'\in\mathcal{P}op_{c'}$ e $t\in T^c_g$. Então,

$$[\![t]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co} = [\![\sigma_{T,g}(t)]\!]^{\theta}_{k,u',co}.$$

prova:

A prova realiza-se por indução na estrutura do termo t.

- t é $y\in X_g$;

$[\![\sigma_{T,g}(y)]\!]^{\theta}_{k,u',co} = [\![y]\!]^{\theta}_{k,u',co}$ [tradução de termos]

$= \theta_g(y)$ [interpretação de termos]

$= [\![t]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}$ [interpretação de termos]

- t é $f(t_1,\ldots,t_n)$, em que $f \in OP_{g_1\ldots g_n,g}$ e $t_i \in T^c_{g_1\ldots g_n,g}$, para i=1,…,n

$[\![\sigma_{T,g}(f(t_1,\ldots,t_n))]\!]^{\theta}_{k,u',co}$

$= [\![f(\sigma_{T,g_1}(t_1), \ldots, \sigma_{T,g_n}(t_n))]\!]^{\theta}_{k,u',co}$ [tradução de termos]

$= f_{td}([\![\sigma_{T,g_1}(t_1)]\!]^{\theta}_{k,u',co},\ldots,[\![\sigma_{T,g_n}(t_n)]\!]^{\theta}_{k,u',co})$ [interpretação de termos]

$= f_{td}([\![t_1]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co})$ [hipótese de indução]

$= [\![f(t_1,\ldots,t_n)]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}$ [interpretação de termos]

- t é $z(t_1,\ldots,t_n)$, em que $z \in ACC^c_{g_1\ldots g_n,g}$ e $t_i \in T^c_{g_i}$, para i=1,…,n

$[\![\sigma_{T,g}(z(t_1,\ldots,t_n))]\!]^{\theta}_{k,u',co}$

$= [\![\mathcal{S}g(\sigma)_{ACC_{g_1\ldots g_n,g}}(z)(\sigma_{T,g_1}(t_1), \ldots, \sigma_{T,g_n}(t_n))]\!]^{\theta}_{k,u',co}$ [tradução de termos]

$= [\mathcal{S}g(\sigma)_{ACC_{g_1\ldots g_n,g}}(z)([\![\sigma_{T,g_1}(t_1)]\!]^{\theta}_{k,u',co},\ldots,[\![\sigma_{T,g_n}(t_n)]\!]^{\theta}_{k,u',co})]$

[interpretação de termos]

$= [\mathcal{S}g(\sigma)_{ACC_{g_1\ldots g_n,g}}(z)([\![t_1]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co})]$

[hipótese de indução]

$= [z([\![t_1]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co})]$ [relação $\approx_c$]

$= [\![z(t_1, \ldots, t_n)]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}$ [interpretação de termos]

- t é $a(t_1,\ldots,t_n)$, em que $a \in ATR^c_{g_1\ldots g_n,g}$ e $t_i \in T^c_{g_i}$, para i=1,…,n

$[\![\sigma_{T,g}(a(t_1,\ldots,t_n))]\!]^{\theta}_{k,u',co}$

$= [\![\mathcal{S}g(\sigma)_{ATR_{g_1\ldots g_n,g}}(a)(\sigma_{T,g_1}(t_1), \ldots, \sigma_{T,g_1}(t_n))]\!]^{\theta}_{k,u',co}$

[tradução de termos]

$= p\,\mathcal{S}g(\sigma)_{ATR_{g_1\ldots g_n,g}}(a)([\![\sigma_{T,g_1}(t_1)]\!]^{\theta}_{k,u',co},\ldots,[\![\sigma_{T,g_n}(t_n)]\!]^{\theta}_{k,u',co})(\Lambda(\langle c',u'\rangle)_{2,k})$

[interpretação de termos]

$= p_a([\![\sigma_{T,g_1}(t_1)]\!]^{\theta}_{k,u',co},\ldots,[\![\sigma_{T,g_n}(t_n)]\!]^{\theta}_{k,u',co})(\Lambda(\langle c,\mathcal{P}op(\sigma)(u')\rangle)_{2,k})$

[imagem de situação]

$= p_a([\![t_1]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co})(\Lambda(\langle c,\mathcal{P}op(\sigma)(u')\rangle)_{2,k})$

[hipótese de indução]

$= [\![a(t_1, \ldots, t_n)]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}$ [interpretação de termos]

- t é $\mu(t_1)$, em que $t_1 \in T^c_g$

$$\begin{aligned}
[\![\sigma_{T,g'}(\mu(t_1))]\!]^{\theta}_{k,u',co} &= [\![\mu(\sigma_{T,g}(t_1))]\!]^{\theta}_{k,u',co} && \text{[tradução de termos]}\\
&= \mathcal{P}op(\mu)([\![\sigma_{T,g}(t_1)]\!]^{\theta}_{k,u',co}) && \text{[interpretação de termos]}\\
&= \mathcal{P}op(\mu)([\![t_1]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}) && \text{[hipótese de indução]}\\
&= [\![\mu(t_1)]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co} && \text{[interpretação de termos]}
\end{aligned}$$

□

Lema 3.47

Sejam $\sigma:c'\rightarrow c$ um morfismo de classes, θ uma atribuição, $k\in \mathbb{N}_\circ$, $u'\in \mathcal{P}op_{c'}$ e $q\in Q^c$. Então,

$$\theta,k,\mathcal{P}op(\sigma)(u'),\Omega,co \Vdash_c q \text{ sse } \theta,k,u',\Omega,co\Vdash_{c'} \sigma_Q(q).$$

prova:

A prova realiza-se por indução na estrutura da fórmula q.

- q é $t\in T^c_{bool}$

  $\theta,k,\mathcal{P}op(\sigma)(u'),\Omega,co \Vdash_c t$

  sse $[\![t]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co} = true_{td}$ [relação de satisfação]

  sse $[\![\sigma_{T,bool}(t)]\!]^{\theta}_{k,u',co} = true_{td}$ [lema **3.46**]

  sse $\theta,k,u',\Omega,co\Vdash_{c'} \sigma_{T,bool}(t)$ [relação de satisfação]

  sse $\theta,k,u',\Omega,co\Vdash_{c'} \sigma_Q(t)$ [tradução de fórmula esquema]

- q é (t=t'), em que $t,t'\in T^c_g$

  $\theta,k,\mathcal{P}op(\sigma)(u'),\Omega,co \Vdash_c (t=t')$

  sse $[\![t]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co} = [\![t']\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}$ [relação de satisfação]

  sse $[\![\sigma_{T,g}(t)]\!]^{\theta}_{k,u',co} = [\![\sigma_{T,g}(t')]\!]^{\theta}_{k,u',co}$ [lema **3.46**]

  sse $\theta,k,u',\Omega,co\Vdash_{c'} (\sigma_{T,g}(t)=\sigma_{T,g}(t'))$ [relação de satisfação]

  sse $\theta,k,u',\Omega,co\Vdash_{c'} \sigma_Q((t=t'))$ [tradução de fórmula esquema]

- q é $(t \equiv t')$, com $t \in T^c_{c''}$ e $t' \in T^c_{c'''}$

  $\theta,k,\mathcal{P}op(\sigma)(u'),\Omega,co \Vdash_c (t\equiv t')$

  sse $[\![t]\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}=[\![t']\!]^{\theta}_{k,\mathcal{P}op(\sigma)(u'),co}$ e $c''^{\top}=c'''^{\top}$ [relação de satisfação]

  sse $[\![\sigma_{T,c''}(t)]\!]^{\theta}_{k,u',co}=[\![\sigma_{T,c'''}(t')]\!]^{\theta}_{k,u',co}$ e $c''^{\top}=c'''^{\top}$ [lema **3.46**]

  sse $\theta,k,u',\Omega,co\Vdash_{c'} (\sigma_{T,c''}(t) \equiv \sigma_{T,c'''}(t'))$ [relação de satisfação]

  sse $\theta,k,u',\Omega,co\Vdash_{c'} \sigma_Q((t \equiv t'))$ [tradução de fórmula esquema]

- q é ❋

$\theta,k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c \divideontimes$

sse $k=0$ [relação de satisfação]

sse $\theta,k,u',\Omega,co \Vdash_{c'} \divideontimes$ [relação de satisfação]

sse $\theta,k,u',\Omega,co \Vdash_{c'} \sigma_Q(\divideontimes)$ [tradução de fórmula esquema]

- q é $(\neg q')$

$\theta,k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c (\neg q')$

sse não $\theta,k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c q'$ [relação de satisfação]

sse não $\theta,k,u',\Omega,co \Vdash_{c'} \sigma_Q(q')$ [hipótese de indução]

sse $\theta,k,u',\Omega,co \Vdash_{c'} (\neg \sigma_Q(q'))$ [relação de satisfação]

sse $\theta,k,u',\Omega,co \Vdash_{c'} \sigma_Q((\neg q'))$ [tradução de fórmula esquema]

A prova é semelhante para os casos em que q for $(q' \Rightarrow q'')$, $(q' \mathbf{U} q'')$ ou $(q' \mathbf{S} q'')$.

