3 FORMALISMO

O objetivo deste capítulo é apresentar o formalismo proposto para modelagem de sistemas envolvidos na automação elétrica de UHEs, sob uma abstração de SDED. Este formalismo tem como base a Rede de Petri Interpretada por Sinais (RPIS) juntamente com a aplicação dos conceitos de OO utilizados em SSOO, com algumas adaptações em atendimento às necessidades do tipo de sistema a ser modelado e analisado.

Primeiramente, na seção 3.1 justifica-se a escolha da RPIS como base do formalismo utilizado, em relação a outras extensões de RP Interpretadas (RPI), relacionando e analisando alguns trabalhos significativos na área.

Na seção 3.2, apresentam-se as definições formais da RP Ordinária[28], RP Condição/Evento (RP C/E), RP Lugar/Transição (RP L/T), RPIS e Alcançabilidade, com a aplicação de um exemplo prático de modelagem de um sistema elementar em RPIS, como preparação para o acompanhamento das seções seguintes.

O paradigma de OO juntamente com a RP é discutido na seção 3.3, onde também se apresentam as conclusões quanto à escolha do formalismo proposto, ou seja, a RPIS com a incorporação dos principais conceitos de OO.

A proposta da Rede de Petri Interpretada por Sinais - Orientada a Objetos (*RPIS_OO*) é introduzida na seção 3.4, onde se define sua estrutura, comportamento, semântica, representação gráfica, comunicação entre as `classes` (sub-redes) e a composição do modelo integrado do sistema modelado.

Para melhor acompanhamento das definições e conceitos apresentados, utiliza-se o formalismo proposto na modelagem de um pequeno sistema que faz parte de uma UHE. Finalmente apresentam-se as considerações finais sobre este capítulo, na seção 3.5.

[28] RP Ordinária (RP): significa que a multiplicidade ou peso dos *arcos* é sempre um, ou seja, os *arcos* não são ponderados.

3.1 Seleção do Formalismo de Apoio

De acordo com a literatura pesquisada, existem diversos formalismos baseados em RP propostos para modelagem de SDED, tendo como objetivos a modelagem e a análise do algoritmo de controle. De relevância neste trabalho, são as diferenças nos conceitos de *marcações* definidas por estes vários modelos e, de igual importância, a definição de sua comunicação com o meio ambiente, ou seja, entre o processo a ser controlado (Objeto de Controle) e o operador ou sua interface, juntamente com o Sistema de Controle. A RP como originalmente proposta, não considera ou não proporciona meios para modelagem da comunicação entre um algoritmo de controle e seu meio ambiente. Com o objetivo de atender esta necessidade, diferentes extensões do modelo da RP têm sido propostas, as mais comuns são a RP Sincronizada (RPS) e sua extensão para a RP Interpretada (RPI), como originalmente proposta por [125] *apud* [93]. Apresenta-se a seguir, uma breve explicação destes tipos de RP, de acordo com as proposições mais significativas de [22] e [93].

Na RPS, as *transições* estão associadas com eventos externos. Uma *transição dispara* se está *habilitada* e quando o correspondente evento ocorrer. Nesta abordagem, dois eventos externos nunca podem ocorrer simultaneamente. Em adição a um conjunto de eventos externos, o evento ***e*** "sempre-ocorrente" é introduzido. Uma *transição* associada com ***e*** *dispara* tão logo esteja *habilitada*. Embora a RPS tenha meios estritamente definidos para os eventos de entrada, não há definição para as saídas.

Mais tarde, a RPS foi estendida para a RPI, a qual é uma RPS que tem uma "unidade de processamento de dados" contendo variáveis e operações, modelando o fluxo de informações "de" e "para" a RP. A unidade de processamento de dados determina o valor das condições, as quais estão associadas com as *transições*. Um *lugar marcado* pode ativar operações sobre as variáveis na parte de processamento de dados. Uma *transição* em uma RPI *dispara* se a mesma está *habilitada*, se suas condições de *disparo* são verdadeiras e quando o evento ***e*** associado ocorrer. As variáveis podem ser usadas como saídas para o meio ambiente, como também internamente nas condições de *disparo*. A entrada para a RPI é ainda baseada somente em eventos, [93], [94]. Tanto a RPS como a RPI não são

restritas a *marcações* binárias; todavia a RP Subjacente[29] é em ambos os casos uma RP L/T[30].

Ainda, em [95] *apud* [22] propõem-se a RP Interpretada para Controle Lógico (RPICL), onde sua conexão com o meio ambiente é realizada com sinais binários de entradas e saídas, do mesmo modo que na RPIS [23]. Porém, de acordo com [22] existem diferenças da RPICL com a RPIS, como por exemplo, a RP Subjacente no modelo de [95] é uma RP L/T, diferente da RP Subjacente da RPIS, a qual é uma RP C/E[31].

Há também, a RP Interpretada para Controle (RPIC) introduzida por [95] e depois definida de uma forma mais ampla por [93] a qual é baseada na definição da RPI apresentada anteriormente. Na definição da RPIC, afirma-se que a mesma é uma RPI não temporizada, a qual é segura[32], determinística e tem possibilidades de entradas e saídas estendidas para ser homogênea com as entradas e saídas do Grafcet[33].

No contexto de Rede de Petri Interpretada, existe ainda a Rede de Petri Interpretada para Processo (RPIP) introduzida por [108]. Formalmente, não há diferença entre RPIP e RPIS. Entretanto, a diferença de nomes é justificada pela aplicação diferenciada e consequentemente dos diferentes critérios de corretude para os dois modelos. A RPIP é usada para modelar processos nominais. "Nominal" se refere ao comportamento esperado da planta controlada (Objeto de Controle), incluindo perturbações esperadas, sob o ponto de vista do processo. Devido a seu caráter nominal, a RPIP pode ser entendida como uma formulação positiva do problema. Ela mostra como o processo em questão deve responder

---

[29] A RP Subjacente é uma RP Ordinária Marcada Binária e sem interpretação. Representa a estrutura da RP, as atividades sequenciais, concorrentes e paralelas. Na RP Subjacente é possível analisar as propriedades estruturais e as denominadas "boas propriedades" como: Vivacidade, Limitabilidade e Reiniciabilidade.

[30] Na RP Lugar/Transição (RP L/T) é permitida mais de uma *marca* em um *lugar*.

[31] Na RP Condição/Evento (RP C/E) é permitido apenas uma *marca* em cada *lugar*, consequentemente, o pós-lugar (*post-set)* da *transição* tem que ser um *lugar* sem *marcação*.

[32] Uma RP *marcada* é Segura (ou binária), se e somente se, todos os seus *lugares* são seguros. Um *lugar* é seguro, se e somente é *k*-limitado, sendo *k=1*, ou seja, o número de *marcas* em um *lugar* não excede um.

[33] Por volta de 1972 a 1973 vários pesquisadores franceses iniciaram a descrição de controladores lógicos com Rede de Petri. O trabalho teve uma grande influência em outros desenvolvimentos na França. Em 1975, um grupo de trabalho da AFCET *(Association Française pour la Cybernétique Economique et Technique)* estabeleceu uma comissão intitulada "*Normalization of the Representation of the Specifications of Logic Controllers*". O objetivo deste grupo de trabalho foi o de encontrar um modelo de descrição comum para controladores lógicos. O resultado deste esforço foi a proposta para especificação da linguagem Grafcet em um relatório final datado de Agosto de 1997. Atualmente na França, o Grafcet é amplamente utilizado na indústria e no meio acadêmico. O Grafcet se tornou uma Norma internacional em 1988 (IEC 60848) [93].

ou atuar, em uma situação específica [23].

Então, de acordo com a pesquisa realizada, optou-se pela RPIS proposta por [22], como formalismo de base para modelagem do Sistema de Controle e do Objeto de Controle (para este, sem o mapeamento de saídas). A abordagem por RPIS resulta em uma representação transparente do comportamento causal[34] do algoritmo de controle e do fluxo de informações, "de" e "para" o processo (Objeto de Controle). Pois, na RPIS, a influência do meio ambiente no sistema é baseada em sinais, ao invés de eventos como usado na abordagem por RPI, sendo que as *transições* estão associadas com uma condição de *disparo* dada como uma função booleana dos sinais de entrada. Os *lugares* estão associados com ações especificando sinais de saída, isto é, permitem interação com o ambiente externo. Como também, a RP Subjacente da RPIS é uma RP Segura (binária).

A seguir, apresentam-se as definições formais de RP e algumas extensões, de interesse para este trabalho.

3.2 **Rede de Petri** (RP)

Serão consideradas neste subitem, de uma forma resumida, as definições formais da RP e algumas de suas extensões.

Antes da definição da RPIS, são apresentadas algumas definições e notações básicas da teoria de RP, necessárias ao entendimento do texto e acompanhamento do trabalho. Inicialmente, faz-se a apresentação da definição da RP, em sequência a RP C/E, a RP L/T e a RPIS. As definições e notações aqui apesentadas podem ser encontradas em livros textos e trabalhos sobre RP; dentre os mais significativos podem-se citar: [05], [06], [22], [29], [35], [36], [93] e [99]. As definições podem se diferenciar entre algumas literaturas citadas, no que diz respeito ao uso de terminologias e notações, porém os conceitos são os mesmos.

3.2.1 **Rede de Petri Ordinária** (RP)

A RP Ordinária é a mais elementar representação de RP. A seguir, apresenta-se sua definição.

[34] Comportamento causal: um *evento* é consequência de outro, ou seja, o *evento* início da operação acontece antes do *evento* final da operação.

Definição 3.1 - Rede de Petri Ordinária (RP).

Uma Rede de Petri Ordinária é uma tripla $RP = \langle P, T, F \rangle$, onde:

P $P = \{p_1,\ p_2,\ \dots,\ p_{|P|},\}$, é um conjunto finito e não vazio de *lugares*. Sendo $|P|$, a cardinalidade do conjunto finito de *lugares*.

T $T = \{t_1,\ t_2,\ \dots,\ t_{|T|},\}$, é um conjunto finito e não vazio de *transições*. Sendo $|T|$, a cardinalidade do conjunto finito de *transições*.
$P \cap T = \emptyset,\ T \cup P \neq \emptyset$, ou seja, P e T são conjuntos disjuntos.

F $F \subseteq (P \times T) \cup (T \times P)$, é um conjunto finito e não vazio de *arcos* que conectam *lugares* a *transições* ou *transições* a *lugares*.

A RP Ordinária apresentada na Definição 3.1 contém somente elementos estruturais. Para definir o comportamento da RP[35] e sua regra de *disparo*, necessita-se apresentar algumas terminologias para identificar conjuntos especiais de *lugares*, *transições* e o conceito de *marcação.* Apresentam-se na sequência, as definições de *Pre*, *Post* e *marcação* de uma RP.

Definição 3.2 – Pre e Post.

Pre $pre: P \times T \longrightarrow \mathbb{N}$: é a aplicação de entrada (*lugares* precedentes ou incidência anterior), sendo $\mathbb{N}$ o conjunto dos números naturais.

O pre $^{\bullet}t_j$ de uma *transição* $t_j \in T$ contém todos os *lugares* que são conectados a t_j por um *arco* direcionado, do *lugar* à *transição*: $^{\bullet}t_j = \{p_i \in P: (p_i, t_j) \in F\}$. Os elementos de $^{\bullet}t_j$ são normalmente chamados de *pré-lugares* ou *lugares* de entrada.

Post $Post: P \times T \longrightarrow \mathbb{N}$: é a aplicação de saída (*lugares* seguintes ou incidência

[35] Ao longo deste texto, todas as RP são consideradas Ordinárias e Conectadas. Conectada significa que existe pelo menos um caminho entre quaisquer dois nós (*lugar* ou *transição*), por meio de um *arco*. A RP que não é conectada é um conjunto de pelo menos duas RP conectadas.

posterior)

O post $t_j^\bullet$ de uma *transição* $t_j \in T$ contém todos os *lugares* que são conectados a t_j por um *arco* direcionado, da *transição* ao *lugar*: $t_j^\bullet = \{p_i \in P: (t_j, p_i) \in F\}$. Os elementos de $t_j^\bullet$ são normalmente chamados de *pós-lugares* ou *lugares de saída*.

Do mesmo modo são definidos, o pre e o post para um *lugar*. O pre ${}^\bullet p_i$ e o post $p_i^\bullet$ de um *lugar* $p_i \in P$ são:

$${}^\bullet p_i = \{t_j \in T: (t_j, p_i) \in F\},$$
$$p_i^\bullet = \{t_j \in T: (p_i, t_j) \in F\}.$$

Apresenta-se a seguir a definição de *marcação* em uma RP.

Definição 3.3 – Marcação de uma RP.

A *marcação* m de uma RP é um mapeamento $m: P \longrightarrow \{0, 1, 2, \ldots\}^{|P|}$, o qual atribui um número inteiro não negativo de *marcas* para cada *lugar* da RP. A *marcação* é descrita como um vetor coluna m, o qual tem uma dimensão que é igual a cardinalidade do conjunto de *lugares* P. Um *lugar* $p_i \in P$ é chamado *não marcado* sob a *marcação* m, se o correspondente elemento deste vetor, denotado por $m(p_i)$ é igual à zero. Por outro lado, é chamado *marcado* com $m(p_i)$ *marcas*. Para uma dada *marcação* m_k, o conjunto de *lugares marcados* P_m é dado por $P_m\ (m_k) = \{p \in P:\ m_k(p) > 0\}$.

Pode-se agora, apresentar a definição de uma RP Ordinária Marcada.

3.2.2 Rede de Petri Ordinária Marcada

Definição 3.4 – RP Ordinária Marcada

Uma Rede de Petri Ordinária *marcada* é uma dupla $R = \langle Q, m_0 \rangle$, onde Q é uma Rede de Petri Generalizada e m_0 uma *marcação* inicial. Q é um grafo ponderado bipartido formado pela quádrupla $Q = \langle P,\ T,\ Pre,\ Post \rangle$, de tal modo que:

$P = \{p_1, p_2, \dots p_n\}$	É um conjunto finito e não vazio de *lugares* ou *posições*, de dimensão n.
$T = \{t_1, t_2, \dots t_m\}$	É um conjunto finito e não vazio de *transições*, de dimensão m.
$P \cap T = \emptyset,\ P \cup T \neq \emptyset$	Os conjuntos P e T são disjuntos.
$Pre: P \times T \longrightarrow \mathbb{N}$	É a aplicação de incidência de entrada, define os *arcos* de entrada das *transições*.
$Post: P \times T \longrightarrow \mathbb{N}$	É a aplicação de incidência de saída, define os *arcos* de saída das *transições*.
$m_0 : P \longrightarrow \mathbb{N}$	É a *marcação* inicial da RP ($\mathbb{N}$ é o conjunto dos números naturais).

Sendo que,

${}^\bullet t_j = \{p_i \in P \mid Pre\ (p_i, t_j) > 0\}$	é o conjunto de *lugares* de entrada de t_j.
$t_j^\bullet = \{P_i \in P \mid Post\ (p_i, t_j) > 0\}$	é o conjunto de *lugares* de saída de t_j.
${}^\bullet p_i = \{t_j \in T \mid Post\ (p_i, t_j) > 0\}$	é o conjunto de *transições* de entrada de p_i.
$p_i^\bullet = \{t_j \in T \mid Pre\ (p_i, t_j) > 0\}$	é o conjunto de *transições* de saída de p_i.

Na sequência apresentam-se as definições de *habilitação* de uma *transição*

Definição 3.5 – Habilitação de uma Transição

A *transição* t_j está *habilitada* para uma *marcação* m_k se e somente se:

$m_k(p_i) \geq Pre\ (p_i,\ t_j)$, para cada $p_i \in {}^\bullet t_j$.

Se a inequação é verdadeira para cada $p_i \in {}^{\bullet}t_j$, é verdadeira para o *lugar* em P desde que $Pre\left(p_i, t_j\right) = p_i \notin {}^{\bullet}t_j$.

Como estipulado na Definição 3.1, uma RP é associada a um grafo com dois tipos de nós, *lugares* e *transições* interligados por um *arco*. A seguir, apresenta-se a definição do grafo associado a uma RP e sua notação matricial.

Definição 3.6 – Grafo e Notação Matricial.

De acordo com a Definição 3.4, um *arco* liga um *lugar* p a uma *transição* t se e somente se $Pre\left(p_i, t_j\right) > 0$ e um *arco* liga uma *transição* t a um *lugar* p se e somente se $Post\left(p_i, t_j\right) > 0$. Então, considerando os elementos $w_{ij}^{-} = Pre\left(p_i, t_j\right)$ os quais indicam o peso do *arco* ligando o *lugar* de entrada p_i à *transição* t_j, define-se a matriz de incidência anterior (ou de entrada) W^{-} de dimensão $m \times n$, onde o número de linhas é igual ao número de *lugares* e o número de colunas é igual ao número de *transições*. Similarmente, define-se matriz de incidência posterior (ou de saída) W^{+}, de dimensão $n \times m$, a partir dos elementos $w_{ij}^{+} = Post\left(p_i, t_j\right)$. Têm-se, então que:

Matriz de Incidência de Entrada:

$$W^{-} = \left[w_{ij}^{-}\right], \text{ onde } w_{ij}^{-} = \text{Pre}\left(p_i,\ t_j\right).$$

Matriz de Incidência de Saída:

$$W^{+} = \left[w_{ij}^{+}\right], \text{ onde } w_{ij}^{+} = \text{Post}\left(p_i,\ t_j\right).$$

Matriz de Incidência:

$$W = W^{+} - W^{-}$$

A seguir, apresenta-se um exemplo de aplicação das definições anteriores, utilizando-se a Figura 3.1, onde se ilustra uma RP Marcada com sua *marcação* inicial m_0, (adaptada de [93]).

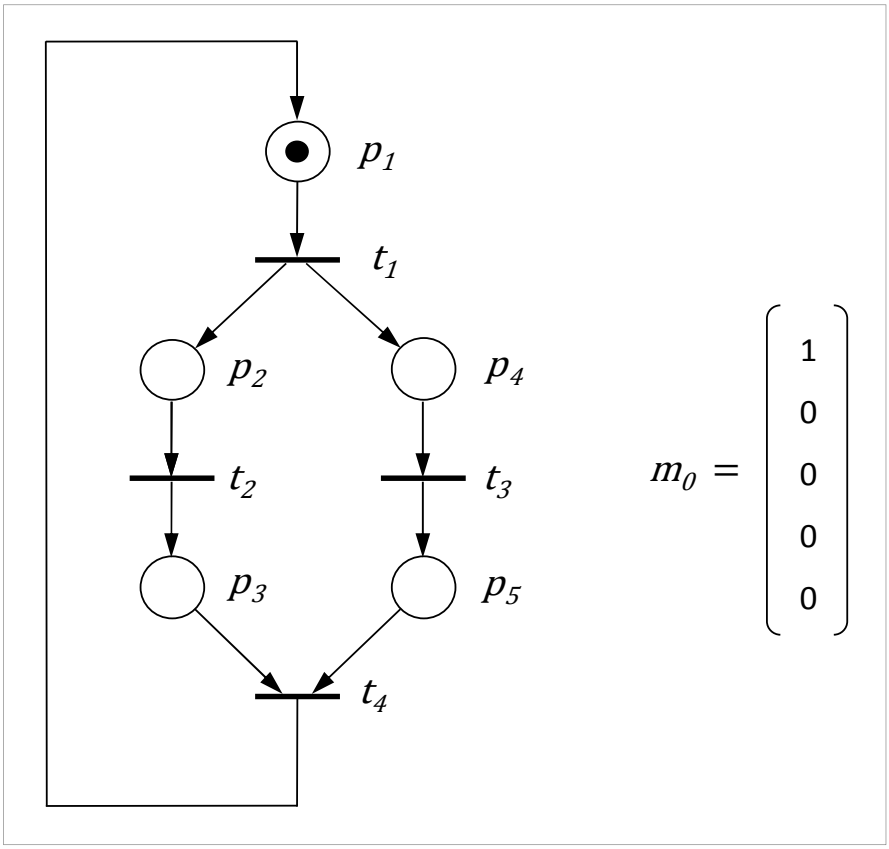


Figura 3.1 – Exemplo de Rede de Petri *Marcada* com sua *marcação* inicial m_0.

Na Figura 3.2, ilustra-se as matrizes de Entrada, de Saída e de Incidência, da RP da Figura 3.1.

$$W^- = \begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & t_4 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{matrix} \quad W^+ = \begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & t_4 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \begin{matrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{matrix} \quad W = \begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & t_4 \end{matrix} \\ \begin{bmatrix} -1 & 0 & 0 & +1 \\ +1 & -1 & 0 & 0 \\ 0 & +1 & 0 & -1 \\ +1 & 0 & -1 & 0 \\ 0 & 0 & +1 & -1 \end{bmatrix} \end{array} \begin{matrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{matrix}$$

Figura 3.2 – Matrizes de Incidência da RP da Figura 3.1.

Uma coluna da matriz corresponde a uma modificação de *marcação* feita pelo *disparo* da correspondente *transição*. Por exemplo, a primeira coluna da matriz W indica que o *disparo* da *transição* t_1 consiste em remover uma *marca* do *lugar* p_1 e adicionar uma *marca* para cada *lugar* p_2 e p_4.

Isto naturalmente não fornece uma indicação a respeito da possibilidade do *disparo* de uma *transição*, uma vez que a Matriz de Incidência é independente da *marcação*. As posições são associadas com as condições relacionadas à ocorrência de uma *transição*, então uma *transição* é *habilitada* quando todas as condições necessárias para sua ocorrência estão satisfeitas.