- q é $(\forall y\ q')$

$\theta,k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c (\forall y\ q')$

sse $\theta',k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c q'$, para qualquer θ', y-equivalente a θ

[relação de satisfação]

sse $\theta',k,u',\Omega,co \Vdash_{c'} \sigma_Q(q')$, para qualquer θ', y-equivalente a θ

[hipótese de indução]

sse $\theta,k,u',\Omega,co \Vdash_{c'} (\forall y\ \sigma_Q(q'))$ [relação de satisfação]

sse $\theta,k,u',\Omega,co \Vdash_{c'} (\sigma_Q(\forall y\ q'))$ [tradução de fórmula esquema]

- q é $\alpha_z(t_1,\ldots,t_n)$

$\theta,k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c \alpha_z(t_1,\ldots,t_n)$

sse $p_{\alpha_z}([\![t_1]\!]^{\theta}_{k,\mathcal{Pop}(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{Pop}(\sigma)(u'),co})(\Omega(<c,\mathcal{Pop}(\sigma)(u')>)_{2,k})=true$ td

[relação de satisfação]

sse $p_{\alpha_z}([\![t_1]\!]^{\theta}_{k,\mathcal{Pop}(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{Pop}(\sigma)(u'),co})(\sigma(\Omega(<c',u')>)_{2,k}))=true$ td

sse $p_{\alpha_{\mathcal{Sg}(\sigma)ACC_{g_1\ldots g_n,g}(z)}}([\![t_1]\!]^{\theta}_{k,\mathcal{Pop}(\sigma)(u'),co},\ldots,[\![t_n]\!]^{\theta}_{k,\mathcal{Pop}(\sigma)(u'),co})(\Omega(<c',u')>)_{2,k})=true$ td

[imagem de contexto]

sse $p_{\alpha_{\mathcal{Sg}(\sigma)ACC_{g_1\ldots g_n,g}(z)}}([\![\sigma_{T,g_1}(t_1)]\!]^{\theta}_{k,u',co},\ldots,[\![\sigma_{T,g_n}(t_n)]\!]^{\theta}_{k,u',co})(\Omega(<c',u')>)_{2,k})=true$ td

[lema **3.46**]

sse $\theta,k,u',\Omega,co \Vdash_{c'} \alpha_{\mathcal{Sg}(\sigma)ACC_{g_1\ldots g_n,g}(z)}(\sigma_{T,g_1}(t_1),\ldots,\sigma_{T,g_n}(t_n))$

[relação de satisfação]

sse $\theta,k,u', \Omega, co \Vdash_{c'} \sigma_Q(\alpha_z(t_1,\ldots,t_n))$ [imagem de fórmula esquema]

A prova é semelhante para o caso em que q é $\beta_{c'',z}(t'',t_1,\ldots,t_n)$. □

Os resultados que se seguem são adaptações para GCL de resultados conhecidos para a lógica de primeira ordem (cf. por exemplo [Hamilton 78]).

Lema 3.48

Sejam c uma classe, $q \in Q^c$, $u \in \mathcal{P}op_c$, $x \in X_c$ e $t \in T_g^c$ livre para x em q [1]. Sejam ainda θ e θ' duas atribuições x-equivalentes tais que $\theta'_c(x) = [\![t]\!]^{\theta}_{k,u,co}$. Então $\theta,k,u, \Omega, co \Vdash_c q[^x_t]$ sse $\theta',k,u, \Omega, co \Vdash_c q$.

prova:

Antes de provar o resultado enunciado, prova-se primeiro o seguinte resultado sobre termos:

$[\![t'[^x_t]]\!]^{\theta}_{k,u,co} = [\![t']\!]^{\theta'}_{k,u,co}$, qualquer que seja o termo $t' \in T^c_{g'}$.

A prova deste resultado faz-se por indução na estrutura do termo t'.

- t' é $y \in X_g$

  Suponha-se que y=x. Então $y[^x_t] = t$, logo $[\![y[^x_t]]\!]^{\theta}_{k,u,co} = [\![t]\!]^{\theta}_{k,u,co}$. Por outro lado,

  $[\![y]\!]^{\theta'}_{k,u,co} = \theta'_c(y)$ [recorde-se que x=y, logo $\theta'_c(x) = \theta'_c(y)$]

  $= [\![t]\!]^{\theta}_{k,u,co}$ [por hipótese sobre θ']

  donde se conclui que $[\![y[^x_t]]\!]^{\theta}_{k,u,co} = [\![y]\!]^{\theta'}_{k,u,co}$.

  Suponha-se agora que $y \neq x$. Então, $y[^x_t] = y$, logo

  $[\![y[^x_t]]\!]^{\theta}_{k,u,co} = [\![y]\!]^{\theta}_{k,u,co}$

  $= \theta_g(y)$

  $= \theta'_g(y)$ [$y \neq x$, e θ e θ' são x-equivalentes]

  $= [\![y]\!]^{\theta'}_{k,u,co}$

- t' é $f(t_1,\ldots,t_n)$, em que $f \in OP_{g_1\ldots g_n,g}$ e $t_i \in T^c_{g_i}$, para i=1,…,n

  Como $f(t_1,\ldots,t_n)[^x_t] = f(t_1[^x_t],\ldots,t_n[^x_t])$, então

  $[\![f(t_1,\ldots,t_n)[^x_t]]\!]^{\theta}_{k,u,co} = [\![f(t_1[^x_t],\ldots,t_n[^x_t])]\!]^{\theta}_{k,u,co}$

  $= f_{td}([\![t_1[^x_t]]\!]^{\theta}_{k,u,co},\ldots,[\![t_n[^x_t]]\!]^{\theta}_{k,u,co})$

  $= f_{td}([\![t_1]\!]^{\theta'}_{k,u,co},\ldots,[\![t_n]\!]^{\theta'}_{k,u,co})$ [hipótese de indução]

---

[1] Um termo t diz-se *livre para uma variável x* numa fórmula q sse x não ocorrer livre no âmbito de um quantificador $(\forall y)$, em que y é uma variável de t.

$$= [\![f(t_1,\ldots,t_n)]\!]^{\theta'}_{k,u,co}$$

A prova para os outros termos ($z(t_1,\ldots,t_n)$, $a(t_1,\ldots t_n)$ e μ (t'')) é semelhante e resulta da aplicação directa da hipótese de indução.

Pode-se agora provar o resultado enunciado. Esta prova realiza-se por indução na estrutura na fórmula q.

- q é $t' \in T^c_{bool}$

  $\theta',k,u, \Omega,co \Vdash_c t'$ sse $[\![t']\!]^{\theta'}_{k,u,co}=true_{td}$

  sse $[\![t'[^x_t]]\!]^{\theta}_{k,u,co}=true_{td}$ [pelo resultado anterior]

  sse $\theta,k,u, \Omega,co \Vdash_c t'[^x_t]$

A prova nos casos (t'=t''), ($t' \equiv t''$), $\alpha_z(t_1,\ldots,t_n)$ e $\beta_{c'',z}(t'',t_1,\ldots,t_n)$ é semelhante (atendendo a que $\alpha_z(t_1,\ldots,t_n)[^x_t]=\alpha_z(t_1[^x_t],\ldots,t_n[^x_t])$ e $\beta_{c'',z}(t'',t_1,\ldots,t_n)[^x_t]=\beta_{c'',z}(t''[^x_t], t_1[^x_t],\ldots,t_n[^x_t])$).

- q é $(\neg q')$

  $\theta',k,u, \Omega,co \Vdash_c (\neg q')$ sse não $\theta',k,u,\Omega,co \Vdash_c q'$

  sse não $\theta,k,u,\Omega, co \Vdash_c q'[^x_t]$ [hipótese de indução]

  sse $\theta,k,u, \Omega,co \Vdash_c (\neg q'[^x_t])$

  sse $\theta,k,u, \Omega,co \Vdash_c (\neg q')[^x_t]$ [definição de substituição]

  A prova para os casos ($q' \Rightarrow q''$), ($q' \mathbf{U} q''$) e ($q' \mathbf{S} q''$) é semelhante.

- q é $(\forall y\, q')$

Suponha-se que $x \neq y$ e que não se verifica $\theta,k,u,\Omega, co \Vdash_c (\forall y\, q'[^x_t])$. Então existe uma atribuição η, y-equivalente a θ, tal que $\eta,k,u, \Omega,co \Vdash_c q'[^x_t]$ não se verifica. Seja η' uma atribuição x-equivalente a η, tal que $\eta'_c(x)=[\![t]\!]^{\eta}_{k,u,co}$. Então, por hipótese de indução, $\eta',k,u, \Omega,co \Vdash_c q'$ não se verifica. Note-se agora que t é livre para x em $(\forall y\, q')$, logo y não ocorre em t, o que implica que a avaliação de t depende apenas dos valores dados por θ às variáveis diferentes de y. Como $\theta_g(x')=\eta_g(x')$ qualquer que seja a variável $x' \neq y$ então $[\![t]\!]^{\theta}_{k,u,co}=[\![t]\!]^{\eta}_{k,u,co}$ [1]. Então, para $x' \neq x$ e $x' \neq y$,

$\theta'_g(x') = \theta_g(x')$ [θ e θ' são x-equivalentes]

$= \eta_g(x')$ [θ e η são y-equivalentes]

$= \eta'_g(x')$ [η e η' são x-equivalentes]

e,

---

[1] *Lema dos símbolos omissos* ([Hamilton 78] prova um resultado semelhante para a lógica de 1ª ordem).

$\theta'_c(x) = [\![t]\!]^{\theta}_{k,u,co} = [\![t]\!]^{\eta}_{k,u,co} = \eta'_c(x)$

o que permite concluir que θ' e η' são y-equivalentes. Como $\eta',k,u, \Omega,co \Vdash_c q'$ não se verifica conclui-se que $\theta',k,u, \Omega,co \Vdash_c (\forall y\ q')$ também não se verifica.

Para provar o recíproco, suponha-se que $\theta',k,u,\Omega,co \Vdash_c (\forall y\ q')$, então

$\eta',k,u, \Omega,co \Vdash_c q'$, qualquer que seja η', y-equivalente a θ'. *(1)*

Seja η uma atribuição qualquer, y-equivalente a θ, e defina-se a seguinte atribuição η'', x-equivalente a η, tal que $\eta''_c(x)=[\![t]\!]^{\eta}_{k,u,co}$. Tal como atrás, pelo facto de t ser livre para x em $(\forall y\ q')$ e portanto y não ocorrer em t, sabe-se que $[\![t]\!]^{\theta}_{k,u,co} = [\![t]\!]^{\eta}_{k,u,co}$, e que η'' e θ' são y-equivalentes. Então, da condição (1), conclui-se que $\eta'',k,u, \Omega,co \Vdash_c q'$. Pela hipótese de indução, $\eta,k,u, \Omega,co \Vdash_c q'[^x_t]$. Como η é uma atribuição qualquer, y-equivalente a θ, conclui-se que $\theta,k,u,\Omega, co \Vdash_c (\forall y\ q'[^x_t])$.

Suponha-se agora que x=y. Então $(\forall y\ q')[^x_t]=(\forall y\ q')$. Por outro lado, qualquer atribuição y-equivalente a θ é também y-equivalente a θ', pois x=y. Então

$\theta,k,u, \Omega,co \Vdash_c (\forall y\ q')[^x_t]$

sse $\theta,k,u, \Omega,co \Vdash_c (\forall y\ q')$

sse $\eta,k,u, \Omega,co \Vdash_c q'$, qualquer que seja a atribuição η, y-equivalente a θ

sse $\eta,k,u, \Omega,co \Vdash_c q'$, qualquer que seja a atribuição η, y-equivalente a θ'

sse $\theta',k,u, \Omega,co \Vdash_c (\forall y\ q')$. □

Lema 3.49

Sejam $\sigma:c'\to c$, $q \in Q^c$, $u \in \mathcal{P}op_c$, $x \in X_c$ e $x' \in X_{c'}$ tal que x' não ocorre em q. Sejam ainda θ e θ' duas atribuições x'-equivalentes tais que $\theta'_c(x)=[\![\sigma(x')]\!]^{\theta'}_{k,u,co}$. Então $\theta,k,u, \Omega,co \Vdash_c q$ sse $\theta',k,u, \Omega,co \Vdash_c q[^x_{\sigma(x')}]$.

prova:

Tal como na prova do lema anterior, antes de provar o resultado enunciado é necessário provar o seguinte resultado auxiliar sobre termos:

$[\![t]\!]^{\theta}_{k,u,co} = [\![t\ [^x_{\sigma(x')}]]\!]^{\theta'}_{k,u,co}$, qualquer que seja o termo $t \in T^c_g$.

Este resultado prova-se por indução na estrutura do termo t, e a prova é semelhante a realizada para o lema **3.48** .

A prova do resultado enunciado é feita por indução na estrutura da fórmula esquema q, e é semelhante à realizada para o lema anterior, excepto no caso das fórmulas esquema $(\forall y\ q')$. Considere-se então que q é $(\forall y\ q')$. Suponha-se que $x \neq y$ e que não se verifica $\theta',k,u, \Omega,co \Vdash_c (\forall y\ q'[^x_{\sigma(x')}])$. Então, existe uma atribuição η', y-equivalente a θ', tal que $\eta',k,u, \Omega,co \Vdash_c q'[^x_{\sigma(x')}]$ não se verifica. Seja η uma atribuição x'-equivalente a η' tal que $\eta_{c'}(x')= \theta'_{c'}(x')$. Note-se que

$\eta'_{c'}(x) = \theta'_{c'}(x)$ [θ' e η ' são y-equivalentes e x≠y]

$= [\![\sigma(x')]\!]^{\theta'}_{k,u,co}$ [hipótese sobre θ ']

$= \mathcal{P}op(\sigma)(\theta'_{c'}(x'))$

$= \mathcal{P}op(\sigma)(\eta'_{c'}(x'))$ [θ' e η ' são y-equivalentes e x'≠y]

$= [\![\sigma(x')]\!]^{\eta'}_{k,u,co}$

logo, por hipótese de indução, $\eta,k,u,\Omega, co \Vdash_c q'$ não se verifica. Por outro lado, como para y'≠x' e y'≠y

$\eta_g(y') = \eta'_g(y')$ [η e η' são x'-equivalentes]

$= \theta'_g(y')$ [θ' e η ' são y-equivalentes]

$= \theta_g(y')$ [θ e θ' são x'-equivalentes]

e, por hipótese sobre η, η(x') = θ(x'), logo conclui-se que θ e η são y-equivalentes. Então, $\theta,k,u, \Omega,co \Vdash_c (\forall\, y\, q')$ não se verifica.

Para provar o recíproco, suponha-se que $\theta',k,u,\Omega,co \Vdash_c (\forall\, y\, q'[^{x}_{\sigma(x')}])$. Então,

$\eta',k,u, \Omega,co \Vdash_c q'[^{x}_{\sigma(x')}]$, qualquer que seja η', y-equivalente a θ'. *(1)*

Seja η uma atribuição qualquer, y-equivalente a θ. Defina-se η'', x'-equivalente a η, tal que $\eta''_{c'}(x')=\theta'_{c'}(x')$. Note-se que, para y'≠y e y'≠x',

$\eta''_g(y') = \eta_g(y')$ [η e η'' são x'-equivalentes]

$= \theta_g(y')$ [θ e η são y-equivalentes]

$= \theta'_g(y')$ [θ e θ' são x'-equivalentes]

logo, θ' e η'' são y-equivalentes. Pela condição (1), sabe-se que $\eta'',k,u, \Omega,co \Vdash_c q'[^{x}_{\sigma(x')}]$. Agora,

$[\![\sigma(x')]\!]^{\eta''}_{k,u,co} = \mathcal{P}op(\sigma)(\eta''_{c'}(x'))$

$= \mathcal{P}op(\sigma)(\theta'_{c'}(x'))$ [hipótese sobre η '']

$= [\![\sigma(x')]\!]^{\theta'}_{k,u,co}$

$= \theta'_c(x)$ [hipótese sobre θ ']

$= \eta''_c(x)$ [θ' e η '' são y-equivalentes]

logo, por hipótese de indução, $\eta,k,u,\Omega, co \Vdash_c q'$. Como η é uma atribuição qualquer, y-equivalente a θ, conclui-se que $\theta,k,u, \Omega,co \Vdash_c (\forall\, y\, q')$.

A prova no caso em que x=y é semelhante à apresentada para o lema anterior. □

Lema 3.50

Sejam $\sigma:c'\to c$ um morfismo de classes, $u'\in \mathcal{P}op_{c'}$, $q\in Q^c$, $x\in X_c$, $x'=\delta_{c'}(X_{c'}\backslash Var_{c'}(q))$.