O conjunto de *marcas* (*fichas* ou *tokens*) é o mecanismo usado para determinar o cumprimento destas condições. O conjunto de *transições habilitadas* em um determinado *estado* da RP é o equivalente ao conjunto de eventos ativos em um determinado *estado* de um autômato finito.

A seguir define-se, a Equação Matricial Fundamental da RP.

Definição 3.7 – Equação Matricial Fundamental da RP.

Seja S a sequência de *disparo* que pode ser executada a partir de uma *marcação* m_i, que pode ser escrita como $m_i \xrightarrow{S}$.

O Vetor Característico da sequência *S*, escrito como s, é o vetor no qual a componente j corresponde ao número de *disparos* da *transição* t_j na sequência S.

Se a sequência de *disparo* é tal que $m_i \xrightarrow{S} m_k$, então se pode obter a Equação Matricial Fundamental:

$$m_k = m_i + W \,.\, s$$

Como aplicação da Definição 3.7, utiliza-se a mesma RP da Figura 3.1, onde se ilustra as sequências de *disparo* desta RP com suas *marcações*, a qual é apresentada na Figura 3.3 e 3.4, a seguir, (adaptada de [93]).

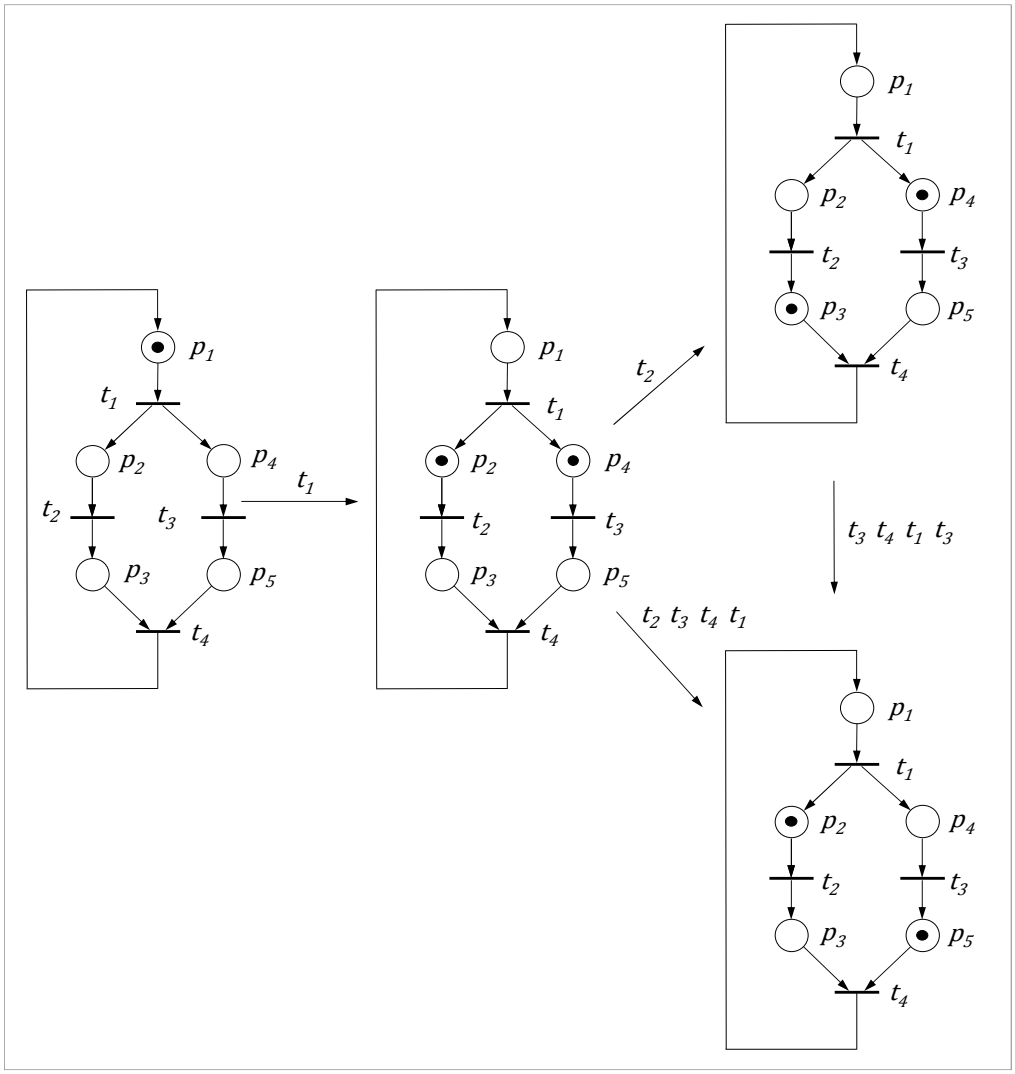


Figura 3.3 – Sequências de *disparo* da RP do exemplo 3.1.

Têm-se as seguintes *marcações*:

$$m_0 = (1,0,0,0,0)\ ;\ m_1 = (0,1,0,1,0);\ m_2 = (0,0,1,1,0);\ m_3 = (0,1,0,0,1)$$

Então, a Equação Fundamental ($m_k = m_i + W\ .\ s$) pode ser obtida, como segue:

$$\underset{(m_1)}{\begin{bmatrix}0\\1\\0\\1\\0\end{bmatrix}} + \underset{(W)}{\begin{bmatrix}-1 & 0 & 0 & +1\\ +1 & -1 & 0 & 0\\ 0 & +1 & 0 & -1\\ +1 & 0 & -1 & 0\\ 0 & 0 & +1 & -1\end{bmatrix}} \cdot \underset{(s_1)}{\begin{bmatrix}0\\1\\0\\0\end{bmatrix}} = \begin{bmatrix}0\\1\\0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\-1\\+1\\0\\0\end{bmatrix} = \underset{(m_2)}{\begin{bmatrix}0\\0\\1\\1\\0\end{bmatrix}}$$

Figura 3.4 – Equação Fundamental da RP da Figura 3.1.

A *marcação* de uma RP em um dado momento é um vetor coluna no qual seu *i*-ésimo componente é a *marcação* da posição P_i neste instante (ver Figura 3.2).

Em relação ao vetor de *marcação* m, neste texto pode ser utilizada qualquer uma das três formas possíveis: em forma de uma matriz coluna escrita na vertical entre colchetes, na sua forma transposta escrita na horizontal entre colchetes, ou ainda, na horizontal entre parênteses. Por exemplo:

$$m_0 = (1,0,0,0,0) = [1,0,0,0,0]^T = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Pode-se interpretar o *estado* de uma RP como sendo seu vetor coluna de *marcações*, isto é, o mecanismo de *transição* de *estados* é capturado pela estrutura do grafo da RP. Nota-se que o número de *marcas* atribuídas a uma posição é um inteiro não negativo arbitrário, não necessariamente limitado. Disto resulta que o número de *estados* pode chegar a infinito. Assim, usando-se uma notação de espaço de *estados*, o espaço de *estados* X de uma RP com n posições é definido por todos os vetores *n*-dimensionais cujas entradas são inteiros não negativos, isto é, $X = \mathbb{N}^n$ [06].

Embora o termo *marcação* seja mais comum do que *estado* na literatura de RP, o termo *estado* é consistente com o papel de estado na dinâmica de um sistema.

Após apresentar-se a definição de um conjunto de regras e notações para determinar a estrutura e o comportamento de uma RP, apresentam-se a seguir as definições de RP C/E, RP L/T e o Grafo de alcançabilidade, ressaltando-se a principal diferença entre estas duas RP, com aplicação de um exemplo.

3.2.3 **Rede de Petri Condição/Evento** (RP C/E)

Definição 3.8 – Rede de Petri Condição/Evento (RP C/E).

ESTRUTURA:

Uma RP C/E é uma quádrupla $RP = \langle P, T, F, m_0 \rangle$, onde $\langle P, T, F \rangle$ é uma RP Ordinária e m_0: $P \longrightarrow \{0,1\}^{|P|}$ é a *marcação* inicial da RP.

COMPORTAMENTO:

Uma *transição* t em uma RP C/E está *habilitada* sob uma *marcação* m, ou seja, $Hab(t,m) = V$ (V = Verdadeiro), se e somente se todos os *pré-lugares*, do seu conjunto *Pre*, estão *marcados* (com uma única *marca*) e todos os *lugares* do seu conjunto *Post*, não estão *marcados*, ou seja:

$Hab(t,m) = V \Leftrightarrow \left(\forall\, p \in {}^{\bullet}t_j \mid m(p) = 1\right) \wedge \left(\forall\, p \in t_j^{\bullet} \mid m(p) = 0\right)$. Desse modo, uma *transição* pode então *disparar*.

O *disparo* remove as *marcas* de todos os *pré-lugares* das *transições* e coloca uma *marca* em todos os *pós-lugares* das *transições*.

Então, na RP C/E é permitida apenas uma *marca* em cada *lugar*, consequentemente, o *pós-lugar* da *transição* deve ser um *lugar* sem *marcação*. Se mais de uma *marca* é permitida em um *lugar*, tem-se em uma RP L/T, como será visto a seguir.

3.2.4 **Rede de Petri Lugar/Transição** (RP L/T)

Definição 3.9 – Rede de Petri Lugar/Transição (RP L/T).

ESTRUTURA:

Uma RP L/T é uma quadrupla $RP = \langle P, T, F, m_0 \rangle$, onde $\langle P, T, F \rangle$ é uma RP Ordinária e m_0: $P \longrightarrow \{0, 1, 2, \ldots\}^{|P|}$ é a *marcação* inicial da RP.

COMPORTAMENTO:

Uma *transição* t em uma RP L/T está *habilitada* se todos os *pré-lugares* estão *marcados* com pelo menos uma *marca*, ou seja:

$Hab(t,m) = V \Leftrightarrow \left(\forall\, p \in {}^{\bullet}t_j \mid m(p) > 0\right)$. Desse modo, uma *transição* pode então *disparar*. O *disparo* remove uma *marca* de todos os *pré-lugares* das *transições* e coloca uma *marca* em todos os *pós-lugares* das *transições*.

Nota-se que as regras de *disparo* para a RP L/T não são as mesmas de uma RP C/E.

Isto é, em uma RP L/T os *pós-lugares* de uma *transição* não são verificados nas condições para *habilitação* da *transição*. Este ponto se torna mais claro quando da construção do Grafo de alcançabilidade para cada tipo de abordagem, como se apresenta a seguir.

3.2.5 Conjunto de Alcançabilidade (CA) e Grafo de Alcançabilidade (GA) de uma RP

Definição 3.10 – Conjunto de Alcançabilidade CA.

O CA de uma RP, $RP = \langle P, T, F, m_0 \rangle$, é o conjunto das *marcações* consistindo de m_0 e todas as *marcações* alcançáveis a partir de m_0, sendo m_0 a *marcação* inicial.

Definição 3.11 – Grafo de Alcançabilidade GA.

O GA é um grafo $GA = (CA, B \subseteq CA \times CA)$ com as *marcações* alcançáveis como vértices. Uma borda $b \in B$ com (m_i, m_k) indica que a *marcação* m_k, é diretamente alcançável a partir de m_i (onde, B: conjunto de bordas no Grafo de alcançabilidade). Os vértices do grafo são etiquetados com a correspondente *marcação* m e as bordas são etiquetadas com as correspondentes *transições*.

A única diferença na definição entre uma RP C/E e uma RP L/T com *marcação* inicial binária é sua regra de *habilitação*. No entanto, a regra de *habilitação* tem severas consequências no comportamento da RP, como ilustrado no exemplo da Figura 3.5. De acordo com [22], a regra de *habilitação* usada em uma RP C/E é chamada de Regra de *Habilitação* Forte, enquanto que a usada em RP L/T é chamada de Regra de *Habilitação* Fraca. Isto em função de que todas as *transições habilitadas* sob a Regra de *Habilitação* Forte são também *habilitadas* sob a Regra de *Habilitação* Fraca, porém o contrário não se aplica.

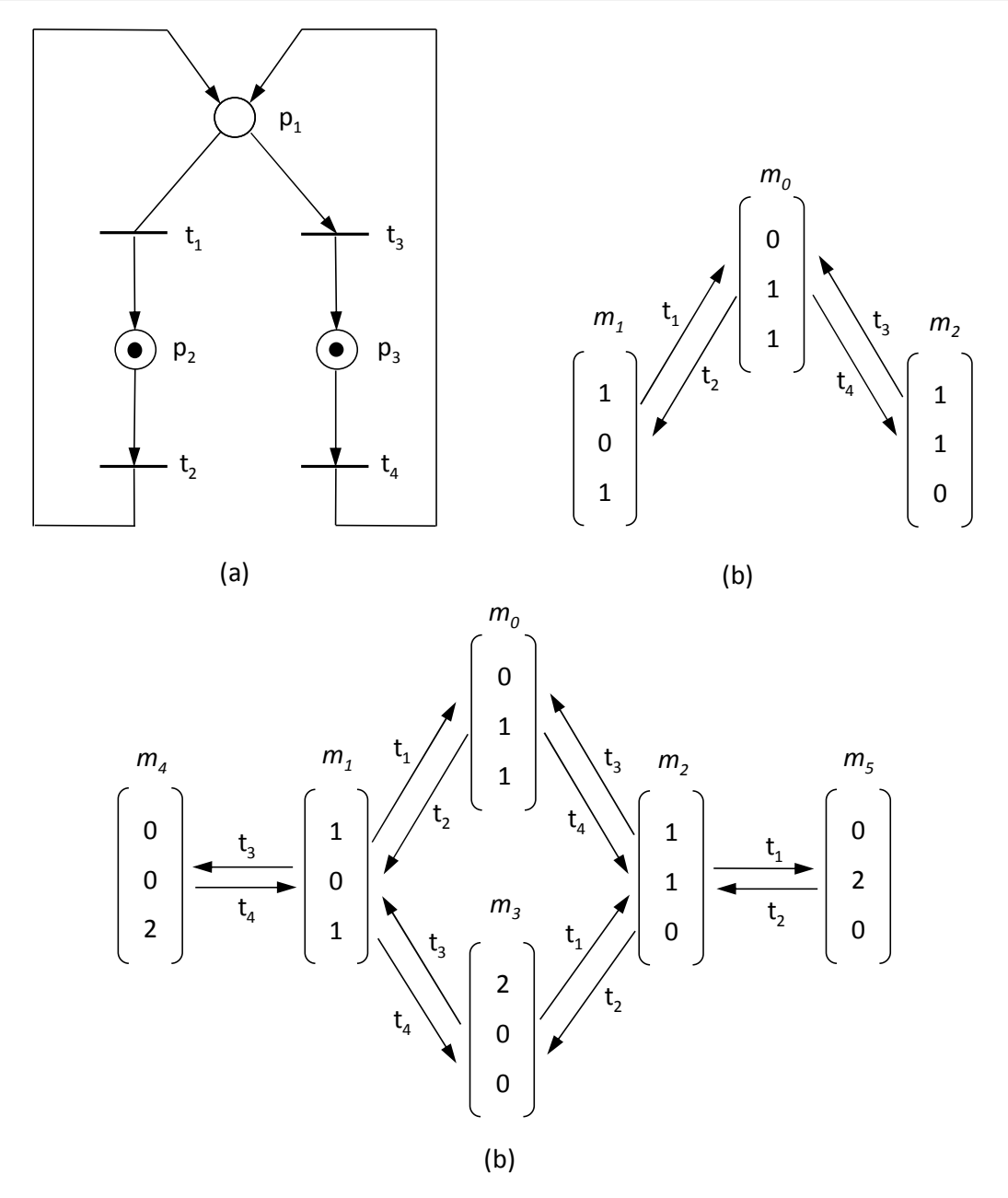


Figura 3.5 – (a) RP com *marcação* inicial. (b) Seu Grafo de alcançabilidade como RP C/E. (c) Seu Grafo de alcançabilidade como RP L/T, (adaptada de [22]).

Se o *disparo* de uma *transição habilitada* t sob uma *marcação* m em uma RP gera uma nova *marcação* m_1, então, diz-se que m_1 é alcançável diretamente a partir de m. Se sob m_1 uma *transição* t_1 *dispara*, resultando em m_2, então m_2 é dita ser alcançável a partir de m, pela sequência de *disparo* $\xrightarrow{t,\ t_1}$. Em geral, uma *marcação* m_2 é dita ser alcançável a partir de uma *marcação* m_1 se existe uma sequência de *disparo* a partir de m_1 que tem m_2 como *marcação* final.

3.2.6 Rede de Petri Interpretada por Sinais (RPIS)

Nesta seção apresenta-se a definição da RPIS, no que diz respeito a sua estrutura e seu comportamento. Também se faz referência a sua RP Subjacente, ou seja, a sua estrutura sem interpretação, que representa as atividades concorrentes, paralelas e sequenciais, a qual pode ser analisada em função da presença das chamadas "boas propriedades", como por exemplo, ausência de bloqueio mortal. As definições e terminologias aqui utilizadas são as mesmas de [22].

Na RPIS, os *lugares* são usados para descrever o *estado* do controlador. Estes *lugares* podem ser *marcados* (indicado por uma *marca* no *lugar*) ou não *marcados*. Enquanto um *lugar* é *marcado*, este pode influenciar seu meio ambiente pela definição de uma função de saída. O *estado* do Controlador é dado a qualquer momento pelo conjunto de *lugares marcados*. Da mesma forma que em uma RP, a dinâmica do modelo em uma RPIS é descrita pelo fluxo de *marcas*, por meio de *arcos* direcionados que conectam *transições* e *lugares*. O *disparo* de uma *transição* remove *marcas* do seu *lugar* de entrada e coloca *marcas* em seu *lugar* de saída. O *disparo* das *transições* depende dos sinais de entrada.

- ESTRUTURA

Definição 3.12 – Rede de Petri Interpretada por Sinais (RPIS).

Uma RPIS é descrita por uma 9-tupla[36] $RPIS = \langle P, T, F, m_{0,} I, O, \varphi, \omega, \Omega \rangle$, sendo que $\langle P, T, F, m_{0,} \rangle$ é uma RP Ordinária Marcada Binária, também chamada RP Subjacente da RPIS, neste texto denotada por RP(RPIS), onde:

I	$I = \{i_1, i_2, \ldots i_{\|I\|}\}$	É um conjunto finito e não vazio dos sinais lógicos de entrada, de dimensão $\|I\|$.

[36] Assim como, dupla ou tripla, uma n-tupla na matemática discreta é uma sequência finita (também conhecida como uma "lista ordenada") de objetos matemáticos. Uma tupla contendo *"n"* objetos matemáticos é conhecida como uma "n-tupla" ou ênupla. Tuplas são utilizadas para descrever entes matemáticos. Por exemplo, um grafo direcionado pode ser definido como uma tupla $\langle \boldsymbol{N}, \boldsymbol{A} \rangle$ onde: $\boldsymbol{N}$ é o conjunto de nós do grafo e $\boldsymbol{A}$ é o conjunto de *arcos* do grafo. $\boldsymbol{A}$ é um subconjunto de $\boldsymbol{N}$ x $\boldsymbol{N}$. Num banco de dados relacional, uma tupla equivale a uma linha de uma tabela. Um tupla ou linha armazena valores (dados) sobre entidades, relacionamentos, objetos ou ligações.

O $O = \{o_1, o_2, \dots o_{|O|}\}$ É um conjunto finito e não vazio dos sinais lógicos de saída, de dimensão $|O|$.

φ É um mapeamento associando cada *transição* $t_i \in T$ com uma condição de *disparo* $\varphi(t_i) = f(i_1, i_2, \dots i_{|I|})$. A condição de *disparo* é uma função booleana em $I, f: \{0,1\}^{|I|} \longrightarrow \{0,1\}$.

ω É um mapeamento associando cada *lugar* $p_i \in P$ com uma saída $\omega(p_i) \in \{0, 1, -\}^{|O|}$.

Ω É uma função de saída combinando a saída ω com todos os *lugares marcados* $\Omega : \{0,1\}^{|P|} \longrightarrow \{0, 1, -, c\}^{|O|}$.

Onde:

0 O sinal de saída é definido como zero.

1 O sinal de saída é definido como um.

- O sinal de saída não é especificado.

c O sinal de saída é especificado contraditório (ao menos um *lugar* o define em um, enquanto pelo menos outro *lugar* o define em zero).

- COMPORTAMENTO

i. Uma *transição* t está *habilitada* sob uma *marcação* m, se e somente se seus *pré-lugares* estão *marcados* e seus *pós-lugares* não estão *marcados*: $Hab(t, m) = V \Leftrightarrow (\forall\, p \in {}^{o}t : m(p) = 1) \wedge (\forall\, p \in t^{o} : m(p) = 0)$.

ii. Uma *transição* t *dispara* imediatamente, se e somente se está *habilitada* $(Hab(t, m) = V)$ e sua condição de *disparo* é satisfeita $(\varphi(t) = V)$. O *disparo* remove as *marcas* de todos os *pré-lugares* da *transição* e coloca uma *marca* em todos os *pós-lugares* da *transição* (V = Verdadeiro).

iii. Todas as *transições* que podem *disparar* e não estão em conflito com outras

transições disparam simultaneamente[37] (duas *transições* estão em conflito se o *disparo* de uma delas *desabilita* outra).

iv. O processo de *disparo* é iterado[38] até que uma *marcação* estável seja alcançada (ou seja, até que nenhuma *transição* não possa mais *disparar*). Uma vez que o *disparo* de uma *transição* é suposto não levar tempo algum (o tempo de *disparo* de uma *transição* é zero ou o *disparo* é instantâneo), o *disparo* iterado é interpretado como simultâneo. Isto também significa que uma mudança no valor do sinal de entrada não pode ocorrer (e por isso, não pode ser considerado) durante o processo de *disparo*.

v. Depois de alcançado um novo *estado* de *marcações*, os sinais de saída são recalculados pela aplicação da função Ω (função de saída) para a *marcação*.