Então,

$\theta,k,\mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c x.q$, quaisquer que sejam a atribuição θ e $k \in \mathbb{N}_\circ$,

sse

$\theta,k,u', \Omega,co \Vdash_{c'} x'.\ \sigma_Q(q[^x_{\sigma(x')}])$, quaisquer que sejam a atribuição θ e $k \in \mathbb{N}_\circ$.

prova:

($\Rightarrow$) Sejam θ uma atribuição e $k \in \mathbb{N}_\circ$. Há duas situações a considerar: (i) $\theta_{c'}(x')\neq u'$; (ii) $\theta_{c'}(x')=u'$. No primeiro caso, resulta trivialmente da definição de satisfação de fórmula, que $\theta,k,u', \Omega,co \Vdash_{c'} x'.\ \sigma_Q(q[^x_{\sigma(x')}])$. Considere-se então a segunda situação. É necessário provar que

$$\theta,k,u', \Omega,co \Vdash_{c'} \sigma_Q(q[^x_{\sigma(x')}]).$$

Seja θ' a atribuição x-equivalente a θ tal que $\theta'_c(x)=\mathcal{Pop}(\sigma)(u')$. Por hipótese, $\theta',k,\mathcal{Pop}(\sigma)(u'),\Omega, co \Vdash_c x.q$, e como $\theta'_c(x)=\mathcal{Pop}(\sigma)(u')$, da definição de satisfação de fórmula, resulta que $\theta',k,\mathcal{Pop}(\sigma)(u'), \Omega,co \Vdash_c q$. Por outro lado,

$$\begin{aligned}
[\![\sigma(x')]\!]^{\theta}_{k,u,co} &= \mathcal{Pop}(\sigma)([\![x']\!]^{\theta}_{k,u,co}) \\
&= \mathcal{Pop}(\sigma)(\theta_{c'}(x')) \\
&= \mathcal{Pop}(\sigma)(u') && \text{[hipótese (ii)]} \\
&= \theta'_c(x) && \text{[definição de } \theta'\text{]}
\end{aligned}$$

Então, do lema **3.48** , resulta $\theta,k,\ \mathcal{Pop}(\sigma)(u'),\Omega ,co \Vdash_c q[^x_{\sigma(x')}]$, e pelo lema **3.47** obtém-se $\theta,k,u', \Omega,co \Vdash_{c'} \sigma_Q(q[^x_{\sigma(x')}])$.

($\Leftarrow$) Sejam θ uma atribuição e $k \in \mathbb{N}_\circ$. Há duas situações a considerar: (i) $\theta_c(x)\neq\mathcal{Pop}(\sigma)(u')$; (ii) $\theta_c(x)= \mathcal{Pop}(\sigma)(u')$. No primeiro caso resulta trivialmente da definição de satisfação de fórmula, que $\theta,k, \mathcal{Pop}(\sigma)(u'),\Omega, co \Vdash_c x.q$. Considere-se então a segunda situação. É necessário provar que

$$\theta,k, \mathcal{Pop}(\sigma)(u'),\Omega,co \Vdash_c q.$$

Seja θ' a atribuição x'-equivalente a θ, tal que $\theta'_{c'}(x')=u'$. Por hipótese, $\theta',k,u', \Omega,co \Vdash_{c'} x'.\sigma_Q(q[^x_{\sigma(x')}])$, e da definição de satisfação de fórmula, atendendo a que $\theta'_{c'}(x')=u'$, resulta que $\theta',k,u', \Omega,co \Vdash_{c'} \sigma_Q(q[^x_{\sigma(x')}])$. Do lema **3.47** , resulta que $\theta',k,\mathcal{Pop}(\sigma)(u'),\Omega, co \Vdash_c q[^x_{\sigma(x')}]$. Por outro lado,

$$\begin{aligned}
[\![\sigma(x')]\!]^{\theta'}_{k,u,co} &= \mathcal{Pop}(\sigma)([\![x']\!]^{\theta'}_{k,u,co}) \\
&= \mathcal{Pop}(\sigma)(\theta'_{c'}(x')) \\
&= \mathcal{Pop}(\sigma)(u') && \text{[definição de } \theta'\text{]} \\
&= \theta_c(x) && \text{[hipótese (ii)]} \\
&= \theta'_c(x) && \text{[x'-equivalência entre } \theta \text{ e } \theta'\text{]}
\end{aligned}$$

Então, pelo lema **3.49** , conclui-se que $\theta,k,\mathcal{P}op(\sigma)(u'),\Omega,co \Vdash_c q$. □

Com base estes resultados, apresenta-se a instituição induzida por uma comunidade de objectos.

Teorema 3.51 *Instituição induzida por comunidade de objectos*

A assinatura Σ, a família X e a comunidade de objectos *co* induzem a *instituição* $I_{\Sigma,X,co}=\langle \mathcal{C}_N^{op}, \mathcal{S}en, Int, \Vdash \rangle$

- $\mathcal{S}en(c)=L^c$;
- $\mathcal{S}en(\sigma:c\rightarrow c')=\sigma_L:L^c\rightarrow L^{c'}$;
- $Int(c)=\mathcal{P}op_c$;
- $Int(\sigma:c\rightarrow c')=\mathcal{P}op(\sigma):\mathcal{P}op_{c'}\rightarrow\mathcal{P}op_c$;
- $u \Vdash_c \varphi$ sse $\theta,k,u,\Omega, co \Vdash_c \varphi$, quaisquer que sejam a atribuição θ, $k \in \mathbb{N}_0$ e o contexto de comunidade Ω .

prova:

Para mostrar que $I_{\Sigma,X,co}$ é de facto uma instituição basta mostrar que a relação $\Vdash$ verifica a condição de satisfação, i.e., $u' \Vdash_{c'} \sigma_L(\varphi)$ sse $\mathcal{P}op(\sigma)(u') \Vdash_c \varphi$, quaisquer que sejam o morfismo de classes $\sigma:c'\rightarrow c$ (em $\mathcal{C}_N$), a instância $u'\in \mathcal{P}op_{c'}$ e a fórmula $\varphi\in L^c$, que resulta directamente do lema **3.50** . □

3.5 Especificações e Teorias

Introduz-se agora a noção de especificação e de teoria. Em geral, pretende-se que uma especificação descreva o comportamento de dado um sistema. Neste contexto, uma teoria surge como o conjunto de fórmulas que é verdadeiro para esse comportamento. A linguagem de especificação deverá ainda permitir *compor* diversas teorias de modo a obter teorias mais complexas ([Goguen e Burstall 92]).

Definição 3.52 *Consequência semântica*

Sejam $\Psi \subseteq L^c$ e $\varphi\in L^c$. Diz-se que φ é *consequência semântica* de Ψ na classe c, o que se denota por $\Psi\models_c\varphi$, sse para toda a comunidade de objectos *co*, se $co \Vdash_c \psi$, qualquer que seja $\psi\in\Psi$, então $co \Vdash_c \varphi$. □

Definição 3.53 *Fecho de conjunto por consequência semântica*

Seja $\Psi \subseteq L^c$. O *fecho por consequência semântica* de Ψ relativamente à classe c é o conjunto $\Psi^{\models c}=\{\psi: \Psi\models_c\psi\}$. □

Definição 3.54 *Especificação de classe*

Uma *especificação de classe* sobre Σ é um par $<c,\Phi>$, tal que $c \in |\mathcal{C}_N|$ e $\Phi \subseteq L^c$. Uma *teoria de classe* sobre Σ é uma especificação de classe $<c,\Phi>$ tal que $\Phi^{\vdash c}=\Phi$. □

A uma especificação também se costuma chamar apresentação de teoria. O conjunto de fórmulas apresentado no exemplo **3.13** juntamente com a classe *Relógio* formam uma especificação de classe.

Definição 3.55 *Morfismo de teorias de classe*

Um *morfismo de teorias de classe* sobre Σ

$$\sigma:<c,\Phi>\rightarrow<c',\Phi'>$$

é um morfismo de classes $\sigma:c'\rightarrow c$ tal que $\sigma_L(\varphi)\in\Phi'$, qualquer que seja $\varphi\in\Phi$. □

Uma das propriedades que se deseja numa instituição, para que esta seja útil na definição da semântica de uma linguagem de especificação, é que a categoria das teorias seja cocartesiana. É esta propriedade que permite combinar teorias de modo a construir novas teorias.

Proposição 3.56 *Categoria das teorias de classe*

As teorias de classe e respectivos morfismos sobre Σ constituem uma categoria, $\mathcal{TeoC}_\Sigma$. Esta categoria é cocartesiana. □

A prova deste resultado para OSL encontra-se em [Sernadas e Sernadas 94].

Definição 3.57 *Satisfação de família de fórmulas em comunidade*

Seja $A=\{A^c\}_{c\in|\mathcal{C}_N|}$, em que $A^c \subseteq L^c$. Então, diz-se que a comunidade *co* satisfaz A, o que se denota por $co\models A$, sse $co\models_c \psi$, qualquer que seja $\psi\in A^c$. □

Definição 3.58 *Fecho de família por consequência*

Seja $A=\{A^c\}_{c\in|\mathcal{C}_N|}$, em que $A^c \subseteq L^c$. O *fecho por consequência* de A é a família $A^{\models}=\{A^{\models c}\}_{c\in|\mathcal{C}_N|}$, tal que

$$A^{\models c}=\{\psi\in L^c: \text{se } co\models A \text{ então } co\models_c \psi\}.$$ □

Convém no entanto salientar que $A^{\models c}$ e $A^{c\models c}$ (definição **3.53**) não devem ser confundidos. Em geral $A^{c\models c}$ está contido em $A^{\models c}$.

Definição 3.59 *Especificação de comunidade de objectos*

Uma *especificação de comunidade de objectos* é um par $eco=<\Sigma,Ax>$, em que:

- $\Sigma=<N, \mathcal{Sg}>$ é uma assinatura de comunidade de objectos;
- $Ax=\{Ax^c\}_{c\in|\mathcal{C}_N|}$, em que cada $Ax^c \subseteq L^c$.