- REPRESENTAÇÃO GRÁFICA

Tal como em uma RP, os *lugares* de uma RPIS são representados por círculos, *transições* por barras, a relação de fluxo por *arcos* e as *marcas* por pontos no círculo do correspondente *lugar*. Em adição, as *transições* de uma RPIS são rotuladas com suas condições de *disparo* e os *lugares* são rotulados com suas saídas. Sendo que, os *lugares* e as saídas não são temporizados; as *transições* e as entradas também não são temporizadas. Adicionalmente, os *lugares* e as *transições* podem conter comentários.

Em função de um possível grande número de sinais de saída, podem-se representar as saídas não somente em forma vetorial, mas também por uma especificação explícita da influência das variáveis. Se, por exemplo, $O = \{o_1, o_2, \ldots o_{100}\}$, então $\omega(p_i)$: $o_1 = 1$; $o_3 = 0$ é melhor do que $\omega(p_i) = (1, -, 0, -, \ldots, -)$.

Como aplicação dos conceitos apresentados anteriormente, na Figura 3.6 apresenta-se o P&ID de um tanque de aquecimento e na Figura 3.7, uma possível solução para seu algoritmo de controle em RPIS (adaptada de [22]).

[37] Embora seja possível fracionar o tempo de tal forma que seus disparos deixem de ser simultâneos, de acordo com a tese de [22], o autor considera que os disparos são simultâneos.

[38] Em matemática, função iterada é uma função que é composta consigo mesma, em forma repetida, em um processo chamado iteração; o termo também define uma técnica utilizada em análise numérica para se chegar a resultados de problemas complexos de serem resolvidas pelo método algébrico.

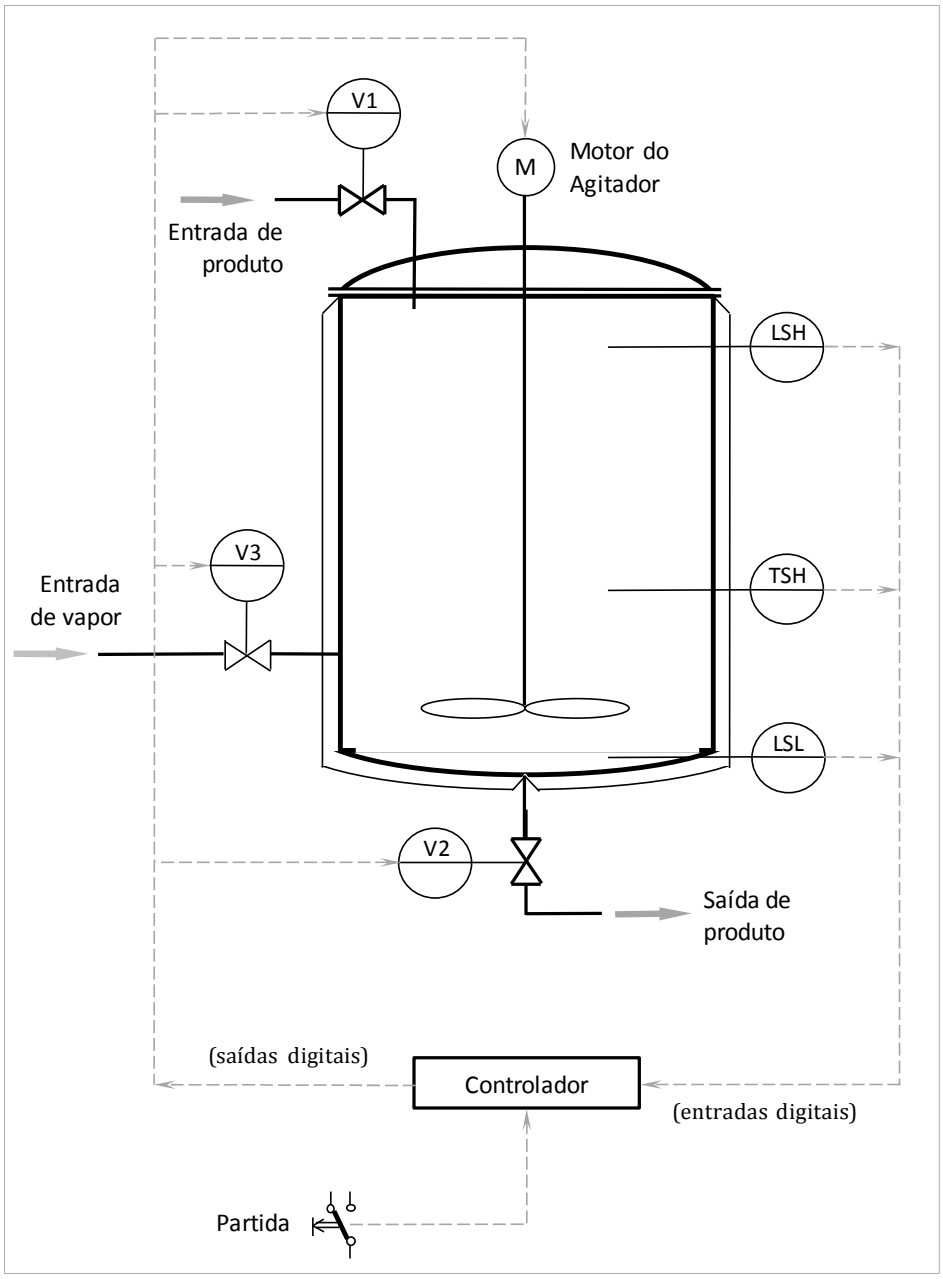


Figura 3.6 - P&ID de um Tanque de Aquecimento (adaptado de [22]).

Tabela 3.1 – Lista de Sinais para o Tanque de Aquecimento representado na Figura 3.6.

Entradas	**Descrição** (Nível Lógico **1**)
i_1	LSL – Tanque Não Vazio
i_2	LSH – Tanque Cheio
i_3	TSH – Temperatura Alta
i_4	Botão de Partida

Saídas	**Descrição** (Nível Lógico **1**)
o_1	Abrir Válvula V1
o_2	Abrir Válvula V2
o_3	Ligar Motor Agitador M
o_4	Abrir Válvula Aquecimento V3

Tem-se a seguinte especificação informal a ser satisfeita pelo Controlador do Tanque de Aquecimento:

Após pressionar o botão de partida ($i_4 = 1$), inicia-se o enchimento do Tanque vazio ($i_1 = 0$) pela operação da Válvula V1. O Tanque cheio ($i_2 = 1$) é aquecido pela operação da Válvula V3 ($o_4 = 1$) até o sensor de temperatura sinalizar que a temperatura limite é alcançada ($i_3 = 1$). Após isto, o tanque aquecido é esvaziado abaixo do nível mínimo ($i_1 = 0$) pela abertura da Válvula V2 ($o_2 = 1$). Durante todo o processo os conteúdos são agitados ($o_3 = 1$).

De acordo com a lista de sinais e a especificação informal do processo, apresenta-se uma possível solução de modelagem do Sistema de Controle em RPIS, na Figura 3.7, a seguir.

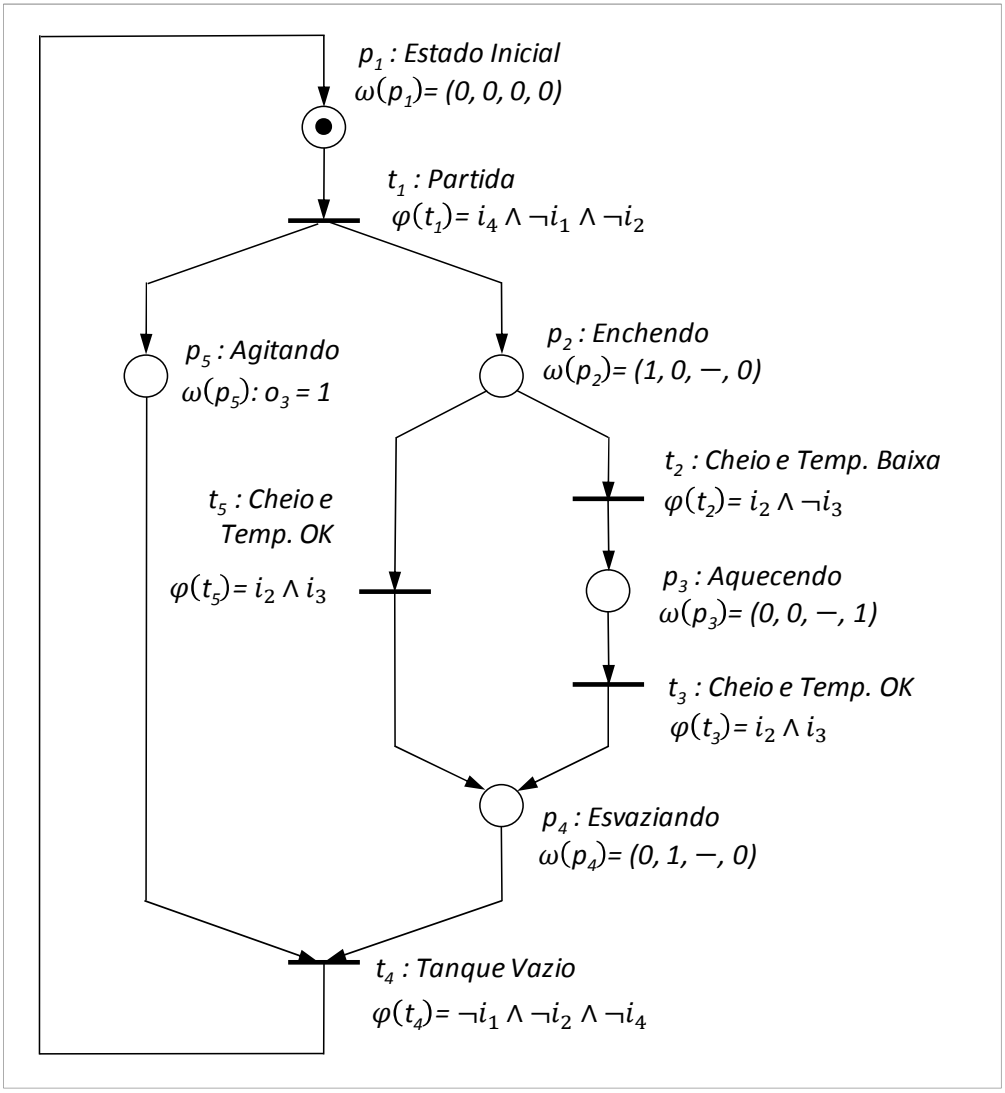


Figura 3.7 – RPIS do Sistema de Controle do Sistema representado na Figura 3.6.

O algoritmo na Figura 3.7 funciona da seguinte maneira:

i. No *estado* inicial somente p_1 é *marcado*, indicado por uma *marca* em p_1 e, consequentemente, as saídas da RPIS $(o_1, o_2, o_3, o_4) = (0, 0, 0, 0)$ e $m_0 = (m_{p_1}, m_{p_2}, m_{p_3}, m_{p_4}, m_{p_5}) = (1, 0, 0, 0, 0)$.

ii. A *transição* t_1 *dispara* quando o botão de partida é pressionado $(i_4 = 1)$ e o tanque está vazio $(i_1 = 0)$ e $(i_2 = 0)$.

iii. A *marca* é removida de p_1 e uma *marca* é colocada em cada um dos *lugares* em p_2 e p_5. A nova saída da RPIS resulta em $(o_1, o_2, o_3, o_4) = (1, 0, 1, 0)$, onde $(o_3 = 1)$ significa "agitando" e $(o_1 = 1)$ "enchendo" e $m_1 = (m_{p_1}, m_{p_2}, m_{p_3}, m_{p_4}, m_{p_5}) = (0, 1, 0, 0, 1)$.

iv. Depois que o nível cheio é alcançado $(i_2 = 1)$, o processo posterior depende da temperatura do tanque.

v. Se a temperatura já está acima do nível desejado $(i_3 = 1)$, então t_5 *dispara*, removendo uma *marca* de p_2 e colocando uma *marca* em p_4, ficando $m_2 = (m_{p_1}, m_{p_2}, m_{p_3}, m_{p_4}, m_{p_5}) = (0, 0, 0, 1, 1)$.

vi. Se, de outra maneira, a temperatura está abaixo do nível $(i_3 = 0)$, então t_2 *dispara*, também removendo uma *marca* de p_2, mas colocando-a em p_3, ficando $m_3 = (m_{p_1}, m_{p_2}, m_{p_3}, m_{p_4}, m_{p_5}) = (0, 0, 1, 0, 1)$.

vii. O tanque é aquecido $(o_4 = 1)$ até que a temperatura esteja OK, $(i_3 = 1)$ e t_3 possa *disparar*, movendo assim a *marca* de p_3 para p_4, ficando $m_4 = (m_{p_1}, m_{p_2}, m_{p_3}, m_{p_4}, m_{p_5}) = (0, 0, 0, 1, 1)$.

viii. Em p_4 o tanque é esvaziado $(o_2 = 1)$.

ix. Quando o tanque está vazio e o botão de partida é liberado, o *disparo* de t_4 remove a *marca* de p_4 e p_5, e coloca uma *marca* em p_1, resultando no *estado* inicial novamente, $m_0 = (m_{p_1}, m_{p_2}, m_{p_3}, m_{p_4}, m_{p_5}) = (1, 0, 0, 0, 0)$.

3.3 Paradigma OO e Rede de Petri

Os trabalhos que buscam a integração entre RP e Orientação a `Objetos`, geralmente podem ser considerados sobre duas abordagens, de acordo com [103], [104], [105], [106]: `objetos` dentro de RP ou RP dentro de `objetos`. Na sequência, faz-se uma breve explicação de cada uma.

`Objetos` dentro de RP: os dados que transitam pela RP são `objetos`, ou seja, as *marcas* da RP são `objetos`. O *disparo* de *transições* invoca a chamada de `métodos` destes `objetos`, atualizando o valor de seus `atributos`, como também, podem instanciar[39] ou destruir um `objeto`.

RP dentro de `objetos`: o comportamento interno dos `objetos` é dado por uma RP e sua *marcação* representa o *estado* do `objeto`. O *disparo* de *transições* representa a execução de um `método` pelo determinado `objeto`.

Um exemplo é a Rede de Petri Orientada a Objetos (RPOO), onde as entidades e relacionamentos de um modelo são descritas por meio de diagramas de `classes`, assim como em outras metodologias OO. Nesta extensão de RP, a formalização prescreve a construção de diagramas que detalham o comportamento de cada `classe`. O detalhamento de uma `classe` em RPOO pode ser definido, por exemplo, por Rede de Petri Colorida (RPC) [107] e suas inscrições de interação. A uma *transição* da RPOO, é permitido associar inscrições que descrevem a relação daquela ação ou condições com outros `objetos` que compõem o sistema.

Outra abordagem conhecida como RPO – Rede de Petri a `Objetos` (OPN – *Object Petri Net*) [60], também definida por alguns autores como Redes-dentro-de-Redes *(Nets-within-Nets)* [106], compreende a utilização das duas primeiras (`objetos` dentro de Rede de Petri e Rede de Petri dentro de `objetos`) pela criação de uma estrutura hierárquica dentro da RP, onde se faz uso mais intenso dos conceitos de OO. O paradigma Redes-dentro-de-Redes fornece uma técnica de modelagem arrojada, possibilitando *marcas* a própria estrutura de uma RP. Esta RP, também chamada de Rede Marca ou Rede Objeto, suporta a técnica de modelagem orientada a `objetos`, pois pode representar `objetos` do mundo real com uma dinâmica adequada. Entre a Rede Objeto e o restante da RP,

[39] Instanciar um `objeto` significa criar um `objeto` ou, neste contexto, definir uma configuração inicial para o mesmo.

qualificada por alguns autores de Rede Sistema, existem vários mecanismos de interação, bem como entre as diferentes Redes Objeto. Na Rede Sistema, as *marcas* no *lugar* podem ser interpretadas como `objetos`, que ainda podem possuir uma Rede Objeto. Uma *marca* na Rede Sistema pode possuir como `atributo`, por exemplo, a *marcação* de um determinado *lugar* de sua Rede Objeto. Uma Rede Objeto, por sua vez, pode ser a RP do sistema para suas *marcas* e assim sucessivamente, gerando uma estrutura hierárquica. Em termos de modelagem de SSOO, a modelagem orientada a `objetos` significa que o *software* é projetado como uma interação de `objetos` discretos, incorporando estrutura de dados e comportamento. Ainda, sob um ponto de vista de RP, é natural representar tais `objetos` por *marcas*, que por sua vez, podem representar também uma RP. Em RP L/T, na maioria dos casos estes `objetos` representam recursos ou indicam o *estado* de controle. `Objetos` mais complexos podem ainda ser modelados por *marcas* (cores) inseridas em uma RP Colorida (RPC) [106].

Com base na análise dos trabalhos citados anteriormente e com o foco na proposta desta tese, apresentam-se abaixo as seguintes conclusões, quanto à questão do formalismo proposto, ou seja, a proposta de uma nova RP: a Rede de Petri Interpretada por Sinais - Orientada a Objetos (*RPIS_OO*):

- A RPIS oferece certa estruturação da parte de controle (esta é descrita pela RP não Interpretada, a qual descreve os encadeamentos de eventos e de atividades) e especificam ligações entre esta parte de controle, a parte dados (ou parte operativa, que descreve os dados internos ao sistema) e o ambiente externo (participa na troca de informações externas ao sistema modelado, sob a forma de condições e tratamentos), porém não permitem um método para estruturar os dados. A Rede a Objetos propõe uma estruturação de dados na própria rede, permitindo uma definição estruturada pela individualização da *marca*, como um `objeto` que detém seus `atributos` e `métodos`.

- Com o uso dos conceitos de OO, espera-se obter condições para uma melhor estruturação, possibilitando modularidade e reutilização dos modelos.

- Possibilidade de construção de um modelo integrado do sistema, o qual é composto pelos `objetos` do Objeto de Controle e pelos `objetos` do Sistema de Controle.

- Por meio do modelo integrado do Sistema de Controle com o Objeto de Controle torna-se possível a sua Verificação formal de uma forma estruturada.

- É possível aproveitar e serem mantidas a estrutura e características dos modelos do sistema para a próxima etapa proposta do desenvolvimento, que é a Realização (Fase 4 da Figura 1.1), quando tratando-se de *softwares* de controle orientados a objetos a serem implementados no CLP como linguagem e estrutura do algoritmo de controle, como por exemplo, o CFC[40] ou o DIPMCS da IEC 61499.

3.4 Rede de Petri Interpretada por Sinais juntamente com o Paradigma de Orientação a Objetos

Considera-se nesta proposta, que um sistema pode ser modelado por um conjunto de `objetos`, que se comunicam de acordo com sua estrutura e comportamento pré-definidos, organizados de acordo com suas `classes` que fazem parte, ou do Objeto de Controle ou do Sistema de Controle. As `classes` e os `objetos` são modelados em um novo tipo de RP, o qual contempla as principais características da RPIS juntamente com conceitos de orientação a `objetos`.

Assim, nesta seção apresenta-se a aplicação dos conceitos de orientação a `objetos` à RPIS, define-se a estrutura e o comportamento das sub-redes (`classes`) e `objetos`, suas interfaces e modos de comunicação.

3.4.1 Estrutura e Dinâmica dos Modelos das Sub-redes (`Classes`) e `Objetos`

De acordo com [07] e [62], resumidamente, cada `classe` é a representação de um conjunto de `objetos` com características, comportamentos, estrutura, relacionamentos e semânticas comuns, ou seja, os `objetos` que pertencem à mesma `classe` possuem os

[40] CFC *(Continuous Function Chart)* é uma extensão das linguagens da norma IEC 61131-3, é uma linguagem semelhante à FBD *(Function Block Diagram)*. Em CFC os blocos lógicos podem ser colocados livremente sobre a tela, ligados um ao outro, podem ser parametrizados e podem-se ainda criar ciclos de realimentação sem variáveis temporárias [116].

mesmos `atributos`, `operações` e `métodos`[41]. Disto, e tendo como referência o trabalho apresentado em [12] com algumas modificações, propõem-se neste desenvolvimento que cada `classe` definida seja associada a uma sub-rede, a qual modela a estrutura e o comportamento dos `objetos` pertencentes à `classe` representada por esta sub-rede. Igualmente, para uma determinada situação, a *marcação* da sub-rede desta `classe` representa o *estado* dos `objetos` pertencentes à mesma, de acordo com a inscrição dos *lugares* em que a *marcação* se encontra.

Então, considerando-se os conceitos e premissas anteriormente descritas, pode-se admitir que um agrupamento de sub-redes de Petri Interpretada por Sinais – Orientada a Objetos e *marcadas,* as quais interagem entre si por meio de suas interfaces, são capazes de modelar um determinado sistema em estudo, no nível de abstração de interesse. Define-se então a seguir, uma *RPIS_OO* como sendo um agrupamento de uma ou mais sub *RPIS_OO* *marcadas* e inter-relacionadas, sendo que a *RPIS_OO* é uma abstração do sistema a ser modelado e as sub *RPIS_OO* representam as `classes` que modelam o comportamento e as características dos `objetos` do sistema.

Ainda, de acordo com o conceito apresentado no Capítulo 1, proposto na Figura 1.1, considera-se que o sistema a ser modelado é composto por dois tipos de sub *RPIS_OO* *marcadas* (`classes`), um tipo que pertence ao Objeto de Controle e outro que pertence ao Sistema de Controle, as quais são apresentadas a seguir.

Em vista disso, um sistema, sob uma abstração de SDED, pode ser modelado por um conjunto de sub *RPIS_OO* *marcadas* que trocam informações entre si por meio de suas `interfaces`.

Definição 3.13 – Rede de Petri Interpretada por Sinais - Orientada a Objetos Marcada - *RPIS_OO*

Uma *RPIS_OO* a qual representa o modelo, a abstração de um determinado sistema, é constituída por um conjunto de sub *RPIS_OO* *marcadas*, diferenciadas em dois tipos, um tipo representa as sub-redes (`classes`) pertencentes ao Objeto de Controle e outro, as

---

[41] `Operação` e `método`: A UML diferencia `operação` e `método`, da seguinte forma: uma `operação` é a especificação de um `serviço` que pode ser requerido por qualquer `objeto` da `classe`; um `método` é uma implementação de uma `operação`.

sub-redes (`classes`) do Sistema de Controle, conforme se define a seguir.

$RPIS_OO$: é a RP que representa o sistema modelado. É composto pelas sub-redes do Objeto de Controle e pelas sub-redes do Sistema de Controle. Onde:

$RPIS_OO = RPIS_OC \cup RPIS_SC$. Sendo que, $RPIS_OC \cap RPIS_SC = \emptyset$ e $RPIS_OC \cup RPIS_SC \neq \emptyset$. Os conjuntos $RPIS_OC$ e $RPIS_SC$ são disjuntos. Onde:

$RPIS_OC$: é o conjunto de sub-redes que representa as `classes` do Objeto de Controle. Sendo:

$RPIS_OC = \{RPIS_OC_{1O}, RPIS_OC_{2O}, \dots, RPIS_OC_{|m|O}\}$. Onde, $|m|$, é a cardinalidade do conjunto finito de `classes` do Objeto de Controle e O é a inscrição a qual determina que a sub $RPIS_OO$ pertence ao Objeto de Controle. Onde:

$RPIS_OC_{mO}$: é uma sub Rede de Petri Interpretada por Sinais - Orientada a Objetos Marcada, de número m, do Objeto de Controle.

$RPIS_SC$: é o conjunto de sub-redes que representa as `classes` do Sistema de Controle. Sendo:

$RPIS_SC = \{RPIS_SC_{1S}, RPIS_SC_{2S}, \dots, RPIS_SC_{|n|S}\}$. Onde, $|n|$ é a cardinalidade do conjunto finito de sub-redes do Sistema de Controle e S é a inscrição que determina que a sub $RPIS_OO$ pertence ao Sistema de Controle. Onde:

$RPIS_SC_{nS}$: é uma sub Rede de Petri Interpretada por Sinais - Orientada a Objetos Marcada, de número n, do Sistema de Controle.

Dessa forma, a $RPIS_OC_{mO}$ e a $RPIS_SC_{nS}$ podem modelar o comportamento e a estrutura de sua respectiva `classe`.

3.4.2 Estrutura de uma $RPIS_OC_{mO}$

Definição 3.14 – Sub Rede de Petri Interpretada por Sinais - Orientada a Objetos Marcada do Objeto de Controle - $RPIS_OC_{mO}$.

Uma $RPIS_OC_{mO}$ é constituída pela 6-tupla $RPIS_OC_{mO} = \langle OC_{mO}, RP_{mO}, V_{mO}, I, \varphi_{mO}, M_{mO} \rangle$, onde:

OC_{mO} É a `classe` de número m do Objeto de Controle constituída pela $RPIS_OC_{mO}$. Sendo que $m \in \mathbb{N}^*$ e O é a inscrição que designa a sub-rede como pertencente ao Objeto de Controle. No contexto desta tese, para reduzir a quantidade de letras, as `classes` do Objeto de Controle serão identificadas individualmente apenas como C_{mO} (`classe` de número m do Objeto de Controle).

RP_{mO} É uma RP Ordinária, da `classe` de número m do Objeto de Controle, definida pela 5-tupla $\langle P_{mO}, T_{TmO}, F_{mO}, Pre_{mO}, Post_{mO} \rangle$, onde a multiplicidade ou *peso* dos *arcos* é sempre um, ou seja, os *arcos* não são ponderados, onde:

P_{mO} $P_{mO} = \{p_{1_mO}, p_{2_mO}, \dots p_{i_mO}\}$, é um conjunto finito e não vazio de *lugares* da `classe` do Objeto de Controle.

T_{TmO} É o conjunto de todos os tipos de *transição* definidos para esta `classe`, sendo:
$T_{TmO} = T_{mO} \cup T_{P_mO}$;
$T_{mO} \cap T_{P_mO} = \emptyset$;
$T_{mO} \cup T_{P_mO} \neq \emptyset$, onde:

T_{mO} : Conjunto de *transições* da $RPIS_OC_{mO}$ de número m do Objeto de Controle (inscrição O), que não são *produtoras* nem são *consumidoras*, onde:

$T_{mO} = \{t_{1_mO}, t_{2_mO}, \dots, t_{j_mO}\}$, sendo:

t_{j_mO}: *Transição* de número j da `classe` de número m do Objeto de Controle (inscrição O), que não é *produtora* nem *consumidora*, denominada como *transição normal* da `classe` C_{mO}. Com,

$t_{j_mO} \in T_{mO}$.

T_{P_mO}: Conjunto de *transições* da $RPIS_OC_{mO}$ que `produz` um `método` a outras `classes`, o qual descreve um `serviço` implementado por esta `classe`, ou seja, é o conjunto de `operações` que a `classe` $RPIS_OC_{mO}$ disponibiliza e pode ser "chamado" por outras `classes`, onde:

$T_{P_mO} = \{t_{p1_mO}, t_{p2_mO}, \dots, t_{px_mO}\}$, sendo:

t_{px_mO}: *Transição produtora* de número x da `classe` (sub-rede) de número m do Objeto de Controle (inscrição O), que pode ser *consumida* por outras `classes`. A *transição produtora* x tem uma `operação` que pode ser "chamada" por outras `classes`. Com, $t_{px_mO} \in T_{P_mO}$.

F_{mO} $F_{mO} \subseteq (P_{mO} \times T_{TmO}) \cup (T_{TmO} \times P_{mO})$, é um conjunto finito e não vazio de *arcos* que conectam *lugares* a *transições* ou *transições* a *lugares* da $RPIS_OC_{mO}$.

Pre_{mO} Cada Pre_{mO} ${}^{\bullet}t_{j_mO}$ e ${}^{\bullet}t_{px_mO}$ das *transições* $t_{j_mO}, t_{px_mO} \in T_{TmO}$, contém todos os *lugares* que são conectados a t_{j_mO}, t_{px_mO} por *arcos* direcionados, dos *lugares* às *transições*:

${}^{\bullet}t_{j_mO} = \{p_{i_mO} \in P_{mO} : (p_{i_mO}, t_{j_mO}) \in F_{mO}\}$;

${}^{\bullet}t_{px_mO} = \{p_{i_mO} \in P_{mO} : (p_{i_mO}, t_{px_mO}) \in F_{mO}\}$.

Os elementos de p_{i_mO} são denominados *pré-lugares* ou *lugares* de entrada, da `classe` C_{mO} do Objeto de Controle.

$Post_{mO}$ Cada $Post_{mO}$ $t_{j_mO}^{\bullet}$ e $t_{px_mO}^{\bullet}$ das *transições* t_{j_mO} e $t_{px_mO} \in T_{mO}$, contém todos os *lugares* que são conectados a t_{j_mO}, t_{px_mO} por *arcos* direcionados, das *transições* aos *lugares*:

$t_{j_mO}{}^{\bullet} = \{p_{imO} \in P_{mO} : (t_{j_mO}, p_{i_mO}) \in F_{mO}\}$;

$t_{px_mO}{}^{\bullet} = \{p_{i_mO} \in P_{mO} : (t_{px_mO}, p_{i_mO}) \in F_{mO}\}$.

Os elementos de $t_{j_mO}{}^{\bullet}$ e $t_{px_mO}{}^{\bullet}$ são denominados *pós-lugares* ou *lugares* de saída, da `classe` C_{mO} do Objeto de Controle. Sendo $Pre_{mO} \cap Post_{mO} = \emptyset$.

V_{mO} Conjunto de variáveis formais que pertence ao conjunto de `atributos` de uma `classe` de número m do Objeto de Controle. São considerados dois tipos. Um para `atributos` numéricos, isto é, variável que designa um tipo de dado numérico, de acordo com a IEC 61131, sendo que neste trabalho se limita a *booleano*, para todo o sinal $i_{|I|} \in I$; e outro para o tipo **"NOME_DO_OBJETO_DO_OBJETO_DE_CONTROLE"**. Este tipo relaciona o nome dos `objetos` com os quais um determinado `objeto` do Objeto de Controle pode trocar informações por meio de suas interfaces, denominada *Variável de Interface*, definida como I_{mO-OK}, onde m é o número da `classe` do Objeto de Controle e k o número do `objeto` desta `classe`.

I $I = \{i_1, i_2, \dots i_{|I|}\}$, é um conjunto finito dos sinais lógicos de entrada, de dimensão $|I|$. Sendo $|I| > 0$.

φ_{mO} É um mapeamento associando cada *transição* $t_{j_mO} \in T_{mO}$ com uma função (ou condição) de *disparo* $\varphi_{mO}(t_{j_mO}) = f(i_1, i_2, \dots\ i_{|I|})$ e também, associando cada *transição* $t_{px_mO} \in T_{mO}$ com uma função de *disparo* $\varphi_{mO}(t_{px_mO}) = f(i_1, i_2, \dots\ i_{|I|})$. As condições de *disparo* $\varphi_{mO}(t_{j_mO})$ e $\varphi_{mO}(t_{px_mO})$ são funções booleanas em I, $f: \{0,1\}^{|I|} \longrightarrow \{0,1\}$.

M_{mO} É o conjunto de *Marcações*, significando que cada uma representa o *estado* de um `objeto` da sub-rede do Objeto de Controle, sendo:

$M_{mO} = \{M_{0_mO},\ M_{1_mO},\ M_{2_mO},\ \dots, M_{K_mO}\}$. Onde:

M_{0_mO}: é a *Marcação* inicial da `classe` m do Objeto de Controle.

M_{K_mO}: é a *Marcação* K da `classe` m do Objeto de Controle. Sendo:

$M_{K_mO} = \langle O_{K_mO}, V_{K_mO}, m_{K_mO}_k \rangle$, onde:

O_{K_mO} Nome do `objeto` K, da $RPIS_OC_{mO}$, onde seu *estado* é representado pela *marcação* $m_{K_mO}_k$.

V_{K_mO} Instância de V_{mO} da $RPIS_OC_{mO}$, para o `objeto` K.

$m_{K_mO}_k$ É a *marcação* alcançável k do `objeto` K da `classe` m na

$RPIS_OC_{mO}$, de sua `classe` C_{mO}. $m_{K_mO}_k : P \longrightarrow \{0,1\}$, (cada *lugar* pode conter apenas uma *marca* de cada `objeto`, ou seja, não é lógico um `objeto` apresentar mais de um *estado* ao mesmo tempo).

3.4.3 Comportamento e Semântica de uma $RPIS_OC_{mO}$

Em uma $RPIS_OC_{mO}$ os *lugares* e *transições* têm os seguintes significados e comportamentos:

Lugares

i. Em uma $RPIS_OC_{mO}$, um *lugar* quando *marcado* representa o *estado* do(s) `objeto(s)` naquele *lugar*, ou seja, significa uma determinada situação do `objeto`. Uma situação é um *estado* da `classe` que pode conter *marcas* (`objetos`) ou não. Um *estado* pode estar ativo ou não ativo para um determinado `objeto`, se o mesmo estiver ou não no específico *lugar*, indicado por uma *marcação*.

ii. O mesmo `objeto` o qual é uma *marca* da sub-rede ($RPIS_OC_{mO}$) também representa uma $RPIS_OO$, onde seus *lugares,* quando *marcados,* representam o *estado* parcial *(marcação)* daquele objeto.

iii. Em uma $RPIS_OC_{mO}$, os *lugares* não são temporizados. Também não está definido um mapeamento que associa o *lugar* a uma ação de saída digital, isto é, os *lugares* quando *marcados* representam somente o último *estado* do `objeto`, de acordo com a inscrição do *lugar*.

Transições

i. As *transições* em uma $RPIS_OC_{mO}$ especificam sobre quais circunstâncias uma situação termina e uma nova situação se torna ativa, pelas condições de *disparo*. A condição de *disparo* é representada por uma função booleana, $\varphi_{mO}(t_{j_mO})$ ou $\varphi_{mO}(t_{px_mO})$, a qual especifica quando é permitido acontecer uma mudança de *estado*

dos `objetos` que estão naquele *lugar*, ou seja, ser retirado do *lugar* atual $\left(p_{i_mO} \in {}^{\bullet}t_{j_{mO}} \text{ ou } p_{i_mO} \in {}^{\bullet}t_{px_mO}\right)$ e ser devidamente colocado em outro *lugar* $\left(p_{i_mO} \in t_{j_{mO}}^{\bullet} \text{ ou } p_{i_mO} \in t_{px_mO}^{\bullet}\right)$. A função de *disparo* $\varphi_{mO}\left(t_{px_mO}\right)$ memoriza e fornece um `método produzido`, como `interface produtora` de informação de sua respectiva `classe`; porém, nem toda *transição* e função de *disparo* pode representar um `método produzido`, como é o caso da *transição normal* t_{j_mO} e sua função de *disparo* $\varphi_{mO}\left(t_{j_mO}\right)$. Na $RPIS_OC_{mO}$ não existem *transições consumidoras*.

ii. Uma *transição normal* t_{j_mO} ou uma *transição consumidora* t_{px_mO} está *habilitada* sob uma *marcação* m_{k_mO}, se e somente se seus *pré-lugares* estão *marcados* com pelo menos uma *marca*[42], isto é:

$$Hab\left(t_{j_mO}, m_{k_mO}\right) = V \Leftrightarrow \left(\forall\, p_{i_mO} \in {}^{\bullet}t_{j_mO} \mid m_{k_mO}\left(p_{i_mO}\right) > 0\right) \text{ e}$$

$$Hab\left(t_{px_mO}, m_{k_mO}\right) = V \Leftrightarrow \left(\forall\, p_{i_mO} \in {}^{\bullet}t_{px_mO} \mid m_{k_mO}\left(p_{i_mO}\right) > 0\right).$$

iii. Uma *transição normal* t_{j_mO} ou uma *transição consumidora* t_{px_mO} *dispara* imediatamente, se e somente se está *habilitada*, $Hab\left(t_{j_mO}, m_{k_mO}\right) = V$ ou $Hab\left(t_{px_mO}, m_{k_mO}\right) = V$ e sua condição de *disparo* é satisfeita, $\varphi_{mO}\left(t_{j_mO}\right) = V$ ou $\varphi_{mO}\left(t_{px_mO}\right) = V$, ($V$ = Verdadeiro). O *disparo* remove uma *marca* de todos os *pré-lugares* ${}^{\bullet}t_{j_mO}$ ou ${}^{\bullet}t_{px_mO}$ da respectiva *transição* e coloca uma *marca* em todos os *pós-lugares* $t_{j_mO}^{\bullet}$ ou $t_{px_mO}^{\bullet}$ da respectiva *transição*. Significando, desta forma, que houve uma mudança de *estado* de um determinado `objeto` desta `classe` (o qual é representado por uma *marca*).

iv. Todas as *transições* podem *disparar* se não estão em conflito com *disparos* de outras *transições* (duas *transições* estão em conflito se o *disparo* de uma delas desabilita outra).

v. É suposto que o *disparo* de uma *transição* não leva tempo algum (o tempo de *disparo* de uma *transição* é zero ou o *disparo* é instantâneo). Isto também significa que uma mudança no valor do sinal de entrada da função $\varphi_{mO}\left(t_{j_mO}\right)$ ou $\varphi_{mO}\left(t_{px_mO}\right)$, não é

[42] Uma *marca* no *lugar* de uma sub-rede $RPIS_OC_{mO}$ representa um `objeto` da `classe` m do Objeto de Controle, em um determinado *estado* definido pela inscrição do correspondente *lugar*.

considerada que ocorra durante o processo de *disparo*.

vi. Quando não é especificada a condição de *disparo* na *transição*, significa que seu *disparo* depende apenas que seu(s) pré-*lugar (es)* estejam marcados e que seu(s) pós *lugar(es)* não estejam *marcados*.

3.4.4 Representação Gráfica de uma $RPIS_OC_{mO}$

Da mesma forma que em uma RP, os *lugares* de uma $RPIS_OC_{mO}$ são representados por círculos, *transições* por barras, a relação de fluxo por *arcos*, as *marcas* (que representam os `objetos` da `classe`, em um determinado *estado*) por pontos no círculo do correspondente *lugar* e as *transições* são rotuladas com sua função de *disparo*, a qual pode representar um `método` fornecido às `classes` do Sistema de Controle (C_{nS}).

Em adição, definem-se a representação gráfica de dois tipos de *transição* e de um *lugar*: uma *transição produtora* que representa a `interface produtora` do `método` (t_{px_mO}), outra para a *transição normal* (t_{j_mO}) e o *lugar* p_{i_mO}. A Figura 3.8 a seguir, ilustra a representação gráfica e seus rótulos das *transições* e *lugares* de uma $RPIS_OC_{mO}$.

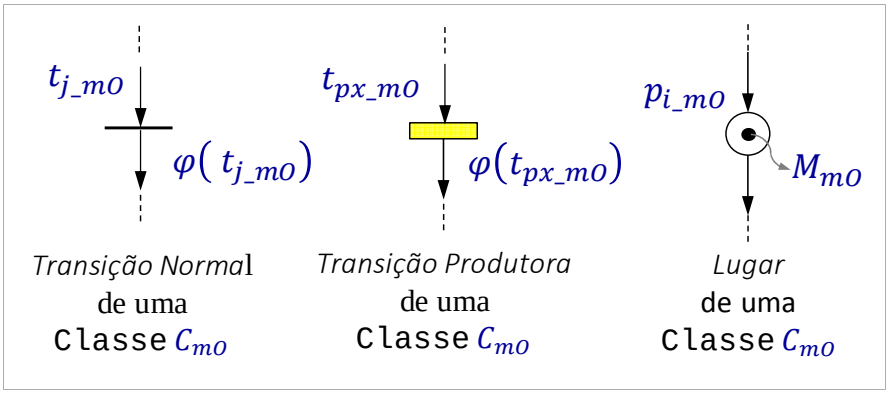


Figura 3.8 – Representação gráfica e rótulos das *transições* e *lugar* de uma $RPIS_OC_{mO}$[43].

A seguir, da mesma forma, define-se uma $RPIS_SC_{nS}$ em termos de estrutura, semântica, comportamento e representação gráfica.

[43] Na Figura 3.8, uma vez que as *transições* já estão identificadas com seus rótulos, prescinde-se da identificação da função de *disparo*. Portanto, em vez da utilização de: $\varphi_{mO}(t_{j_mO})$ e $\varphi_{mO}(t_{px_mO})$, utiliza-se: $\varphi(t_{j_mO})$ e $\varphi(t_{px_mO})$.

3.4.5 Estrutura de uma $RPIS_SC_{nS}$

Definição 3.15 – Sub Rede de Petri Interpretada por Sinais - Orientada a Objetos Marcada do Sistema de Controle - $RPIS_SC_{nS}$

Uma $RPIS_SC_{nS}$ é constituída pela 9-tupla $RPIS_SC_{nS} = \langle SC_{nS}, RP_{nS}, V_{nS}, I, O, \varphi_{nS}, \omega_{nS}, \Omega_{nS}, M_{nS} \rangle$, onde:

SC_{nS} É a `classe` de número n do Sistema de Controle constituída pela $RPIS_SC_{nS}$. Sendo que $n \in \mathbb{N}^*$ e S é a inscrição que designa a sub-rede como pertencente ao Sistema de Controle. Também as `classes` do Sistema de Controle serão identificadas individualmente como C_{nS}.

RP_{nS} É uma RP Ordinária da `classe` de número n do Sistema de Controle definida pela 5-tupla $\langle P_{nS}, T_{TnS}, F_{nS}, Pre_{nS}, Post_{nS} \rangle$, onde a multiplicidade ou *peso* dos *arcos* é sempre um, ou seja, os *arcos* não são ponderados, onde:

P_{nS} $P_{nS} = \{p_{1_nS}, p_{2_nS}, \dots p_{i_nS}\}$, é um conjunto finito e não vazio de *lugares* da `classe` do Sistema de Controle.

T_{TnS} É o conjunto de todos os tipos de *transição* definidos para esta `classe`, sendo:
$T_{TnS} = T_{nS} \cup T_{C_{nS}} \cup T_{P_{nS}}$,
$T_{nS} \cap T_{C_nS} \cap T_{P_nS} = \emptyset$,
$T_{nS} \cup T_{C_nS} \cup T_{P_{nS}} \neq \emptyset$, onde:

T_{nS}: Conjunto de *transições* da sub-rede $RPIS_OC_{nS}$ de número n do Sistema de Controle (inscrição S), que não são *produtoras* nem são *consumidoras*, onde:
$T_{nS} = \{t_{1_nS}, t_{2_nS}, \dots, t_{j_nS}\}$, sendo:

t_{j_nS}: *Transição* de número j da `classe` (sub-rede) de número n do Objeto de Controle (inscrição S), que não é *produtora* nem *consumidora*, denominada como *transição normal* da `classe` C_{nS}. Com, $t_{j_nS} \in T_{nS}$.