Uma *teoria de comunidade de objectos* é uma especificação de comunidade de objectos *eco* $=<\Sigma,Ax>$ tal que $Ax^{\vdash}=Ax$. □

Ainda no contexto de uma especificação de comunidade de objectos, define-se uma relação de consequência, necessária no próximo capítulo para provar a correcção do sistema dedutivo.

Definição 3.60 *Consequência semântica em especificação de comunidade*

Seja *eco* $=<\Sigma,Ax>$ uma especificação de comunidade de objectos, $H=\{H^c\}_{c\in|\mathcal{C}_N|}$ em que $H^c\subseteq L^c$ e $\varphi\in L^c$. Então φ *diz-se consequência de H em eco*, o que se denota por $H\models_c^{eco}\varphi$, sse $\varphi\in (A\cup H)^{\vdash c}$. □

Definição 3.61 *Fecho por consequência semântica*

Seja *eco* $=<\Sigma,Ax>$ uma especificação de comunidade de objectos. O *fecho por consequência semântica* de *eco* é $eco^{\vdash}=<\Sigma,Ax^{\vdash}>$. □

Teorema 3.62 *Functor induzido por especificação de comunidade*

Uma especificação de comunidade de objectos *eco*$=<\Sigma,Ax>$ induz o seguinte functor $\mathcal{Th}:\mathcal{C}_N{}^{op}\rightarrow\mathcal{Teo}\mathcal{C}_\Sigma$ tal que:

- $\mathcal{Th}(c)_2=Ax^{\vdash c}$;
- $\mathcal{Th}(\sigma)=\sigma_L$. □

Observação 3.63 *Axiomas de interacção*

Na especificação de produtos de classes, $c_1\otimes c_2$, é típico incluir três tipos de axiomas de interacção: (i) axiomas de chamada, (ii) axiomas de exclusão e (iii) axiomas sobre o ambiente.

(i) axiomas de chamada

- $(cond_1 \Rightarrow ((x.\nabla_1=z_1(t)) \Rightarrow (x.\nabla_2=z_2(t'))))$

  em que $cond_1$ é uma expressão booleana não envolvendo símbolos de c_2, z_1 é um símbolo de acção de c_1 e z_2 é um símbolo de acção externa de c_2.

(ii) axiomas de exlusão

- $(cond_1 \Rightarrow ((x.\nabla_1=z_1(t)) \Rightarrow (\neg(x.\nabla_2=z_2(t')))))$

  em que $cond_1$ é uma expressão booleana não envolvendo símbolos de c_2, z_1 é um símbolo de acção de c_1 e z_2 é um símbolo de acção externa de c_2.

(iii) axiomas sobre o ambiente

- $(\text{cond}_1 \Rightarrow (x.\beta_{c_2,z}(t',t) \Leftrightarrow \text{cond}_2))$

  em que cond_1 é uma expressão booleana não envolvendo símbolos de c_2, e cond_2 é uma expressão booleana não envolvendo símbolos de c_1 nem símbolos de predicado α_{z_2}. □

Existem outros tipos de axiomas de interacção. Em particular, existem outros tipos de axiomas de chamada, mais específicos da axiomatização que se pretende para linguagem GNOME, e que serão ilustrados na próxima secção.

Os axiomas sobre o ambiente pretendem dar uma descrição dos predicados relativos ao ambiente. A sua descrição será mais detalhada na próxima secção.

Recorde-se o exemplo **3.15** . A fórmula *(1)* é um axioma de chamada, e a fórmula *(3)* é um axioma sobre o ambiente.

3.6 Semântica da linguagem GNOME

Pretende-se neste capítulo definir uma axiomatização em GCL para o fragmento da linguagem GNOME considerado. Suponha-se então fixada uma especificação GNOME, *spec* .

Para manter informação sobre a identificação das instâncias de cada uma das classes, é necessário considerar definidas meta-classes, mc_c, uma para cada classe topo. Estas meta-classes vão ser responsáveis pelas gestão das identificações das instâncias de cada uma das classes.

1º passo *Rede de herança*

A especificação *spec* induz a seguinte rede de herança linear $N=<CP \cup \{mc_c : c \in CP^{\top}\}, \leq \cup \{(mc_c, mc_c) : c \in CP^{\top}\}>$:

- CP é o conjunto formado por todos os identificadores de classe que apareçam em *spec* , admitindo, sem perda de generalidade, que $mc_c \notin CP$, para $c \in CP^{\top}$;
- $\leq$ é uma ordem parcial linear à direita induzida pela relação de subclasse (*sub* ou *mon sub*) da linguagem GNOME. □

Exemplo 3.64 *Especificação relógios*

Recorde-se a especificação do problema dos relógios, já enriquecida com a classe dos relógios com defeito, apresentada na fig. 2.2, do capítulo anterior. Esta especificação induz a seguinte rede de herança $N_{relógios} = <CP \cup \{mc_{\text{Relógio}}, mc_{\text{Utilizador}}\}, \leq \cup \{(mc_{\text{Relógio}}, mc_{\text{Relógio}}), (mc_{\text{Utilizador}}, mc_{\text{Utilizador}})\}>$ em que:

- CP={Utilizador, Relógio, Drelógio};
- ≤={(Drelógio,Relógio)}$^{\bullet}$.[1] □

2º passo *Assinatura das meta-classes*

A assinatura de cada meta-classe mc_c, é definida do seguinte modo, para cada $c \in CP^{\top}$: $<mc_c, ACC^{mc_c}, ACI^{mc_c}, ATR^{mc_c}>$ tal que $ACC^{mc_c}_{\varepsilon,\tau_{mc_c}} = \{\perp\}$, $ACC^{mc_c}_{c,\tau_{mc_c}} = \{\text{inc}\}$, $ACI^{mc_c} = \{\perp\}$, $ATR^{mc_c}_{\varepsilon,\text{nat}} = \{\nu\}$.

Tal como na definição da semântica operacional ([Sernadas e Ramos 94], [Ramos e Sernadas 94]) o atributo ν denota a identificação da próxima instância a ser criada na classe c. A operação inc_c mantém actualizado esse valor, e deverá ser chamada sempre que ocorra uma acção (serviço) de nascimento.

3º passo *Assinatura de classe*

A especificação *spec* induz para cada classe $c \in CP$ uma assinatura de classe $<c, ACC^c, ACI^c, ATR^c>$, definida do seguinte modo:

- $v.n \in ACC^c_{g_1 \ldots g_n, \tau_c}$, para cada serviço v da classe c, com parâmetros de géneros $g_1, \ldots, g_n \in G_N$, e alternativa n;[2]
- $a.n \in ACC^c_{\varepsilon,\tau_c}$, para cada acção a da classe c e alternativa n de a;
- $a.n \in ACI^c$, para cada acção a da classe c e alternativa n de a;
- $s \in ATR^c_{\varepsilon,g}$, para cada atributo s, de tipo g, da classe c, tal que $g \in G_N$. □

Notação *Métodos simples*

No caso de o método de um serviço v ser simples, i.e., sem alternativas, convenciona-se que a única acção induzida é v. O mesmo se aplica às acções da especificação GNOME. □

Recordem-se as assinaturas de classe apresentadas em **3.6** e **3.7**. Como se verifica facilmente, as assinaturas apresentadas são exactamente as assinaturas de classe

---

1 O operador $\_^{\bullet}$, sobre relações binárias, denota o fecho reflexivo e transitivo.

2 Por alternativa, entenda-se o identificador da alternativa do método do serviço.

induzidas pela especificação GNOME, apresentada em **2.2** (sem considerar o caso dos relógios com defeito).

Antes de passar a descrição dos axiomas induzidos por uma especificação GNOME, apresentam-se algumas restrições à especificação, tendo em vista maior simplicidade. Embora estas restrições possam parecer muito limitativas, a passagem ao caso geral não é demasiado complicada.

Assume-se que existe apenas um atributo declarado no corpo da classe, denotado por **slot**, de género $\mathbf{g}_C$. Então, existe quanto muito uma expressão de valoração no método de cada alternativa n de cada serviço v ou de cada acção a. Essa expressão será denotada por $val_{v.n}$, no caso do serviço, e por $val_{a.n}$, no caso da acção.

Embora cada alternativa n do método de um serviço v ou de uma acção a possa ter várias cláusulas de activação, estas podem ser substituidas por apenas uma cláusula, correspondente à conjunção de todas as cláusulas, e que será denotada por $act_{v.n}$, no caso do serviço, e por $act_{a.n}$, no caso da acção. Sempre que uma alternativa do método de um serviço (resp. de uma acção) não tenha cláusulas de activação, convenciona-se que $act_{v.n}$ (resp. $act_{a.n}$) corresponde à condição universalmente verdadeira.