T_{C_nS}: Conjunto de *transições* de uma $RPIS_SC_{nS}$ que `consome` (utiliza) métodos vindos de outras `classes`, as quais descrevem um `serviço` que é chamado por esta `classe`, ou seja, é o conjunto de operações que a $RPIS_SC_{nS}$ utiliza de outras `classes`, onde:

$T_{C_nS} = \{t_{c1_nS},\ t_{c2_nS}, \dots, t_{cy_nS}\}$, sendo:

t_{cy_nS}: *Transição consumidora* de número y da `classe` (sub-rede) de número n do Sistema de Controle (inscrição S). A *transição consumidora* y tem uma `operação` que pode "chamar" outras `classes`. Com, $t_{cy_nS} \in T_{C_nS}$.

T_{P_nS}: Conjunto de *transições* da $RPIS_SC_{nS}$ que `produz` (fornece) um `método` somente a outras `classes` do Sistema de Controle, o qual descreve um `serviço` implementado por esta `classe`, ou seja, é o conjunto de `operações` que uma `classe` do Sistema de Controle disponibiliza e pode ser "chamada" por outras `classes` do Sistema de Controle, em nível heterárquico ou hierárquico, onde:

$T_{P_nS} = \{t_{p1_nS},\ t_{p2_nS}, \dots, t_{px_nS}\}$.

t_{px_nS} *Transição produtora* (fornecedora) de número x da `classe` ou sub-rede de número n do Sistema de Controle (inscrição s) que produz um método o qual pode ser *consumido* por outras `classes` do Sistema de Controle. A *transição produtora* de número x do Sistema de Controle tem uma `operação` que pode ser "chamada" por outras `classes` do Sistema de Controle. Com, $t_{px_nS} \in T_{P_nS}$.

F_{nS} $F_{nS} \subseteq (P_{mnS} \times T_{TnS}) \cup (T_{TnS} \times P_{nS})$, é um conjunto finito e não vazio de *arcos* que conectam *lugares* a *transições* ou *transições* a *lugares* da `classe` (sub-rede) $RPIS_OC_{nS}$.

Pre_{nS} Cada Pre_{nS} $^{\bullet}t_{j_nS}$, $^{\bullet}t_{cy_nS}$, $^{\bullet}t_{px_nS}$ das *transições* t_{j_nS}, t_{cy_nS}, $t_{px_nS} \in T_{TnS}$, contém todos os *lugares* que são conectados a t_{j_nS}, t_{cy_nS},

t_{px_nS} por *arcos* direcionados, dos *lugares* às *transições*:

$${}^{\bullet}t_{j_nS} = \{p_{i_nS} \in P_{nS}: (p_{i_nS}, t_{j_nS}) \in F_{nS}\},$$

$${}^{\bullet}t_{cy_nS} = \{p_{i_nS} \in P_{nS}: (p_{i_nS}, t_{cy_nS}) \in F_{nS}\},$$

$${}^{\bullet}t_{px_nS} = \{p_{i_nS} \in P_{nS}: (p_{i_mO}, t_{px_nS}) \in F_{nS}\}.$$

Os elementos de ${}^{\bullet}t_{j_nS}$, ${}^{\bullet}t_{cy_nS}$ e ${}^{\bullet}t_{px_nS}$ são denominados pré-*lugares* ou *lugares* de entrada, da `classe` C_{nS} do Sistema de Controle.

$Post_{nS}$ Cada $Post_{nS}$ $t^{\bullet}_{j_nS}$, $t^{\bullet}_{cy_nS}$, $t^{\bullet}_{px_nS}$ das transições t_{j_nS}, t_{cy_nS}, $t_{px_nS} \in T_{TnS}$, contém todos os *lugares* que são conectados a t_{j_nS}, t_{cy_nS}, t_{px_nS} por *arcos* direcionados, das *transições* aos *lugares*:

$$t^{\bullet}_{j_nS} = \{p_{i_nS} \in P_{nS}: (t_{j_nS}, p_{i_nS}) \in F_{nS}\},$$

$$t^{\bullet}_{cy_nS} = \{p_{i_nS} \in P_{nS}: (t_{cy_nS}, p_{i_nS}) \in F_{nS}\}.$$

$$t^{\bullet}_{px_nS} = \{p_{i_nS} \in P_{nS}: (t_{px_nS}, p_{i_mO}) \in F_{nS}\}.$$

Os elementos de $t^{\bullet}_{j_nS}$, $t^{\bullet}_{cy_nS}$ e $t^{\bullet}_{px_nS}$ são denominados pós-*lugares* ou *lugares* de saída, da `classe` C_{nS} do Sistema de Controle. Sendo, $Pre_{nS} \cap Post_{nS} = \emptyset$.

V_{nS} Conjunto de variáveis formais que pertence ao conjunto de `atributos` de uma `classe` de número n do Sistema de Controle. São considerados dois tipos. Um para `atributos` numéricos, isto é, variável que designa um tipo de dado numérico, de acordo com a IEC 61131, sendo que neste trabalho se limita a booleano, para todo o sinal $i_{|I|} \in I$, $o_{|O|} \in O$ e outro para o tipo **"NOME_DO_OBJETO_DO_SISTEMA_DE_CONTROLE".** Este tipo relaciona o nome dos `objetos` com os quais um determinado `objeto` do Sistema de Controle pode trocar informações pela sua interface, denominada *`Variável de Interface`*, definida como I_{nS-OL}, onde n é o número da `classe` do Sistema de Controle e L o número do `objeto` desta `classe`.

I $I = \{i_1, i_2, \ldots i_{|I|}\}$, é um conjunto finito dos sinais lógicos de entrada, de dimensão $|I|$. Sendo $|I| > 0$.

O $O = \{o_1, o_2, \dots o_{|O|}\}$. É um conjunto finito e não vazio dos sinais lógicos de saída, de dimensão $|O|$. Sendo $|O| > 0$.

φ_{nS} É um mapeamento associando cada *transição* $t_{j_nS} \in T_{nS}$ com uma função (ou condição) de *disparo* $\varphi_{nS}(t_{j_nS}) = f(i_1, i_2, \dots i_{|I|})$, cada *transição* $t_{cy_nS} \in T_{C_nS}$ com uma condição de *disparo* $\varphi_{nS}(t_{cy_nS}) = f(i_1, i_2, \dots i_{|I|})$ e também, associando cada *transição* $t_{px_nS} \in T_{P_nS}$ com uma condição de *disparo* $\varphi_{nS}(t_{px_nS}) = f(i_1, i_2, \dots i_{|I|})$. As condições de *disparo* $\varphi_{nS}(t_{j_nS})$, $\varphi_{nS}(t_{cy_nS})$ e $\varphi_{nS}(t_{px_nS})$ são funções *booleanas* em I, $f: \{0,1\}^{|I|} \longrightarrow \{0,1\}$.

ω_{nS} É um mapeamento associando cada *lugar* $p_{i_nS} \in P_{nS}$ com uma saída $\omega_{nS}(p_{i_nS}) \in \{0, 1, -\}^{|O|}$. Sob a ótica do paradigma de OO, considera-se este mapeamento uma ação de comando que faz parte do conjunto de `atributos` do `objeto`. Onde " $-$ " significa sinal não especificado.

Ω_{nS} É uma função de saída combinando a saída ω_{nS} com todos os *lugares marcados* de um *estado* alcançável. $\Omega_{nS}: \{0,1\}^{|P|} \longrightarrow \{0, 1, -, c\}^{|O|}$. Onde c significa sinal contraditório[44].

M_{nS} É o conjunto de *Marcações*, significando que cada uma representa o *estado* de um `objeto` da sub-rede do Sistema de Controle, sendo:

$M_{nS} = \{M_{0_nS}, M_{1_nS}, M_{1_nS}, \dots, M_{L_nS}\}$. Onde:

M_{0_nS}: é a *Marcação* inicial da `classe` n do Sistema de Controle.

M_{L_nS}: é a *Marcação* L da `classe` n do Sistema de Controle. Sendo:

$M_{L_nS} = \langle O_{L_nS}, V_{L_nS}, m_{L_nS}_l, \Omega_{L_nS} \rangle$. Onde:

O_{L_nS}: nome do `objeto` L, da `classe` n do Sistema de Controle, onde seu *estado* é representado pela *marcação* $m_{L_nS}_l$.

V_{L_nS}: instância de V_{nS} da `classe` n do Sistema de Controle, para o `objeto` L.

$m_{L_nS}_l$: é a *marcação* alcançável l do `objeto` L da `classe` n na

[44] A álgebra de sinais envolvida nesta definição é apresentada no Capítulo 4. Análise.

$RPIS_SC_{nS}$, de sua `classe` C_{nS}. Onde, $m_{L_nS}_l: P \longrightarrow \{0,1\}$, (cada *lugar* pode conter apenas uma *marca* de cada `objeto`, isto é, um único *estado* de cada vez).

Ω_{L_nS}: é a possível saída produzida por um `objeto` L de uma `classe` n do Sistema de Controle, durante sua evolução. Esta representa a soma ou conjunção das saídas de todas as *marcações* alcançáveis (diferentes *estados*) de um `objeto` desta `classe`[45].

Define-se ainda:

t_n: *transição* qualquer pertencente ao conjunto T_T;

p_m: *lugar* qualquer pertencente ao conjunto P_T;

T_T: conjunto de todas as *transições* de uma $RPIS_OO$;

P_T: conjunto de todos os *lugares* de uma $RPIS_OO$;

Onde:

$t_n \in T_T$

$p_m \in P_T$

$T_T = T_{TnS} \cup T_{TmO}$. Sendo que: $T_{TnS} \cap T_{TmO} = \emptyset;\ T_{TnS} \cup T_{TmO} \neq \emptyset;$

$P_T = P_{mO} \cup P_{nS}$. Sendo que: $P_{mO} \cap P_{nS} = \emptyset;\ P_{mO} \cup P_{nS} \neq \emptyset;$

3.4.6 Comportamento e Semântica de uma $RPIS_SC_{nS}$

Em uma $RPIS_SC_{nS}$ os *lugares* e *transições* têm sua semântica e comportamento ligeiramente diferentes da $RPIS_OC_{mO}$, conforme explanado a seguir:

Lugares

i. Da mesma forma que em uma $RPIS_OC_{mO}$, em uma $RPIS_SC_{nS}$ uma *marca* (`objeto`) em um *lugar* confirma o *estado* do `objeto` de acordo com a inscrição do respectivo *lugar*, indicado por uma *Marcação* (M_{L_nS}). Sendo que, o mesmo `objeto` também representa outra $RPIS_OO$, onde um *lugar* quando *marcado*

[45] A álgebra de sinais envolvida nesta definição é proposta no Capítulo 4. Análise.

representa o *estado* parcial *(marcação)* daquele objeto.

ii. Os *lugares* de um `objeto` em uma $RPIS_SC_{ns}$ quando *marcados*, além de representarem os *estados* dos `objetos` desta `classe`, também podem influenciar o meio ambiente, ou seja, enviar sinais para os `objetos` do Objeto de Controle ou enviar sinais a outros `objetos` do Sistema de Controle, via as correspondentes saídas digitais binárias memorizadas em $\omega_{ns}(p_{i_ns})$. As saídas binárias são consideradas como um tipo de `atributo` do `objeto` (ações de comando ativadas com a *marcação*).

iii. Também, em uma $RPIS_SC_{ns}$, os *lugares* não são temporizados.

Transições

i. Também em uma $RPIS_SC_{ns}$ as *transições*, por meio de uma função de *disparo,* determinam sobre quais características uma situação termina e uma nova situação se torna ativa. A função *booleana* de *disparo*, $\varphi_{ns}(t_{j_ns})$ ou $\varphi_{ns}(t_{cy_ns})$ ou $\varphi_{ns}(t_{px_ns})$, especifica quando é permitido acontecer uma mudança de *estado* dos `objetos` que estão naquele *lugar*, ou seja, serem retirados do *lugar* atual $p_{i_ns} \in {}^{\bullet}t_{j_ns}$ ou $p_{i_{ns}} \in {}^{\bullet}t_{cy_ns}$ ou $p_{i_ns} \in {}^{\bullet}t_{px_ns}$ e serem colocados em outro *lugar* $p_{i_ns} \in t_{j_ns}^{\bullet}$ ou $p_{i_ns} \in t_{cy_ns}^{\bullet}$ ou $p_{i_ns} \in t_{px_ns}^{\bullet}$.

ii. A função de *disparo* $\varphi_{ns}(t_{cy_ns})$ representa um `método` que é `consumido` ou utilizado pela *transição* t_{cy_ns}, como `interface consumidora`; a função de *disparo* $\varphi_{ns}(t_{px_ns})$ representa um `método` que é produzido ou fornecido pela *transição* t_{px_ns}, como `interface produtora` e a função de *disparo* $\varphi_{ns}(t_{j_ns})$ é somente uma condição de *disparo* da *transição* t_{j_ns}, esta não representa uma `interface` de comunicação, não é `produtora` nem `consumidora`.

iii. Também nesta sub-rede, as *transições,* as funções de *disparo* e os *arcos* não são temporizados e o *disparo* das *transições* é considerado instantâneo.

iv. Uma *transição* t_{j_ns} ou t_{cy_ns} ou t_{px_ns} está *habilitada* sob uma *marcação* m_{l_ns} , se e

somente se seus *pré-lugares* estão *marcados* com pelo menos uma *marca*[46], ou seja:

$Hab(t_{j_ns}, m_{l_ns}) = V \Leftrightarrow (\forall\, p_{i_ns} \in {}^{\bullet}t_{j_ns} \mid m_{l_ns}(p_{i_ns}) > 0)$ ou

$Hab(t_{cy_ns}, m_{l_ns}) = V \Leftrightarrow (\forall\, p_{i_ns} \in {}^{\bullet}t_{cy_ns} \mid m_{l_ns}(p_{i_ns}) > 0)$ ou

$Hab(t_{px_ns}, m_{l_ns}) = V \Leftrightarrow (\forall\, p_{i_ns} \in {}^{\bullet}t_{px_ns} \mid m_{l_ns}(p_{i_ns}) > 0)$.

v. Uma transição t_{j_ns} ou t_{cy_ns} ou t_{px_ns} *dispara* imediatamente, se e somente se está *habilitada*, $Hab(t_{j_ns}, m_{l_ns}) = V$ ou $Hab(t_{cy_ns}, m_{l_ns}) = V$ ou $Hab(t_{px_ns}, m_{l_ns}) = V$ e sua condição de *disparo* é satisfeita, $\varphi_{ns}(t_{j_ns}) = V$ ou $\varphi_{ns}(t_{cy_ns}) = V$ ou $\varphi_{ns}(t_{px_ns}) = V$, ($V$ = Verdadeiro). O *disparo* remove uma *marca* (`objeto`) de todos os pré-*lugares* ${}^{\bullet}t_{j_ns}$ ou ${}^{\bullet}t_{cy_ns}$ ou ${}^{\bullet}t_{px_ns}$, da respectiva transição e coloca uma *marca* em todos os pós-*lugares* $t_{j_ns}^{\bullet}$ ou $t_{cy_ns}^{\bullet}$ ou $t_{px_ns}^{\bullet}$, da respectiva *transição*. Significando, desta forma, que houve uma mudança de *estado* de um determinado `objeto` desta `classe` (o qual é representado por uma *marca*).

vi. Todas as *transições* podem *disparar* se não estão em conflito com outras *transições* (duas *transições* estão em conflito se o *disparo* de uma delas *desabilita* outra).

vii. É suposto que o *disparo* de uma *transição* não leva tempo algum (o tempo de *disparo* de uma *transição* é zero ou o *disparo* é instantâneo). Isto também significa que uma mudança no valor do sinal de entrada da função $\varphi_{ns}(t_{j_ns})$ ou $\varphi_{ns}(t_{cy_ns})$ ou $\varphi_{ns}(t_{px_ns})$, não é considerada ocorrer durante o processo de *disparo*.

viii. Quando não é especificada a condição de *disparo* na *transição*, significa que seu *disparo* depende apenas que seu(s) pré-*lugar(es)* estejam *marcados* e que seu(s) pós-*lugar(es)* não estejam *marcados*.

3.4.7 Representação Gráfica de uma $RPIS_SC_{ns}$

Igualmente, os *lugares* de uma $RPIS_SC_{ns}$ são representados por círculos, *transições* por barras, a relação de fluxo por *arcos*, as *marcas* (que representam os `objetos` da

[46] Uma *marca* no *lugar* de uma sub-rede $RPIS_SC_{ns}$ representa um `objeto` da `classe` n do Sistema de Controle, em um determinado *estado* definido pela inscrição do correspondente *lugar*.

`classe` e ativação da função de saída $\omega_{ns}(p_{i_ns})$, em um determinado *estado*) por pontos no círculo do correspondente *lugar* e as *transições* são rotuladas com sua específica função de *disparo*, a qual pode representar um `método` fornecido às `classes` do Sistema de Controle (C_{ns}).

Definem-se então, três tipos de *transição* e um *lugar*, sendo: uma *transição* que representa a `interface consumidora` do `método` (t_{cy_ns}), outra que representa a `interface produtora` (t_{px_ns}), outra que representa a *transição* normal (t_{j_ns}) e o *lugar* p_{i_ns} desta sub-rede. A Figura 3.9 ilustra a representação gráfica e seus rótulos das *transições* e *lugares* do modelo em $RPIS_SC_{ns}$.

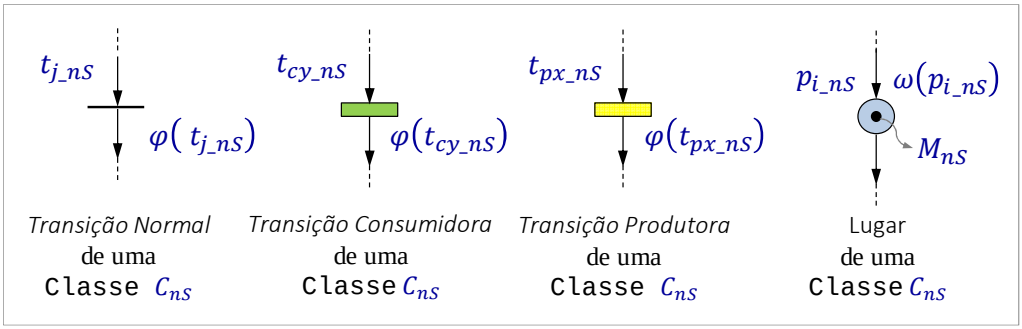


Figura 3.9 – Representação gráfica e rótulos das *transições* e *lugares* de uma $RPIS_SC_{ns}$ [47].

3.4.8 `Interfaces` e Comunicação entre `Classes`

Dentro dos conceitos de SSOO e da UML, a comunicação entre `classes` e `objetos` se dá entre as `interfaces` dos mesmos. Para que esta comunicação ocorra, não é necessário conhecer a estrutura interna do `objeto`, a qual pode ficar oculta, tornando a visão externa suficiente para que a comunicação aconteça (princípio do encapsulamento em OO). Uma `interface` corresponde a um conjunto de `serviços` fornecidos e requeridos por uma `classe`, um subsistema, componente ou um sistema. Um `serviço` é composto de sua especificação e de seu `método`. A especificação define o que o `serviço` realiza e, o `método` corresponde ao modo de realizar o `serviço`, ou seja, é a implementação de uma `operação`. De acordo com [07] e no contexto deste

[47] Da mesma forma que em uma $RPIS_OC_{mo}$, uma vez que o *lugar* as *transições* já estão identificadas com seus rótulos, prescinde-se da identificação da função de *disparo* e da função de saída. Portanto, em vez da utilização de: $\omega_{ns}(p_{i_ns})$, $\varphi_{ns}(t_{j_ns})$, $\varphi_{ns}(t_{cy_ns})$ e $\varphi_{ns}(t_{px_ns})$, utiliza-se: $\omega(p_{i_ns})$, $\varphi(t_{j_ns})$, $\varphi(t_{cy_ns})$ e $\varphi(t_{px_ns})$.

trabalho, os principais objetivos em relação ao conceito de interface são: especificar as operações que uma classe deve implementar e facilitar o desacoplamento entre os elementos de um sistema. A versão 2.0 da UML especifica que as interfaces podem ser de dois tipos: interface fornecida e interface requerida. Neste trabalho utilizam-se os termos: interface produtora e interface consumidora, em analogia ao modelo de comunicação em redes industriais, Produtor-Consumidor[48].

Uma interface produtora descreve um serviço implementado por uma classe. É o conjunto de operações que uma determinada classe oferece a outras classes, ou seja, é uma operação que pode ser "chamada" por objetos de outras classes. É a interface produtora da informação. Nesta proposta, tanto as classes do Objeto de Controle como as do Sistema de Controle podem disponibilizar uma interface produtora. No caso do Objeto de Controle, isto é bastante claro, uma vez que seus objetos informam o *estado* dos dispositivos reais do processo, por meio do envio de sinais binários de entrada. No caso do Sistema de Controle, seus objetos podem também disponibilizar sinais de entrada a outro objeto, em um nível distribuído ou hierárquico da estrutura do Sistema de Controle.

Uma interface consumidora descreve um serviço que outras classes devem fornecer para que seja utilizada pela mesma, ou seja, é o conjunto de operações que os objetos desta classe podem utilizar de outros objetos de outras classes. É uma operação que pode requerer ou "chamar" serviços de outras classes. É a interface consumidora da informação. Nesta proposta, somente as classes e objetos do Sistema de Controle possuem interface consumidora.

Os objetos de uma classe se comunicam por meio das chamadas de métodos que são modeladas pela fusão dinâmica de *transições*[49] e da ação de comando dos *lugares*.

A fusão dinâmica de *transições* pode ser utilizada entre objetos do Objeto de Controle e objetos do Sistema de Controle, como Interface Produtora e

[48] Também denominado *publisher-subscriber* (publicador-assinante). Neste modelo os dados possuem um identificador único, origem ou destino. Dessa forma, todos os nós podem ser sincronizados, ou seja, vários nós (produtores) podem transmitir dados para outros nós (consumidores).