Numa alternativa n método de um serviço v ou de uma acção a, podem surgir várias chamadas a serviços de várias classes. A expressão que surge no argumento da chamada ao serviço v_i da classe c_i será denotada por $alvo^i_{v.n}$, no caso do serviço, e por $alvo^i_{a.n}$, no caso da acção. Tal como no caso dos atributos, assume-se que cada serviço tem, quanto muito, um parâmetro, denotado por **par**, de género $\mathbf{g_{p\,c.v}}$. Então, a expressão relativa à instanciação do parâmetro do serviço v_i (no contexto da respectiva cláusula de chamada) será denotada por $arg^i_{v.n}$ ou $arg^i_{a.n}$ consoante a cláusula surja num método de um serviço ou de uma acção.

Finalmente, impõe-se que um serviço v, de uma classe c, pode devolver, quanto muito, um resultado, denotado por **res**, de género $\mathbf{gr}_{c.v}$. Então, existe quanto muito uma expressão para a instanciação desse resultado em cada alternativa do método, que será denotada por $ret_{v.n}$.

De acordo com estas hipótese, a estrutura típica de uma classe GNOME é a que se encontra descrita na figura 3.1.

```
class c
    interface
        serv v
            par par: gp_c.v
            res res: gr_c.v
        …
    body
        slot slot: g_c
        serv v
            alt n
                enb act_v.n
                call v_1 of alvo^1_v.n: c_1
                    arg v_1.par = arg^1_v.n
                …
                call v_k of alvo^k_v.n: c_k
                    arg v_k.par = arg^k_v.n
                val slot << val_v.n
                ret v.res = ret_v.n
            …
        act a
            alt n
                enb act_a.n
                call v_1 of alvo^1_a.n: c_1
                    arg v_1.par = arg^1_a.n
                …
                call v_k of alvo^k_a.n: c_k
                    arg v_k.par = arg^k_a.n
                val slot << val_a.n
end c
```

fig. 3.1 - *Estrutura típica de uma classe*

Com base nesta caracterização de uma classe típica em GNOME, é agora possível descrever os axiomas induzidos pela especificação *spec* . Assume-se que a especificação contém apenas três classes, c_1 , c_2 e c_3.

Num 4º passo descrevem-se os axiomas relativos às meta-classes. Num 5º passo definem-se os axiomas locais a cada classe, i.e., os axiomas relativos ao nascimento do sistema, e os relativos às condições de activação e efeitos das acções. Finalmente, num 6º e último passo, introduzem-se os axiomas de interação, i.e., os axiomas que estabelecem as diversas relações de interacção entre as classes. Para este último passo, assume-se que que estão a definir os axiomas relativos a uma instância da classe $c_1 \otimes c_2$, tendo as instâncias de c_3 e as outras instâncias das classes c_1 e c_2 como ambiente.

4º passo *Axiomas relativos às meta-classes*

A especificação de cada meta-classe mc_c contém os seguintes axiomas (esquema):

(i) Axiomas de activação

$(x.\Diamond_{mc_c}(\text{inc}(t)) \Leftrightarrow (t=x.\nu))$

em que $x \in X_{mc_c}$, $t \in T_{c'}^{mc_c}$, em que $c' \leq c$.

(ii) Axiomas de valoração

$(((x.\nabla_{mc_c}=\text{inc}(t)) \wedge (A[^{x.\nu}_{x.\nu+1}])) \Rightarrow (\mathbf{X}A))$

em que $x \in X_{mc_c}$, $t \in T_{c'}^{mc_c}$, em que $c' \leq c$.

(iii) Axiomas de interacção

$((x.\nabla=v.n(t)) \Rightarrow (x.\nabla_{mc_c\,7}=\text{inc}(\pi_1(x))))$

tal que $c \in CP$, $x \in X_{c \otimes mc_c}$, $t \in T_{c'}^{mc_c}$, e ν é um serviço de nascimento. □

Destes axiomas resulta que uma instância só pode ser criada, se a sua identificação coincidir com o valor do atributo ν da meta-classe que lhe está associada. Nessas condições, a ocorrência desse serviço implica a ocorrência do serviço inc da meta-classe, que actualiza o valor da identificação da próxima instância.

De agora em diante, e embora não explicitadas, assumem-se sempre presentes as meta-classes, e assume-se que o atributo ν de cada classe mc_c é sempre traduzido para ν_c.

5º passo *Axiomas locais*

A especificação de uma classe GNOME, $<c_1, \Phi>$, induzida por *spec*, é tal que Φ contém os seguintes axiomas:

(i) Axiomas relativos ao instante inicial

O instante inicial induz o seguinte axioma:

- $x.(\divideontimes \Rightarrow (ph=\text{PN}))$

(ii) Axiomas de activação

- a acção $v_1.n \in ACE^{c_1}_{\mathbf{gp}\ c_1.v_1, \tau_{c_1}}$ induz o seguinte axioma activação:

$(x.\Diamond(v_1.n(t)) \Leftrightarrow (\underline{\text{act}}_{v_1.n} \wedge cond_{ph} \wedge x.\alpha_{v_1.n}(t) \wedge$

$\underline{\text{alvo}}^2_{v_1.n} \neq 0 \wedge \underline{\text{alvo}}^3_{v_1.n} \neq 0 \wedge \underline{\text{alvo}}^2_{v_1.n} \not\equiv \underline{\text{alvo}}^3_{v_1.n} \wedge$

$\underline{\text{alvo}}^2_{v_1.n} \not\equiv x \wedge \underline{\text{alvo}}^3_{v_1.n} \not\equiv x \wedge$

$(x.\beta_{c_2,v_2.n'_1}(\underline{\text{alvo}}^2_{v_1.n}, \underline{\text{arg}}^2_{v_1.n}) \vee \ldots \vee x.\beta_{c_2,v_2.n'_k}(\underline{\text{alvo}}^2_{v_1.n}, \underline{\text{arg}}^2_{v_1.n})) \wedge$

$$(x.\beta_{c_3,v_3.n''_1}(\underline{alvo}^{3}_{v_1.n},\underline{arg}^{3}_{v_1.n}) \vee \ldots \vee x.\beta_{c_3,v_3.n''_{k'}}(\underline{alvo}^{3}_{v_1.n},\underline{arg}^{3}_{v_1.n})))$$

em que:

- $\underline{act}_{v_1.n} = act_{v_1.n}[{}^{v_1.\mathbf{par}}_{t}][{}^{\mathbf{slot}}_{x.\mathbf{slot}}]$;
- $\underline{alvo}^{i}_{v_1.n} = alvo^{i}_{v_1.n}[{}^{v_1.\mathbf{par}}_{t}][{}^{\mathbf{slot}}_{x.\mathbf{slot}}][{}^{\mathbf{next}}_{v_{c_i^\top}}]$, para i=2,3;
- $\underline{arg}^{i}_{v_1.n} = arg^{i}_{v_1.n}[{}^{v_1.\mathbf{par}}_{t}][{}^{\mathbf{slot}}_{x.\mathbf{slot}}][{}^{\mathbf{self}}_{x}][{}^{c_2.\mathbf{next}}_{v_{c_2^\top}}][{}^{c_3.\mathbf{next}}_{v_{c_3^\top}}]$, para i=2,3;
- $cond_{ph} = \begin{cases} x.ph = \text{PN} & \text{se } v_1 \text{ for um serviço de nascimento} \\ x.ph = \text{AN} & \text{caso contrário} \end{cases}$

e que $x \in X_{c_1}$, $t \in T^{c_1}_{\mathbf{gp}\,v_1}$, $v_2.n'_1,\ldots,v_2.n'_k \in ACE^{c_2}_{\mathbf{gp}\,v_2,\tau_{c_2}}$, $v_3.n''_1,\ldots,v_3.n''_{k'} \in ACE^{c_3}_{\mathbf{gp}\,v_3,\tau_{c_3}}$ admitindo que $n'_1,\ldots n'_k$ são as únicas alternativas do serviço v_2, e que $n''_1,\ldots,n''_{k'}$ são as únicas alternativas do serviço v_3.