[49] De acordo com [60], [105], [110] *apud* [54] e [126], a modelagem das interfaces entre modelos em RP pode ser feita em uma das principais formas, dentre outras: fusão de *lugares*, comunicação por *arcos habilitadores* ou fusão de *transições*. Neste trabalho optou-se pela fusão de *transições*, devido às vantagens e características apresentadas em [12], [54] e [126].

`Interface Consumidora` respectivamente; como também, entre `objetos` do Sistema de Controle. A fusão de *transições* de diferentes `classes` é *habilitada* se e somente se as *transições* das `classes` envolvidas estão *habilitadas* pelo cumprimento de suas regras de *disparo*. Quando *habilitadas*, todas *disparam* como se fossem uma única *transição*, considerando-se que seus *disparos* são atômicos e que durante os *disparos* não há mudança de *estado* ou de sinais das funções de *habilitação*.

A ação de comando dos *lugares* é utilizada somente para os `objetos` do Sistema de Controle, como Ação de Comando a qual faz parte do conjunto de `atributos` do `objeto`. Assim, define-se a seguir, `Interface Produtora` para uma $RPIS_OC_{mO}$.

Definição 3.16 – Interface `Produtora` de uma `Classe` do Objeto de Controle - T_{P_mO}

A `interface produtora` de uma $RPIS_OC_{mO}$ pertencente ao Objeto de Controle é representada pela *transição* t_{px_mO}, sendo que $t_{px_mO} \in T_{P_mO}$, conforme consta na Definição 3.14. Uma *transição* t_{px_mO} especifica um `serviço` disponibilizado pela sua respectiva `classe`, o qual pode ser requerido somente por um ou mais `objetos` de uma ou mais `classes` do Sistema de Controle.

Define-se também `Interface Consumidora` e `Interface Produtora` para uma $RPIS_SC_{nS}$.

Definição 3.17 – Interface `Consumidora` de uma `Classe` do Sistema de Controle - T_{C_nS}

A `interface consumidora` de uma $RPIS_SC_{nS}$ pertencente ao Sistema de Controle é representada por t_{cn_nS}, sendo que $t_{cn_nS} \in T_{C_nS}$, conforme Definição 3.15. Uma *transição* t_{cn_nS} especifica um ou mais `serviços` que podem ser utilizados pela sua classe, o qual é disponibilizado por `objetos` de uma `classe` do Objeto de Controle ou de uma `classe` do Sistema de Controle.

Definição 3.18 – Interface `Produtora` de uma `Classe` do Sistema de Controle - T_{P_nS}

A `interface produtora` de uma $RPIS_SC_{nS}$ pertencente ao Sistema de Controle é representada por t_{px_nS}, sendo que $t_{px_nS} \in T_{P_nS}$, conforme consta na Definição 3.15. Uma *transição* t_{px_nS} especifica um `serviço` disponibilizado pela sua `classe`, o qual pode ser requerido por um ou mais `objetos` de uma ou mais `classes` do Sistema de Controle.

Torna-se necessário definir ainda, uma especificação para identificação das `classes` que têm suas `interfaces` envolvidas nas `operações` entre os `objetos` destas `classes`, como também, identificar o nome do `objeto` da `classe` que armazena a `interface` que contém o `método` a ser utilizado.

Definição 3.19 – Identificação das Associações das Interfaces entre as Classes do Objeto de Controle e as Classes do Sistema de Controle.

- Cada *transição consumidora* $t_{cy_nS} \in T_{C_nS}$ da `classe` C_{nS} do Sistema de Controle está associada pelo menos a uma *transição produtora* $t_{px_mO} \in T_{P_mO}$ da `classe` C_{mO}, do Objeto de Controle.
- Tal associação indica que a *transição* t_{cy_nS} requisita o `método` associado à *transição* t_{px_mO}.
- Cada *transição* $t_{cy_nS} \in T_{C_nS}$ pode estar associada a uma variável de `interface` I_{mO-OK} do conjunto de variáveis V_{mO} da `classe` do Objeto de Controle.
- A inscrição $t_{cy_nS} \rightarrow t_{px_mO}_I_{mO-OK}$ declara que para um `objeto` da `classe` C_{nS}, a *transição* t_{cy_nS} chama o `método` associado à t_{px_mO}, cujo nome está armazenado na variável I_{mO-OK} do `objeto` de número K da `classe` de número m do Objeto de Controle.

Têm-se ainda, o caso em uma estrutura hierárquica ou heterárquica do Sistema de Controle, onde podem coexistir diferentes `objetos` do Sistema de Controle interagindo entre si em diferentes níveis hierárquicos, por meio de suas `interfaces`, como `interfaces produtoras` ou `consumidoras`. A seguir, apresenta-se esta definição.

Definição 3.20 – Identificação das Associações das Interfaces entre as Classes do Sistema de Controle.

- Cada *transição consumidora* $t_{cy_nS} \in T_{C_nS}$ da `classe` C_{nS} do Sistema de Controle pode estar associada a uma *transição produtora* $t_{px_nS} \in T_{P_nS}$ da `classe` C_{nS}, do Sistema de Controle.
- Esta associação indica que a *transição* t_{cy_nS} chama o `método` associado à t_{px_nS}.
- Cada *transição* $t_{cy_nS} \in T_{C_nS}$ está associada a uma variável de `interface` I_{L_nS} do conjunto de variáveis V_{nS} .

A inscrição $t_{cy_nS} \rightarrow t_{px_nS}_I_{nS-OL}$ declara que para um `objeto` da `classe` C_{nS}, a *transição* t_{cy_nS} chama o `método` associado à t_{px_nS}, cujo nome está armazenado na variável I_{nS-OL} do `objeto` de número L da `classe` de número n do Sistema de Controle.

3.4.9 Composição dos Modelos Integrados do Sistema

Conforme apresentado anteriormente, nesta proposta o modelo completo do sistema é a composição dos modelos do Sistema de Controle e do Objeto de Controle, os quais interagem entre si por meio de suas interfaces, de acordo com as chamadas de `métodos` definidas, ou seja, pela ação de comando dos *lugares* dos `objetos` do Sistema de Controle e pela fusão de *transições* entre os modelos. Como a fusão de *transições* é dinâmica, isto significa que um `método` produzido por uma `classe` pode ser `consumido` por diferentes `classes`, como também, uma `classe` pode consumir `métodos` de diferentes outras `classes`; o que pode levar a uma estrutura e *marcação*

variáveis do sistema em função de diferentes *marcações* iniciais. Isto faz com que não seja possível a análise dos modelos, com as técnicas já formalizadas e conhecidas para a RP Subjacente. Então, além de se fazer uma análise por partes (dos `objetos` do Objeto de Controle e do Sistema de Controle), busca-se também uma estrutura fixa que representa a estrutura discreta do modelo do sistema e o seu comportamento a partir de uma determinada *marcação* inicial, a qual deve compreender a interação dos `objetos` do Objeto de Controle e dos `objetos` do Sistema de Controle.

De acordo com [36], uma RP de Alto Nível[50] pode ser considerada como uma versão estruturalmente "dobrada" *(folded)* de uma RP Ordinária, se o número de cores (diferentes tipos de *marcas*) é finito. Isto quer dizer que uma RP de Alto Nível pode ser "desdobrada" *(unfolded)* em uma RP Ordinária. Analogamente, uma vez que uma $RPIS_OC_{mo}$ ou uma $RPIS_SC_{ns}$ pode comportar uma quantidade definida e não variável de `objetos` devidamente identificados e representados pelas suas *marcas*, pode-se então tratar estas sub-redes (`classes`) como uma RP de alto nível estruturalmente dobrada, que pode ser desdobrada em uma RP Ordinária ou RP Subjacente. Desse modo, pode-se fazer uso do desdobramento da sub-rede para construção de um modelo integrado, uma vez que o número de `objetos` de cada `classe` é conhecido, constante e que a sub-rede de cada `classe` é limitada em k *marcas*. Assim, nesta abordagem considera-se que RP de cada `objeto` (tanto do Objeto de Controle como do Sistema de Controle) seja $k = 1$, isto é, a RP é limitada e segura, também denominada por alguns autores de RP Ordinária Marcada Binária. Então, para o procedimento do desdobramento das sub-redes e composição do modelo integrado, utiliza-se um conceito similar ao utilizado em [12], porém com algumas modificações.

3.4.9.1 Procedimento para o Desdobramento das Sub-Redes e Composição dos Modelos Integrados

Passo 1 Identificação da fusão das *Transições* das `classes`.

[50] Em uma RP de Alto Nível, a *marca* pode ser individualizada permitindo o transporte de informação durante a evolução da RP, ou seja, a estrutura de dados é modelada pela própria *marca* por meio de `objetos` ou constantes (RP a Objetos ou RP Predicado-Transição) ou por intermédio de *cores* (RP Colorida), aumentando dessa forma o nível de abstração do modelo [05].

Para cada *transição consumidora* t_{cy_nS} que requisita uma *transição produtora* t_{px_mO}, combina-se a varável de interface I_{mO-OK}. Desta forma, tem-se: $t_{cy_nS} \rightarrow t_{px_mO}.I_{mO-OK}$. Onde, I_{mO-OK} é a variável interface I onde está armazenado o nome do `objeto` K da `classe` m do Objeto de Controle. Procede-se da mesma forma para cada t_{cy_nS} que requisita um `método` para uma t_{px_nS}, do Sistema de Controle. Neste caso, tem-se $t_{cy_nS} \rightarrow t_{px_nS}.I_{nS-OL}$.

Passo 2 Desdobramento das `classes` em `objetos` e sua renomeação para o nível de `objeto`.

Tanto para o Objeto de Controle como para o Sistema de Controle cria-se separadamente, para cada `classe` definida, uma $RPIS_OO$ para cada `objeto` existente, em um determinado *estado* inicial (*marcação* inicial). Ou seja, cada sub-rede $RPIS_OC_{mO}$ e $RPIS_SC_{nS}$ é representada novamente na mesma quantidade de `objetos`, que são instanciados em sua *marcação* inicial. Depois, faz-se a renomeação dos componentes para o nível de `objeto`. Renomeiam-se os *lugares* e as *transições* de cada `objeto`, conforme Tabelas 3.2(a) e 3.2(b). Ou seja: da Tabela 3.2 (a) no nível de `classe`, para a Tabela 3.2(b) no nível de `objeto`.

Tabela 3.2 (a) – Nomenclatura e significado dos *lugares*, *transições* e *marcas* no nível de `classe`. Passo **2**: Renomeação dos *Lugares* e *Transições* das `Classes`, para abordagem como `Objetos`.

De:	Significado para `Classe`:
t_{j_mO}	*Transição* normal j da `classe` m do Objeto de Controle.
t_{j_nS}	*Transição* normal j da `classe` n do Sistema de Controle.
t_{px_mO}	*Transição produtora* x da `classe` m do Objeto de Controle.
t_{cy_nS}	*Transição consumidora* y da `classe` n do Sistema de Controle.
t_{px_nS}	*Transição produtora* x da `classe` n do Sistema de Controle.
p_{i_nS}	*Lugar* i da `classe` n do Sistema de Controle.
p_{i_mO}	*Lugar* i da `classe` m do Objeto de Controle.
OC_{mO}	`Classe` de número m, do `Objeto` de Controle.
SC_{nS}	`Classe` de número n, do Sistema de Controle.

Tabela 3.2 (b) – Nomenclatura e significado dos *lugares*, *transições* e *marcas* para o nível de `objeto`.
Passo **2**: Renomeação dos *Lugares* e *Transições* das `Classes`, para abordagem como `Objetos`.

Para:	Significado para `Objeto`:
$t_{j_{mO-OK}}$	*Transição* normal j do `objeto` K, da `classe` m do Objeto de Controle.
t_{j_nS-OL}	*Transição* normal j do objeto L, da `classe` n do Sistema de Controle.
t_{px_mO-OK}	*Transição produtora* x do `objeto` K, da `classe` m do Objeto de Controle.
t_{cy_nS-OL}	*Transição consumidora* y do `objeto` L, da `classe` n do Sistema de Controle.
t_{px_nS-OL}	*Transição produtora* x do `objeto` L, da `classe` n do Sistema de Controle.
p_{i_nS-OL}	*Lugar* i do `objeto` L, da `classe` n do Sistema de Controle.
p_{i_mO-OK}	*Lugar* i do `objeto` K, da `classe` m do Objeto de Controle.
$OC_{mO_}OK$	`Objeto` K da `classe` de número m, do Objeto de Controle de um sistema modelado em *RPIS_OO*.
$SC_{nS_}OL$	`Objeto` L da `classe` de número n, do Sistema de Controle de um sistema modelado em *RPIS_OO*.

Passo 3 Fusão e renomeação das *transições* dos `objetos` do Objeto de Controle.

Para toda *transição produtora* na RP de cada `objeto` do Objeto de Controle, copiam-se as *transições produtoras* que fornecem os `métodos` requisitados pela mesma. Na RP de cada `objeto` do Objeto de Controle, cada *transição produtora* é renomeada de t_{px_mO-OK} para $t_{px_mO-OK} \wedge t_{cy_nS-OL}$. Como também, a função de *habilitação* de cada *transição produtora* passa de $\varphi(t_{px_mO-OK})$ para $\varphi(t_{px_mO-OK}) \wedge \varphi(t_{cy_nS-OL})$.

Passo 4 Fusão e renomeação das *transições* dos `objetos` do Sistema de Controle.

Da mesma forma, para os `objetos` do Sistema de Controle, para cada *transição consumidora* copiam-se na mesma RP as *transições produtoras* que fornecem os `métodos` requisitados pela mesma. Na RP de cada `objeto` do

Sistema de Controle, cada *transição consumidora* é renomeada de t_{cy_nS-OL} para $t_{cy_nS-OL} \wedge t_{px_mO-OK}$ e a função de *habilitação* de cada *transição consumidora* passa de $\varphi(t_{cy_nS-OL})$ para $\varphi(t_{cy_nS-OL}) \wedge \varphi(t_{px_mO-OK})$.

Passo 5 Modelo integrado dos `objetos` do Sistema de Controle.

O modelo integrado dos `objetos` do Sistema de Controle se dá pela conclusão da RP conforme Passo 4. Estes modelos, assim obtidos, representam as $RPIS_SC_{nS}_OL$ que se comunicam com as outras $RPIS_OC_{mO}_OK$ do sistema, como também, apresentam as saídas binárias mapeadas por $\omega(p_{i_{nS-OL}})$. Dessa forma, as RP integradas possuem uma estrutura e um comportamento (dependendo de sua *marcação* inicial) similar a uma RPIS (considerando-se que as outras informações podem ser mantidas encapsuladas) a qual pode ser devidamente analisada com algumas técnicas e métodos já investigados na área acadêmica. Da mesma forma, uma *RP* obtida por meio do Passo 3, a qual representa um `objeto` do Objeto de Controle, sob um ponto de vista discreto pode ser representada por uma RP Interpretada por Sinais, onde a análise pode ser feita pela sua correspondente RP Subjacente. Esta parte sobre a análise dos modelos é apresentada no Capítulo 4 - Análise.

3.5 Exemplo de Aplicação

Para elucidar os conceitos e definições apresentados, utiliza-se como exemplo de aplicação um sistema real ilustrado na Figura 3.10.

3.5.1 Descrição Informal do Sistema de Drenagem do Óleo de Infiltração do SHRV

Este sistema, dentro dos conceitos de componentização, pode ser considerado como um subsistema do SHRV, em uma UHE. Sua função é coletar e armazenar possíveis vazamentos de óleo das instalações e bombeá-lo de volta ao reservatório principal do SHRV, conforme ilustração da Figura 3.10.

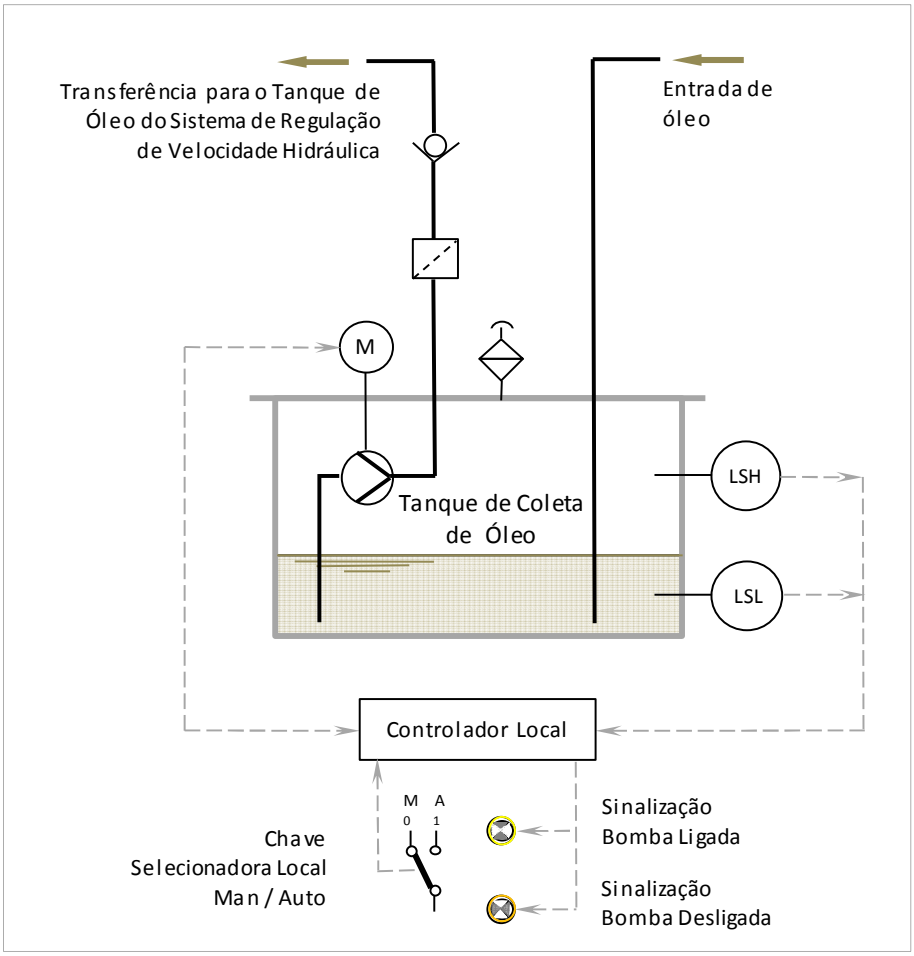


Figura 3.10 – Exemplo de aplicação - Sistema de Drenagem do Óleo de Infiltração do SHRV.

Seu funcionamento é elementar: quando o nível alto (LSH) é detectado, a bomba é ligada e assim permanece até que o nível baixo (LSL) seja atingido e a bomba é desligada. Existe uma chave selecionadora local, para escolher os modos manual ou automático, a qual, em modo manual, desabilita o controlador local.

O modo manual não será tratado neste exemplo. Além da abordagem e abstração do modelo do sistema ser discreta, não é foco deste exemplo considerar os dimensionamentos do tanque e da bomba, no que diz respeito a volume, vazão, pressão, tempos de enchimento e de esvaziamento.

Estão previstos os sinais digitais de entrada e saída descritos na Tabela 3.3.

Tabela 3.3 – Sinais Digitais de Entrada e de Saída para o sistema representado na Figura 3.10.

Entradas	**Descrição** (Nível Lógico 1)	**Saídas**	**Descrição** (Nível Lógico 1)
i_1	Chave em operação remota	o_1	Ligar bomba
i_2	Chave de nível baixo - LSL	o_2	Sinalizar bomba ligada
i_3	Chave de nível alto - LSH	o_3	Desligar bomba
i_4	Bomba ligada	o_4	Sinalizar bomba desligada
i_5	Bomba desligada	vo_1	Sinalizar pré-condições para operar bomba - satisfeitas [51]
i_6	Bomba não disponível	vo_2	Sinalizar pré-condições para ligar bomba - satisfeitas
i_7	Botão de Emergência acionado	vo_3	Sinalizar pré-condições para desligar bomba - satisfeitas
vi_1	Sistema pronto para operar (via rede) [52]		

Observação 3.1 - Textos de denominação das `classes`, `objetos` e `marcações`.

Neste trabalho, utiliza-se a seguinte convenção para os nomes de `classes` e `objetos`: os textos de denominação das `classes` e dos `objetos` se darão respectivamente da seguinte forma: `nome_da_classe` e **`NOME_DO_OBJETO`**. Ou seja: o nome da `classe` em letras minúsculas e nome do `objeto` em letras maiúsculas em negrito e ambos na fonte *Courier New*.

Observação 3.2 - Exemplos de aplicação da terminologia definida.

De acordo com a terminologia definida anteriormente, como exemplo, têm-se as seguintes inscrições para os modelos apresentados nesta aplicação:

$p_{i_mO} := p_{1_1O}$ *Lugar* p_1 da `classe 1` do Objeto de Controle (O indica Objeto de Controle).

[51] vo_i é um sinal de saída interno *(virtual output)* gerado e utilizado internamente no programa do CLP, para lógicas internas ou sinalização em IHM (Interface Humano Máquina).

[52] vi_i é um sinal de entrada interno *(virtual input)* recebido via rede de outros *IEDs.*

$t_{j_mO} := t_{2_1O}$ *Transição* normal t_2 da `classe` 1 do Objeto de Controle

$p_{i_ns} := p_{2_1S}$ *Lugar* p_2 da `classe` 1 do Sistema de Controle (S indica Sistema de Controle).

$t_{i_ns} := t_{6_1S}$ *Transição* t_6 da `classe` 1 do Sistema de Controle.

$C_{mO} := C_{1O}$ `Classe` 1 do Objeto de Controle (O).