- a acção $a_1.n \in ACI^{c_1}$ induz o seguinte axioma (esquema) de activação:

$$\begin{aligned}(x.\Diamond(a_1.n) \Leftrightarrow \quad & (\underline{act}_{a_1.n} \wedge x.ph = \text{AN} \wedge \\ & \underline{alvo}^{2}_{a_1.n} \neq 0 \wedge \underline{alvo}^{3}_{a_1.n} \neq 0 \wedge \underline{alvo}^{2}_{a_1.n} \not\equiv \underline{alvo}^{3}_{a_1.n} \wedge \\ & \underline{alvo}^{2}_{a_1.n} \not\equiv x \wedge \underline{alvo}^{3}_{a_1.n} \not\equiv x \wedge \\ & (x.\beta_{c_2,v_2.n'_1}(\underline{alvo}^{2}_{a_1.n},\underline{arg}^{2}_{a_1.n}) \vee \ldots \vee x.\beta_{c_2,v_2.n'_k}(\underline{alvo}^{2}_{a_1.n}\,\underline{arg}^{2}_{a_1.n})) \wedge \\ & (x.\beta_{c_3,v_3.n''_1}(\underline{alvo}^{3}_{a_1.n},\underline{arg}^{3}_{a_1.n}) \vee \ldots \vee x.\beta_{c_3,v_3.n''_{k'}}(\underline{alvo}^{3}_{a_1.n},\underline{arg}^{3}_{a_1.n})))\end{aligned}$$

em que:

- $\underline{act}_{a_1.n} = act_{a_1.n}[{}^{\mathbf{slot}}_{x.\mathbf{slot}}]$;
- $\underline{alvo}^{i}_{a_1.n} = alvo^{i}_{a_1.n}[{}^{\mathbf{slot}}_{x.\mathbf{slot}}][{}^{\mathbf{next}}_{v_{c_i^\top}}]$, para i=2,3;
- $\underline{arg}^{i}_{a_1.n} = arg^{i}_{a_1.n}[{}^{\mathbf{slot}}_{x.\mathbf{slot}}][{}^{\mathbf{self}}_{x}][{}^{c_2.\mathbf{next}}_{v_{c_2^\top}}][{}^{c_3.\mathbf{next}}_{v_{c_3^\top}}]$, para i=2,3;

e que $x \in X_{c_1}$, $v_2.n'_1,\ldots,v_2.n'_k \in ACE^{c_2}_{\mathbf{gp}\,v_2,\tau_{c_2}}$, $v_3.n''_1,\ldots,v_3.n''_{k'} \in ACE^{c_3}_{\mathbf{gp}\,v_3,\tau_{c_3}}$ admitindo que $n'_1,\ldots n'_k$ são as únicas alternativas do serviço v_2, e que $n''_1,\ldots,n''_{k'}$ são as únicas alternativas do serviço v_3.

(iii) Axiomas de valoração

- a acção $v_1.n \in ACE^{c_1}_{\mathbf{gp}\,v_1,\tau_{c_1}}$ induz o seguinte axioma (esquema) de valoração:

$$(x.(\nabla = v_1.n(t)) \wedge (\forall y_2,y_3 A[{}^{x.\mathbf{slot}}_{\underline{val}_{v_1.n}}][{}^{x.ph}_{exp_{ph}}]) \Rightarrow (\mathbf{X}A))$$

em que:

- $\underline{val}_{v_1.n} = val_{v_1.n}[{}^{v_1.\mathbf{par}}_{t}][{}^{\mathbf{slot}}_{x.\mathbf{slot}}][{}^{\mathbf{self}}_{x}][{}^{v_2.\mathbf{res}}_{y_2}][{}^{v_3.\mathbf{res}}_{y_3}][{}^{c_2.\mathbf{next}}_{v_{c_2^\top}}][{}^{c_3.\mathbf{next}}_{v_{c_3^\top}}]$
- $exp_{ph} = \begin{cases} \text{AN} & \text{se } v_1 \text{ for um serviço de nascimento} \\ \text{PM} & \text{se for um serviço de morte} \\ x.ph & \text{caso contrário} \end{cases}$

e que $x \in X_{c_1}$, $t \in T^{c_1}_{\mathbf{gp}\ v_1}$ e $A \in CE^{c_1}$;

- a acção $a_1.n \in ACI^{c_1}$, induz o seguinte axioma (esquema) de valoração:

$$(x.(\nabla = a_1.n) \wedge (\forall y_2, y_3 A[^{x.\mathbf{slot}}_{\underline{val}_{a_1.n}}] [^{x.ph}_{exp_{ph}}]) \Rightarrow (\mathbf{X}A))$$

em que:

- $\underline{val}_{a_1.n} = val_{a_1.n}[^{\mathbf{slot}}_{x.\mathbf{slot}}] [^{\mathbf{self}}_{x}] [^{v_2.\mathbf{res}}_{y_2}] [^{v_3.\mathbf{res}}_{y_3}] [^{c_2.\mathbf{next}}_{\nu_{c_2^\top}}] [^{c_3.\mathbf{next}}_{\nu_{c_3^\top}}]$
- $exp_{ph} = \begin{cases} PM & \text{se } a_1 \text{ for uma acção de morte} \\ x.ph & \text{caso contrário} \end{cases}$

e que $x \in X_{c_1}$ e $A \in CE^{c_1}$; □

Notação *Expressões vs expressões GCL*

No passo anterior, as expressões sublinhadas denotam a *tradução* de uma expressão GNOME para a sua correspondente em GCL. Essa tradução é sempre explicitada. □

Para definir os axiomas de interacção induzidos pela especificação GNOME, é necessário trabalhar sobre uma assinatura com mais do que uma classe. Assim, assume-se definida a assinatura de classe de $c_1 \otimes c_2$, dada pelo coproduto em $\mathcal{A}ss_N$ das assinaturas das classes c_1 e c_2. Assume-se que os **slots** surgem indexados pelo número da classe correspondente e que não há conflitos nos nomes das acções das diferentes classes (e portanto as aplicações $\pi_{i_{ACC}}$ podem ser apenas inclusões).

6º passo *Axiomas de interacção*

A especificação $\langle c_1 \otimes c_2, \Phi \rangle$, induzida por *spec*, é tal que Φ contém os seguintes axiomas:

(i) Axiomas de chamada

- a acção $v_1.n \in ACE^{c_1 \otimes c_2}_{\mathbf{gp}\ c_1.v_1, \tau_{c_1}}$, induz os seguintes axiomas (esquema) de interacção:

$$((\pi_2(x) = \underline{alvo}^2_{v_1.n}) \Rightarrow ((x.\nabla_1 = v_1.n(t)) \Rightarrow (x.\nabla_2 = v_2.n'_1\ (\underline{arg}^2_{v_1.n}) \vee \ldots \vee x.\nabla_2 = v_2.n'_k\ (\underline{arg}^2_{v_1.n}))))$$

$$((\pi_2(x) = \underline{alvo}^2_{v_1.n}) \Rightarrow (((x.\nabla_1 = v_1.n(t)) \wedge (x.\nabla_2 = v_2.n'(\underline{arg}^2_{v_1.n})) \wedge$$
$$(\forall y_3, y'_2 A[^{x.\mathbf{slot}_1}_{\underline{val}_{v_1.n}}] [^{x.ph_1}_{exp_{ph_1}}] [^{x.\mathbf{slot}_2}_{\underline{val}_{v_2.n'}}] [^{x.ph_2}_{exp_{ph_2}}]) \Rightarrow (\mathbf{X}A))),$$

para cada uma das alternativas n' do serviço v_2;

em que:

- $\underline{alvo}^2_{v_1.n} = alvo^2_{v_1.n}[^{v_1.\mathbf{par}}_{t}] [^{\mathbf{slot}_1}_{x.\mathbf{slot}_1}] [^{\mathbf{next}}_{\nu_{c_2^\top}}]$;
- $\underline{arg}^2_{v_1.n} = arg^2_{v_1.n}[^{v_1.\mathbf{par}}_{t}] [^{\mathbf{slot}_1}_{x.\mathbf{slot}_1}] [^{\mathbf{self}}_{\pi_1(x)}] [^{c_2.\mathbf{next}}_{\nu_{c_2^\top}}] [^{c_3.\mathbf{next}}_{\nu_{c_3^\top}}]$;

- $\underline{val}_{v_1.n} = val_{v_1.n}\ [^{v_1.\mathbf{par}}_{t}]\ [^{\mathbf{slot}\ 1}_{x.\mathbf{slot}\ 1}]\ [^{\mathbf{self}}_{\pi_1(x)}]\ [^{v_2.\mathbf{res}}_{\underline{ret}_{v\ 2.n'}}]\ [^{v_3.\mathbf{res}}_{y_3}]\ [^{c_2.\mathbf{next}}_{\nu_{c_2}\top}]\ [^{c_3.\mathbf{next}}_{\nu_{c_3}\top}]$;
- $\underline{val}_{v_2.n'} = val_{v_2.n'}\ [^{v_2.\mathbf{par}}_{\underline{arg}^2_{a\ 1.n'}}]\ [^{\mathbf{slot}\ 2}_{x.\mathbf{slot}\ 2}]\ [^{\mathbf{self}}_{\pi_2(x)}]\ [^{v'_2.\mathbf{res}}_{y'_2}]\ [^{c'_2.\mathbf{next}}_{\nu_{c'_2}\top}]$;
- $\underline{ret}_{v_2.n'} = ret_{v_2.n'}\ [^{v_2.\mathbf{par}}_{\underline{arg}^2_{a\ 1.n'}}]\ [^{\mathbf{slot}\ 2}_{x.\mathbf{slot}\ 2}]\ [^{\mathbf{self}}_{\pi_2(x)}]\ [^{v'_2.\mathbf{res}}_{y'_2}]\ [^{c'_2.\mathbf{next}}_{\nu_{c'_2}\top}]$;
- $exp_{ph} = \begin{cases} AN & \text{se } v_1 \text{ for um serviço de nascimento} \\ PM & \text{se } v_1 \text{ for um serviço de morte} \\ x.ph & \text{caso contrário} \end{cases}$

$A \in CE^{c_1 \otimes c_2}$, admitindo que o serviço v_2 chama um serviço v'_2 de uma classe c'_2.