$C_{nS} := C_{1S}$ `Classe` 1 do Sistema de Controle (S).

$M_{mO} := M_{1O}$ Conjunto de *Marcações* da `classe` 1 do Objeto de Controle.

$M_{K_mO} := M_{1_1O}$ *Marcação* 1 da `classe` 1 do Objeto de Controle.

$O_{K_mO} := O_{1_1O}$ Objeto 1 da classe 1 do Objeto de Controle.

$V_{mO} := V_{1O}$ Conjunto de variáveis formais que pertence ao conjunto de `atributos` da `classe` 1 do Objeto de Controle

$V_{K_mO} := V_{1_1O}$ Instância de V_{1O} para o `objeto` 1 da `classe` 1 do Objeto de Controle.

$m_{K_mO}_k := m_{3_2O}_1$ *Marcação* alcançável 1 do `objeto` 3 da `classe` 2 do Objeto de Controle.

$I_{mO-Ok} := I_{1O-O1}$ Variável de `interface` I da `classe` 1 do Objeto de Controle, onde está armazenado o nome do `objeto` 1 que está associado ao `método` que pode ser chamado por um `objeto` do Sistema de Controle.

$M_{nS} := M_{1S}$ Conjunto de *Marcações* da `classe` 1 do Sistema de Controle.

$M_{L_nS} := M_{1_1S}$ *Marcação* 1 da `classe` 1 do Sistema de Controle.

$O_{L_nS} := O_{1_1S}$ Objeto 1 da classe 1 do Sistema de Controle.

$V_{nS} := V_{1S}$ Conjunto de variáveis formais que pertence ao conjunto de `atributos` da `classe` 1 do Sistema de Controle.

$V_{L_nS} := V_{1_S}$ Conjunto de variáveis formais que pertence ao conjunto de `atributos` da `classe` 1 do Sistema de Controle.

$m_{L_nS}_l := m_{3_2O}_1$ *Marcação* alcançável 1 do `objeto` 3 da `classe` 2 do Sistema de Controle.

$I_{nS-OL} := I_{1S-O1}$ Variável de `interface` I da `classe` 1 do Sistema de Controle, onde está armazenado o nome do `objeto` 1 que está associado ao `método` que pode ser chamado por um `objeto` do Sistema de Controle.

3.5.2 Definição das `Classes` e `Objetos` do modelo do Sistema

Voltando-se ao exemplo de aplicação do sistema ilustrado na Figura 3.10 (Sistema de Drenagem do Óleo de Infiltração do SRVH), primeiramente faz-se a seleção das `classes` para o Objeto de Controle e para o Sistema de Controle. Uma possível solução de modelagem, de acordo com as definições apresentadas anteriormente e com a descrição informal do sistema, é a seguinte:

- `Classes` e `objetos` do Objeto de Controle:

`Classes`:

C_{1O} = `ch_l/r` (chave modo de operação local/remoto)

C_{2O} = `ch_n` (chave de nível)

C_{3O} = `mb` (conjunto motor-bomba)

`Objetos`:

O_{1_1O} = **`CH_L-R`**

O_{1_2O} = **`CH_N-B`** (chave de nível baixo)

O_{2_2O} = **`CH_N-A`** (chave de nível alto)

O_{1_3O} = **`MB`** (conjunto motor-bomba)

- `Classes` e `objetos` do Sistema de Controle:

`Classes`:

C_{1S} = `pr_con_op_b` (pré-condições para operar bomba)

C_{2S} = `pr_con_l_b` (pré-condições para ligar bomba)

C_{3S} = `pr_con_d_b` (pré-condições para desligar bomba)

C_{4S} = `ctr_b` (controle liga/desliga bomba)

`Objetos`:

O_{1_1S} = **`PR_CON_OP_B`** (pré-condições para operar bomba)

O_{1_2S} = **`PR_CON_L_B`** (pré-condições para ligar bomba)

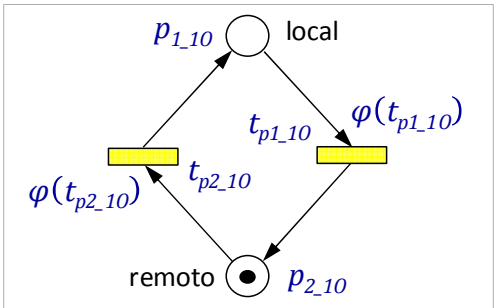
O_{1_3S} = **`PR_CON_D_B`** (pré-condições para desligar bomba)

O_{1_4S} = **`CTR_B`** (controle liga/desliga bomba)

A seguir, apresenta-se cada `classe` do modelo do sistema (Sistema de Drenagem do

Óleo de Infiltração do SRVH, representado na Figura 3.10), ilustrando-se suas sub-redes com seus `atributos` e `métodos`. Uma possível *marcação* dos `objetos`, `métodos`, *transições produtoras*, *transições consumidoras* e variáveis de `interface` serão apresentado na sequência.

3.5.2.1 `Classes` do Objeto de Controle



`Classe`: C_{1O} = `ch_l-r`

`Atributos`:
- Variáveis
  Booleanas: $V_{1O} = \{0, 1\}$
  Interfaces: $I_{mO-Ok} := I_{1O-O1} = O_{1_1O}$

`Métodos`:
t_{p1_1O}: ch na posição remoto
t_{p2_1O}: ch na posição local

`Objeto`:
$O_{K_mO} := O_{1_1O}$ = **`CH_L-R`**

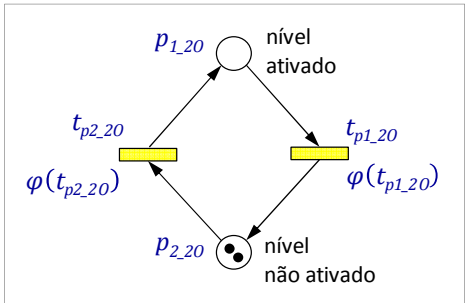
Marcação inicial da `classe` C_{1O}
$M_{0_mO} := M_{0_1O} = [0, 1]^T$

Figura 3.11 – Modelo da `Classe` C_{1O}: `ch_l-r` (chave modo local/remoto).

- Na Figura 3.11, o modelo da `Classe` C_{1O} é interpretado da seguinte forma:

- A `classe` possui dois *estados* possíveis: chave em modo local (*lugar* P_{1_1O}) ou em modo remoto (*lugar* P_{2_1O}).
- A `classe` está representada com um `objeto` (O_{1_1O} = **`CH_L-R`**) instanciado no *estado* remoto.
- A *transição* t_{p1_1O} indica um `método` fornecido por esta `classe`, o qual é "chave na posição remota", o qual é disponibilizado somente se sua função de habilitação $\varphi(t_{p1_1O})$ é satisfeita. Porém, pode ser disparada somente se também tiver uma *marca* em seu pré-*lugar* e nenhuma *marca* em seu pós-*lugar*.
- A *transição* t_{p2_1O} indica outro `método` fornecido por esta `classe`, que é "chave na posição local", o qual é disponibilizado somente se sua função de habilitação $\varphi(t_{p2_1O})$ é satisfeita. Também, pode ser disparada somente se ainda tiver uma *marca* em seu pré-*lugar* e nenhuma *marca* em seu pós-*lugar*.

- A variável de interface I_{1O-O1} armazena o nome do `objeto` O_{1_1O} que é igual à **CH_L-R**. que pode ser chamado por outras `classes.`
- A *marcação* inicial da `classe` é $M_{0_1O} = [0,1]^T$, ou seja, está no estado modo remoto.



`Classe:` C_{2O} = `ch_n`

`Atributos:`
- Variáveis
  Booleanas: $V_{2O} = \{0,1\}$
  Interfaces: $I_{mO-Ok} := I_{2O-O1} = O_{1_2O}$
  $I_{2O-O2} = O_{2_2O}$

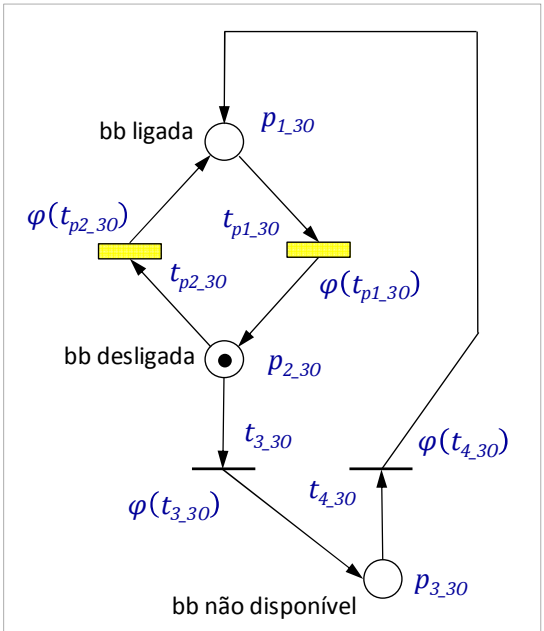
`Métodos:`
t_{p1_2O}: nível não ativado
t_{p2_2O}: nível ativado

`Objetos:`
$O_{K_mO} := O_{1_2O}$ = **CH_N-B**
$= O_{2_2O}$ = **CH_N-A**

Marcação da `classe` C_{2O}
$M_{0_mO} := M_{0_2O} = [0,2]^T$

Figura 3.12 – Modelo da `Classe:` C_{2O} = `ch_n` (chave de nível).

- Na Figura 3.12, a `classe` C_{2O} está representada com dois `objetos` (O_{1_2O} = **CH_N-B** e O_{2_2O} = **CH_N-A**) instanciados no *estado* nível não ativado.



`Classe:` C_{3O} = `mb`

`Atributos:`
- Variáveis
  Booleanas: $V_{3O} = \{0,1\}$
  Interfaces: $I_{mO-Ok} := I_{3O-O1} = O_{1_3O}$

`Métodos:`
t_{p1_3O}: bomba desligada
t_{p2_3O}: bomba ligada
t_{3_3O}: bomba não disponível
t_{4_3O}: bomba ligada

`Objeto:`
$O_{K_mO} := O_{1_3O}$ = **MB**

Marcação da `classe` C_{2O}
$M_{0_mO} := M_{0_3O} = [0,1,0]^T$

Figura 3.13 – Modelo da `Classe` C_{3O}: `mb` (conjunto motor-bomba).

3.5.2.2 `Classes` do Sistema de Controle

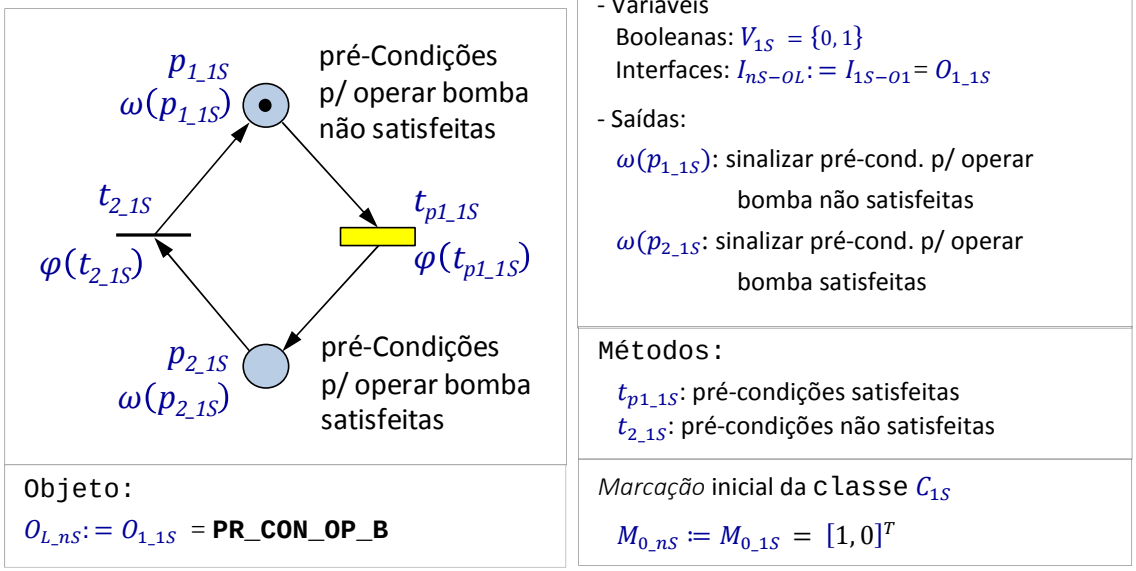
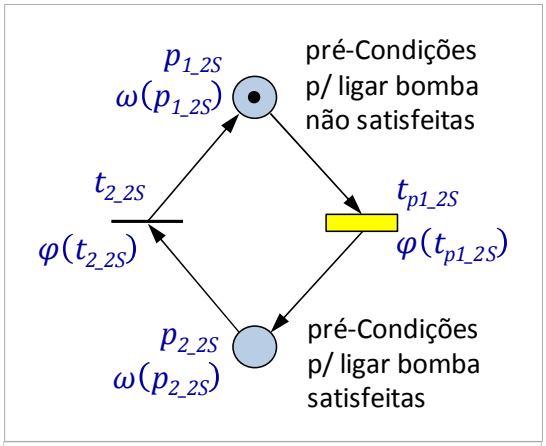


Figura 3.14 – Modelo da `classe` C_{1S} = `pr_con_op_b` (pré-condições para operar bomba).

- Na Figura 3.14, o modelo da `Classe` C_{1S} é interpretado da seguinte forma:

- A `classe` possui dois *estados* possíveis: pré-cond. p/ operar bomba não satisfeitas (*lugar* P_{1_1S}) ou em pré-cond. p/ operar bomba não satisfeitas (*lugar* P_{2_1S}).
- As funções de saída $\omega(P_{1_1S})$ e $\omega(P_{1_1S})$ representam ações de comando que fazem parte do conjunto de `atributos` do `objeto` instanciado neste *estado*.
- A `classe` está representada com um `objeto` (O_{1_1S} = **PR_CON_OP_B**) instanciado no *estado* pré-cond. p/ operar bomba não satisfeitas.
- A *transição* t_{p1_1S} indica um `método` fornecido por esta `classe`, o qual é "pré-condições satisfeitas", o qual é disponibilizado somente se sua função de habilitação $\varphi(t_{p1_1S})$ é satisfeita. Porém, pode ser disparada somente se também tiver uma *marca* em seu pré-*lugar* e nenhuma marca em seu pós-*lugar*.
- A *transição* t_{2_1S} indica uma *transição* normal (não é *produtora* nem *consumidora*), que é "pré-condições não satisfeitas", a qual pode ser disparada somente se sua função de habilitação $\varphi(t_{2_1S})$ é satisfeita, tiver uma *marca* em seu pré-*lugar* e nenhuma marca em seu pós-*lugar*.

- A variável de interface I_{1S-O1} armazena o nome do `objeto` O_{1_1S} que é igual à **PR_CON_OP_B**. que pode ser chamado por outras `classes`.
- A *marcação* inicial da `classe` é $M_{0_1S} = [1,0]^T$, ou seja, está no *estado* pré-condições para operar bomba não satisfeitas.



Objeto:
$O_{L_nS} := O_{1_2S}$ = **PR_CON_L_B**

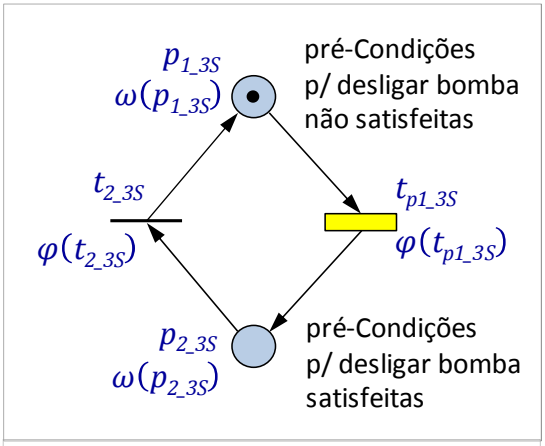
Classe: C_{2S} = pr_con_l_b

Atributos:
- Variáveis
  Booleanas: $V_{2S} = \{0,1\}$
  Interfaces: $I_{nS-OL} := I_{2S-O1} = O_{1_2S}$
- Saídas:
  $\omega(p_{1_2S})$: sinalizar pré-cond. p/ ligar bomba não satisfeitas
  $\omega(p_{2_2S}$: sinalizar pré-cond. p/ ligar bomba satisfeitas

Métodos:
t_{p1_2S}: pré-condições satisfeitas
t_{2_2S}: pré-condições não satisfeitas

Marcação inicial da `classe` C_{2S}
$M_{0_nS} := M_{0_2S} = [1,0]^T$

Figura 3.15 – Modelo da `classe` C_{2S} = pr_con_l_b (pré-condições para ligar bomba).



Objeto:
$O_{L_nS} := O_{1_3S}$ = **PR_CON_D_B**

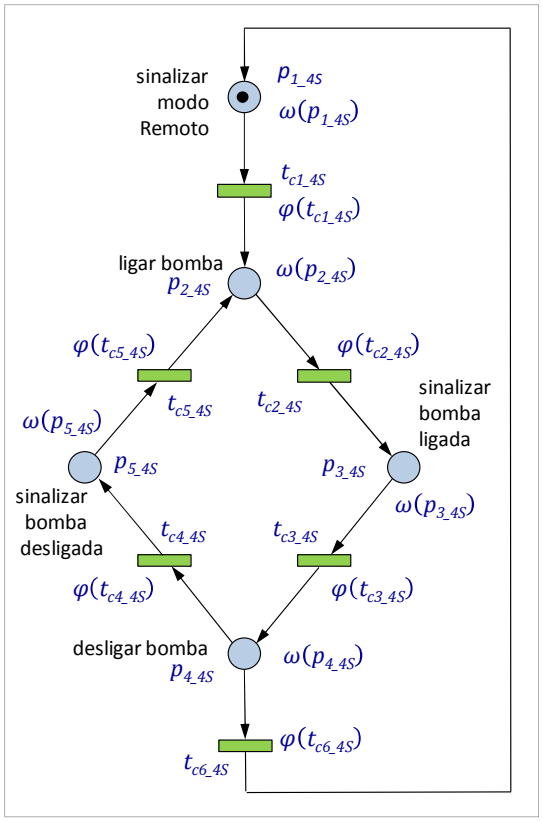
Classe: C_{3S} = pr_con_d_b

Atributos:
- Variáveis
  Booleanas: $V_{3S} = \{0,1\}$
  Interfaces: $I_{nS-OL} := I_{3S-O1} = O_{1_3S}$
- Saídas:
  $\omega(p_{1_3S})$: sinalizar pré-cond. p/ desligar bomba não satisfeitas
  $\omega(p_{2_3S}$: sinalizar pré-cond. p/ desligar bomba satisfeitas

Métodos:
t_{p1_3S}: pré-condições satisfeitas
t_{2_3S}: pré-condições não satisfeitas

Marcação inicial da `classe` C_{3S}
$M_{0_nS} := M_{0_3S} = [1,0]^T$

Figura 3.16 – Modelo da `classe` C_{3S} = pr_con_d_b (pré-condições para desligar bomba).



Classe: C_{4S} = ctr_b

Atributos:

- Variáveis:

  Booleanas: $V_{4S} = \{0, 1\}$

  Interfaces: $I_{nS-OL} := I_{4S-O1} = O_{1_4S}$

- Saídas:

$\omega(p_{1_4S})$: sinalizar modo remoto
$\omega(p_{2_4S})$: ligar bomba
$\omega(p_{3_4S})$: sinalizar bomba ligada
$\omega(p_{4_4S})$: desligar bomba
$\omega(p_{5_4S})$: sinalizar bomba desligada

Métodos:

t_{c1_4S}: pré-condições p/ operar bomba
t_{c2_4S}: condições de bomba ligada
t_{c3_4S}: pré-condições p/ desligar bomba
t_{c4_4S}: condições de bomba desligada
t_{c5_4S}: pré-condições p/ ligar bomba
t_{c6_4S}: modo de operação remota selecionado

Objeto:

$O_{L_nS} := O_{1_4S}$ = **CTR_B**

Marcação inicial da classe C_{4S}

$M_{0_nS} := M_{0_4S} = [1, 0, 0, 0, 0]^T$

Figura 3.17 – Modelo da classe C_{4S} = ctr_b (controle liga/desliga bomba).

- Na Figura 3.17, o modelo da Classe C_{4S} é interpretado da seguinte forma:

- A classe e os objetos instanciados desta classe possuem cinco *estados* alcançáveis possíveis: $M_{0_4S} = [1, 0, 0, 0, 0]^T$: em modo remoto (*estado* inicial); $M_{1_4S} = [0, 1, 0, 0, 0]^T$: bomba em partida; $M_{2_4S} = [0, 0, 1, 0, 0]^T$: bomba ligada; $M_{3_4S} = [0, 0, 0, 1, 0]^T$: bomba em desligamento; $M_{4_4S} = [0, 0, 0, 0, 1]^T$: bomba desligada.
- As funções de saída $\omega(P_{1_4S})$, $\omega(P_{2_4S})$, $\omega(P_{3_4S})$, $\omega(P_{4_4S})$ e $\omega(P_{5_4S})$ representam ações de comando que fazem parte do conjunto de atributos do objeto instanciado neste *estado*.
- A classe está representada com um objeto (O_{1_4S} = **CTR_B**) instanciado no *estado* em modo remoto (*estado* inicial).
- As *transições* t_{c1_4S}, t_{c2_4S}, t_{c3_4S}, t_{c4_4S}, t_{c5_4S} e t_{c6_4S} indicam que consomem (ou chamam) um ou mais métodos fornecidos por outras classes.