- a acção $a_1.n \in ACI^{c_1 \otimes c_2}$, induz os seguintes axiomas (esquema) de interacção:

$$((\pi_2(x) = \underline{alvo}^2_{a\ 1.n}) \Rightarrow ((x.\nabla_1 = a_{1.n}) \Rightarrow (x.\nabla_2 = v_{2.n'1}\ (\underline{arg}^2_{a\ 1.n})\ \forall \ldots\ \forall\ x.\nabla_2 = v_{2.n'k}\ (\underline{arg}^2_{a\ 1.n}))))$$

$$((\pi_2(x) = \underline{alvo}^2_{v\ 1.n}) \Rightarrow (((x.\nabla_1 = a_{1.n}(t)) \wedge (x.\nabla_2 = v_{2.n'}(\underline{arg}^2_{v\ 1.n})) \wedge$$
$$(\forall y_3, y'_2\ A[^{x.\mathbf{slot}\ 1}_{\underline{val}_{a_1.\ n}}]\ [^{x.ph\ 1}_{exp_{ph\ 1}}]\ [^{x.\mathbf{slot}\ 2}_{\underline{val}_{v\ 2.n'}}]\ [^{x.ph\ 2}_{exp_{ph\ 2}}])) \Rightarrow (\mathbf{X}A))),$$

para cada uma das alternativas n' do serviço v_2;

em que:

- $\underline{alvo}^2_{a\ 1.n} = alvo^2_{a\ 1.n}\ [^{\mathbf{slot}\ 1}_{x.\mathbf{slot}\ 1}]\ [^{\mathbf{next}}_{\nu_{c_2}\top}]$;
- $\underline{arg}^2_{a\ 1.n} = arg^2_{a\ 1.n}\ [^{\mathbf{slot}\ 1}_{x.\mathbf{slot}\ 1}]\ [^{\mathbf{self}}_{\pi_1(x)}]\ [^{c_2.\mathbf{next}}_{\nu_{c_2}\top}]\ [^{c_3.\mathbf{next}}_{\nu_{c_3}\top}]$;
- $\underline{val}_{a\ 1.n} = val_{a\ 1.n}\ [^{\mathbf{slot}\ 1}_{x.\mathbf{slot}\ 1}]\ [^{\mathbf{self}}_{\pi_1(x)}]\ [^{v_2.\mathbf{res}}_{\underline{ret}_{v\ 2.n'}}]\ [^{v_3.\mathbf{res}}_{y_3}]\ [^{c_2.\mathbf{next}}_{\nu_{c_2}\top}]\ [^{c_3.\mathbf{next}}_{\nu_{c_3}\top}]$;
- $\underline{val}_{v_2.n'} = val_{v_2.n'}\ [^{v_2.\mathbf{par}}_{\underline{arg}^2_{a\ 1.n'}}]\ [^{\mathbf{slot}\ 2}_{x.\mathbf{slot}\ 2}]\ [^{\mathbf{self}}_{\pi_2(x)}]\ [^{v'_2.\mathbf{res}}_{y'_2}]\ [^{c'_2.\mathbf{next}}_{\nu_{c'_2}\top}]$;
- $\underline{ret}_{v_2.n'} = ret_{v_2.n'}\ [^{v_2.\mathbf{par}}_{\underline{arg}^2_{a\ 1.n'}}]\ [^{\mathbf{slot}\ 2}_{x.\mathbf{slot}\ 2}]\ [^{\mathbf{self}}_{\pi_2(x)}]\ [^{v'_2.\mathbf{res}}_{y'_2}]\ [^{c'_2.\mathbf{next}}_{\nu_{c'_2}\top}]$;
- $exp_{ph} = \begin{cases} PM & \text{se a for uma acção de morte} \\ x.ph & \text{caso contrário} \end{cases}$

$A \in CE^{c_1 \otimes c_2}$, admitindo que o serviço v_2 chama um serviço v'_2 de uma classe c'_2.

(ii) Axiomas sobre o ambiente

- $((x.\underline{alvo}^2_{v\ 1.n} = \pi_2(x)) \Rightarrow (x.\beta_{c_2, v_2.n'}(\pi_2(x), b) \Leftrightarrow cond))$

em que:

- $\underline{alvo}^2_{v\ 1.n} = alvo^2_{v\ 1.n}\ [^{v_1.\mathbf{par}}_{t}]\ [^{\mathbf{slot}\ 1}_{x.\mathbf{slot}\ 1}]\ [^{\mathbf{next}}_{\nu_{c_2}\top}]$
- $((x.\Diamond(v_2.n'(b)) \Leftrightarrow (cond \wedge \alpha_{v_2.n(b)}))$

e n' é uma alternativa do serviço v_2. □

Exemplo 3.65 *Especificação relógios*

Considere-se novamente a especificação *relógios*, apresentada no exemplo **2.2**. A

especificação <*Utilizador*, Φ> induzida por *relógios* é tal que Φ contém, entre outros, os seguintes axiomas:

1. $(u.\Diamond(\text{consultar}) \Leftrightarrow$

$((u.sit=rico) \wedge (u.ph=AN) \wedge (u.rl\neq 0) \wedge (u.rl \not\geq u) \wedge u.\beta_{Relógio,obs}(u.rl)))$;

2. $(((u.\nabla=\text{consultar}) \wedge (\forall y\ A[^{u.tp}_{u.ta}][^{u.ta}_{y}])) \Rightarrow (\mathbf{X}A))$,

qualquer que seja $A \in CE^{Utilizador}$.

Se se considerar agora a especificação <*Relógio* ⊗*Utilizador*, Φ> induzida por *relógios*, Φ é tal que contém, entre outros, os seguintes axiomas:

3. $((v.rl_u=\pi_r(v)) \Rightarrow ((v.\nabla_u=\text{consultar}_u) \Rightarrow (v.\nabla_r=obs_r)))$;

4. $((v.rl_u=\pi_r(v)) \Rightarrow$

$(((v.\nabla_u=\text{consultar}_u) \wedge (v.\nabla_r=obs_r) \wedge (A[^{v.tp}_{v.ta}{}_u][^{v.ta}_{v.tempo}{}_u])) \Rightarrow (\mathbf{X}A)))$,

qualquer que seja $A \in CE^{Utilizador \otimes Relógio}$

5. $((v.rl_u=\pi_r(v)) \Rightarrow (v.\beta_{Relógio,obs}(v.rl_u) \Leftrightarrow v.ph_r=AN))$.

O axioma 1 caracteriza as condições de activação da acção *consultar*. Entre outras condições, esta acção só poderá ocorrer se o serviço *obs* instância *u.rl* da classe *Relógio* estiver activado. Quando se considera a classe *Relógio* ⊗*Utilizador*, já é possível determinar quando é que a referida acção está activada: se $((v.ph=AN) \wedge v.\alpha_{obs})$ (cf. exemplo **3.13**). É exactamente isto que o axioma 5 traduz. Note-se que o predicado $\beta_{Relógio,obs}$ representa as condições em que uma chamada pode ocorrer. Não faz sentido fazer depender essas condições do predicado α_{obs}, que representa o facto de haver uma chamada do ambiente. Daí que este não seja considerado nas condições de activação de *obs*, ao descrever $\beta_{Relógio,obs}$.

O axioma 2 caracteriza os efeitos da acção *consultar* sobre constrangimentos de estado da classe *Utilizador*. Entre outros efeitos, atribui ao atributo *ta*, um valor *qualquer* (representado pela variável y, quantificada universalmente), que corresponde ao valor retornado pela acção *obs* (cf. **2.2**), que neste contexto é desconhecido. Quando se explicita a interacção entre *Utilizadores* e *Relógios*, na classe *Relógio* ⊗*Utilizador*, para além de impôr que a ocorrência da acção *consultar* num *Utilizador* obriga à ocorrência da acção *obs* num *Relógio* (axioma 3), é necessário definir também os efeitos da ocorrência conjunta destas acções sobre constrangimentos de estado da classe *Relógio* ⊗*Utilizador* (axioma 4). Nestas condições já é possível determinar o valor retornado pela acção *obs* e atribuir esse valor ao atributo *ta*. Compare-se a expressão de valoração relativa ao atributo *ta* nos axiomas 2 e 4.

Qualquer dos axiomas de interacção está condicionado pela expressão $v.rl_u=\pi_u(v)$.

Dada uma instância da classe *Relógio⊗Utilizador*, denotada pela variável v, só poderá haver uma chamada da instância $\pi_u(v)$ da classe *Utilizador* à instância $\pi_r(v)$ da classe *Relógio*, se esta instância for aquela a que se refere o atributo *rl* do *Utilizador* (expressão que surge na cláusula de chamada, cf. exemplo **2.2**). □