- Da mesma forma que nas `classes` apresentadas anteriormente, as *transições* t_{cy_4S} podem disparar somente se suas respectivas funções de habilitação $\varphi(t_{cy_4S})$ são satisfeitas e também tiverem uma *marca* em seu pré-*lugar* e nenhuma marca em seu pós-*lugar*.
- A variável de interface I_{4S-O1} armazena o nome do `objeto` O_{1_4S} que é igual à **`CTR_B`** que pode chamar `objetos` de outras `classes`.

3.5.2.3 Grafo de Marcação dos *Estados* Alcançáveis das `Classes`

A seguir na Figura 3.18, apresentam-se os *estados* alcançáveis para os `objetos` das `classes` do Sistema de Drenagem do Óleo de Infiltração, representado na Figura 3.10.

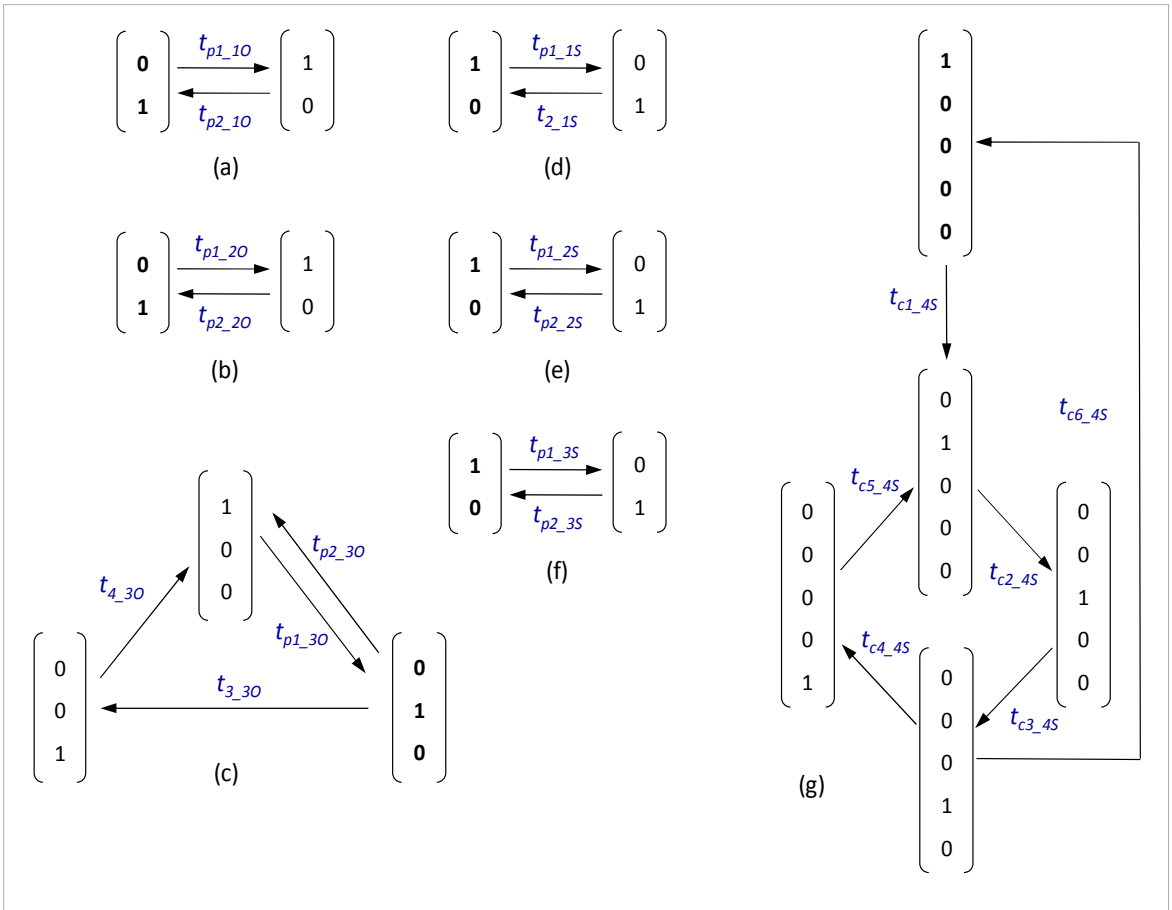


Figura 3.18 - Grafo das Marcações Alcançáveis para os `objetos` das `classes` do Sistema de Drenagem do Óleo de Infiltração do SRVH representado na Fig. 3.10. Onde: (a) `classe` C_{1O} = `ch_l/r` (chave local/remoto. (b) `classe` C_{2O} = `ch_n` (chave de nível). (c) `classe` C_{3O} = `mb` (conjunto motor-bomba). (d) `classe` C_{1S} = `pr_con_op_b` (pré-condições para operar bomba). (e) `classe` C_{2S} = `pr_con_l_b` (pré-condições para ligar bomba). (f) `classe` C_{3S} = `pr_con_d_b` (pré-condições para desligar bomba). (g) `classe` C_{4S} = `ctr_b` (controle liga/desliga bomba).

Observa-se que a *marcação* inicial em todos os Grafos de Marcações Alcançáveis na Figura 3.18 está representada em negrito.

3.5.3 Construção dos Modelos Integrados

Uma vez que as `classes` e os *estados* alcançáveis de seus `objetos` estão definidos, procede-se com a identificação da fusão das *transições* das `classes`, o desdobramento das `classes` (sub-redes), a modelagem dos `objetos` e suas `interfaces` e, finalmente, a construção dos modelos integrados.

3.5.3.1 Identificação da Fusão das *Transições* das `Classes` - Passo 1.

Conforme *Passo 1* definido na Seção 3.4.9.1, identificam-se as fusões das *transições* das `classes` do Sistema de Controle.

`Classe`: C_{1S} = `pr_con_op_b` (pré-condições para operar a bomba):

t_{p1_1S}: produz o `método` pré-condições satisfeitas p/ operação da bomba, disponível em sua variável de interface I_{1S-O1}.

`Classe`: C_{2S} = `pr_con_l_b` (pré-condições para ligar bomba):

t_{p1_2S}: produz o `método` pré-condições satisfeitas p/ ligar a bomba, disponível em sua variável de interface I_{2S-O1}.

`Classe`: C_{3S} = `pr_con_d_b` (pré-condições para desligar bomba):

t_{p1_3S}: produz o `método` pré-condições satisfeitas p/ desligar a bomba, disponível em sua variável de interface I_{3S-O1}.

`Classe`: C_{4S} = `ctr_b` (controle liga/desliga bomba):

$t_{c1_4S} \rightarrow t_{p1_1S}.I_{1S-O1}$: chama o `método` pré-condições satisfeitas para operar bomba, do `objeto` O_{1_1S} = **`PR_CON_OP_B`.**

$t_{c2_4S} \rightarrow t_{p2_3O}.I_{3O-O1}$: chama o `método` condições de bomba ligada, do `objeto` O_{1_3O} = **`MB`.**

$t_{c3_4S} \rightarrow t_{p1_3S}.I_{3S-O1} \wedge : \rightarrow t_{p2_2O}.I_{2O-O2}$ chama os `métodos` pré-condições satisfeitas para desligar bomba, do `objeto` O_{1_3S} = **PR_CON_D_B** e nível alto ativado, do objeto O_{2_2O} = **CH_N-A.**

$t_{c4_4S} \rightarrow t_{p1_3O}.I_{3O-O1}$: chama o `método` condições de bomba desligada, do `objeto` O_{1_3O} = **MB.**

$t_{c5_4S} \rightarrow t_{p1_2S}.I_{2S-O1} \wedge : \rightarrow t_{p2_2O}.I_{2O-O1}$ chama os `métodos` pré-condições satisfeitas para ligar bomba, do `objeto` O_{1_2S} = **PR_CON_L_B** e nível baixo ativado, do objeto O_{1_2O} = **CH_N-B.**

$t_{c6_4S} \rightarrow t_{p1_1O}.I_{1O-O1}$: chama o `método` chave na posição remota, do `objeto` O_{1_1O} = **CH_L-R.**

3.5.3.2 Desdobramento das `Classes` em `Objetos` e Renomeação para o Nível de Objetos - Passo 2.

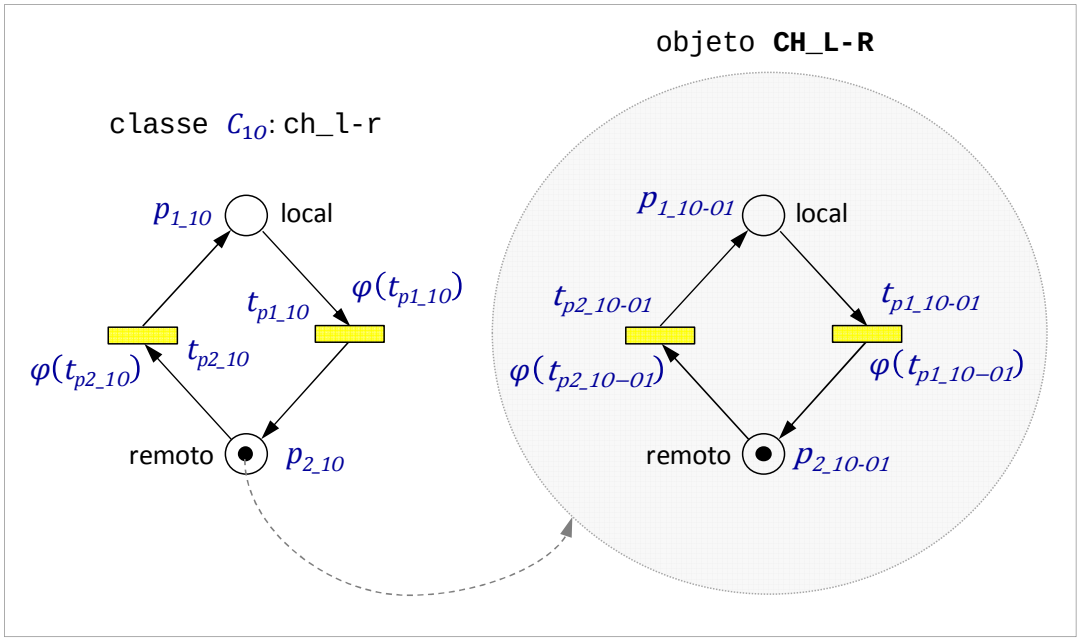


Figura 3.19 – `Classe` C_{1O}: `ch_l-r` e `objeto` O_{1_1O} = **CH_L-R** (chave local/remoto).

Onde:

$\varphi(t_{p1_1O-01}) = i_1$ (chave em operação remota)

$\varphi(t_{p2_1O-01}) = \neg i_1$ (chave em operação local)

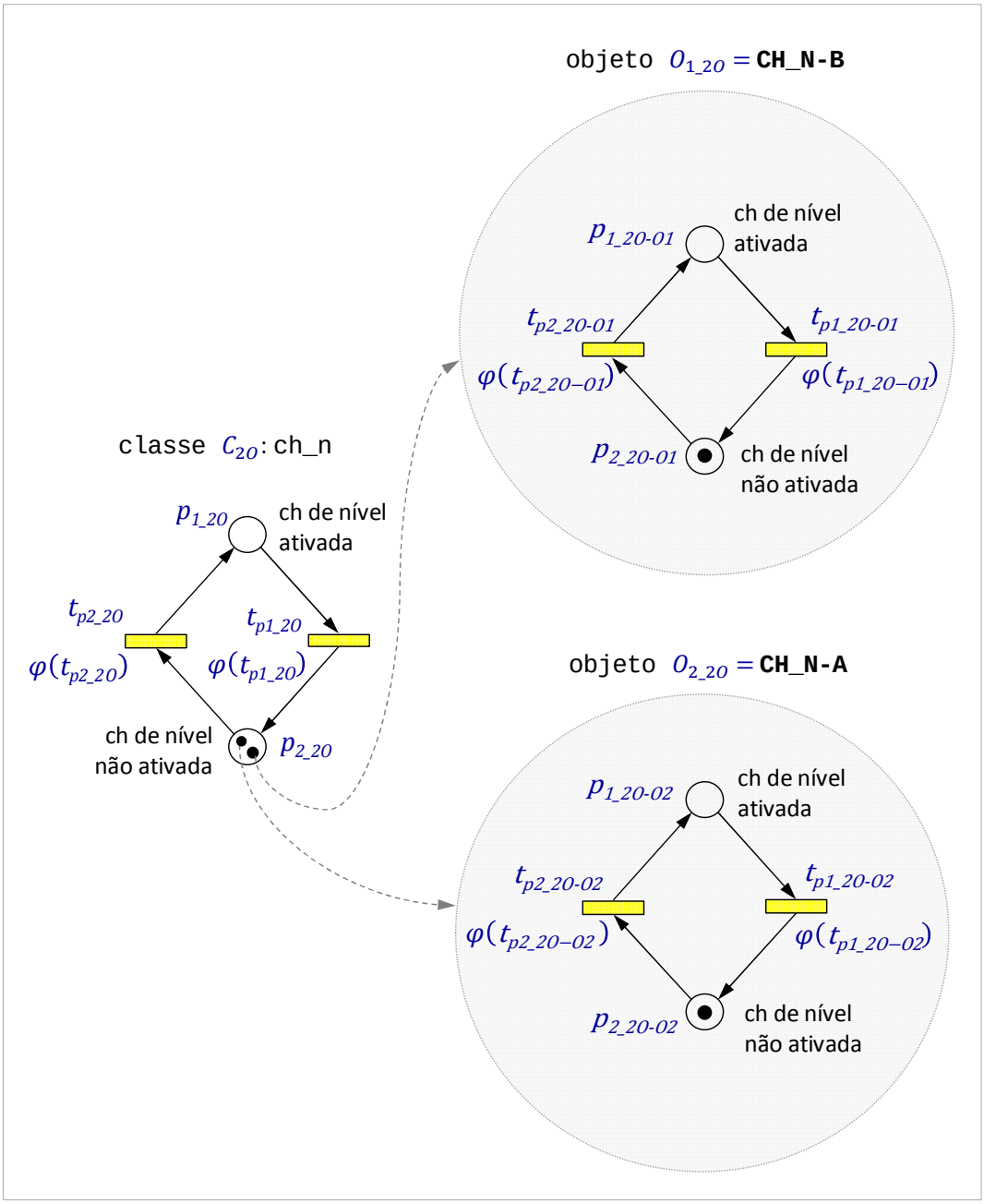


Figura 3.20 – Classe C_{20}: ch_n com a instanciação dos objetos O_{1_20} = **CH_N-B** e O_{2_20} = **CH_N-A**.

Onde:

$\varphi(t_{p1_20-01}) = \neg i_2$ (chave de nível baixo - não ativada)

$\varphi(t_{p2_20-01}) = i_2$ (chave de nível baixo - ativada)

$\varphi(t_{p1_20-02}) = \neg i_3$ (chave de nível alto - não ativada)

$\varphi(t_{p2_20-02}) = i_3$ (chave de nível alto - ativada)

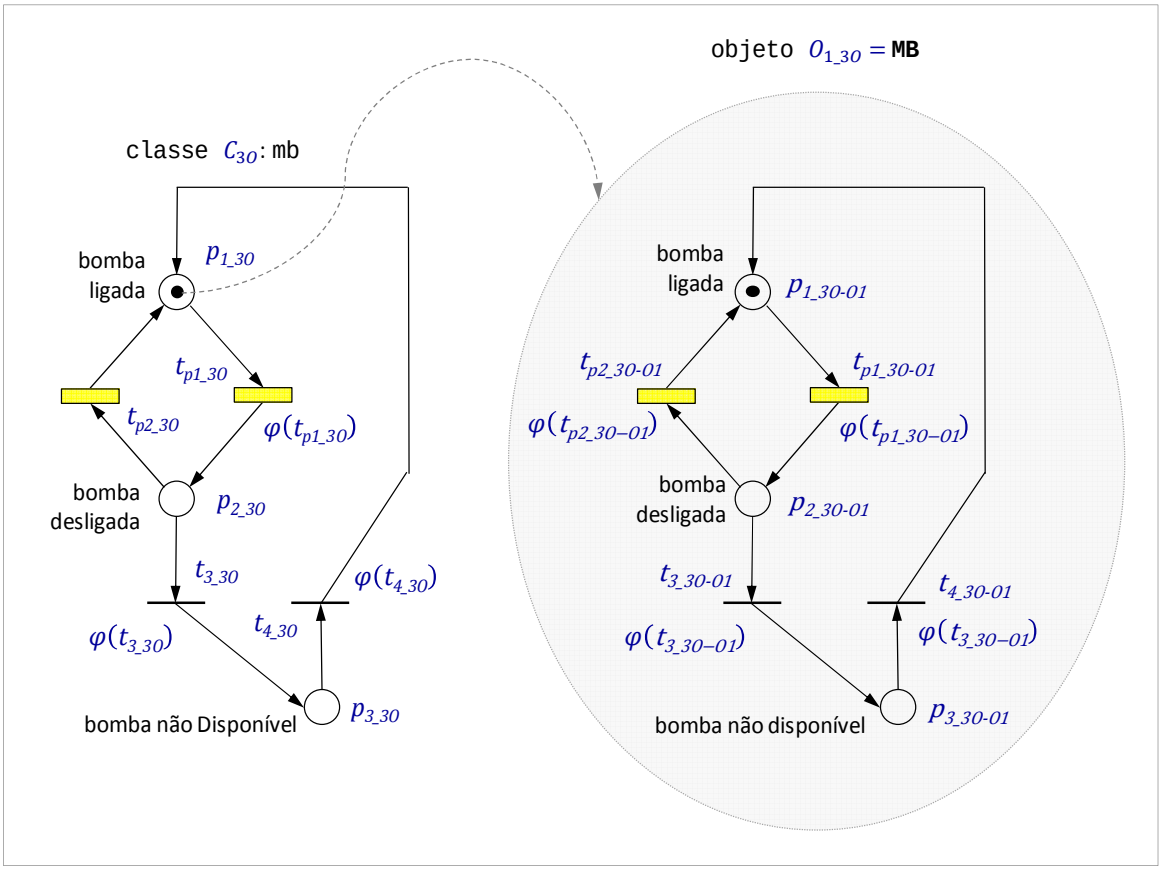


Figura 3.21 – `Classe` C_{30}: `mb` e `objeto` instanciado O_{1_30} = **MB** – bomba.

Onde:

$\varphi(t_{p1_30-01}) = i_5$ (bomba desligada)

$\varphi(t_{p2_30-01}) = i_4$ (bomba ligada)

$\varphi(t_{3_30-01}) = i_6$ (bomba não disponível)

$\varphi(t_{4_30-01}) = i_4$ (bomba ligada)

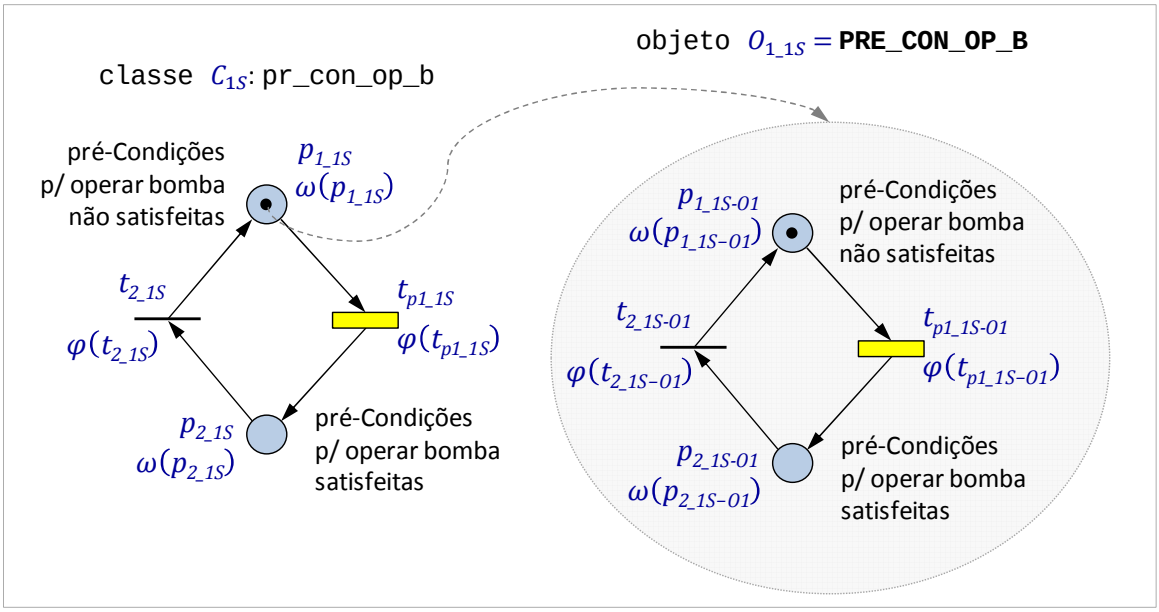


Figura 3.22 – `Classe` C_{1S} = `pr_con_op_b` com a instanciação do `objeto` O_{1_1S} = **PR_CON_OP_B** - pré-condições p/ operar bomba.

Onde:

$\varphi(t_{p1_1S-O1}) = vi_1 \wedge \neg i_7$ (pré-condições para operar bomba - satisfeitas)

$\varphi(t_{2_1S-O1}) = \neg vi_1 \vee i_7$ (pré-condições para operar bomba - não satisfeitas)

$\omega(p_{i_1S-O1}) = (vo_1) \Longrightarrow$

$\omega(p_{1_1S-O1}) = (0)$

$\omega(p_{2_1S-O1}) = (1)$

i_7: Botão de emergência acionado

vi_1: Sistema pronto para operar (via rede)

vo_1: Sinalizar pré-condições para operar bomba - satisfeitas

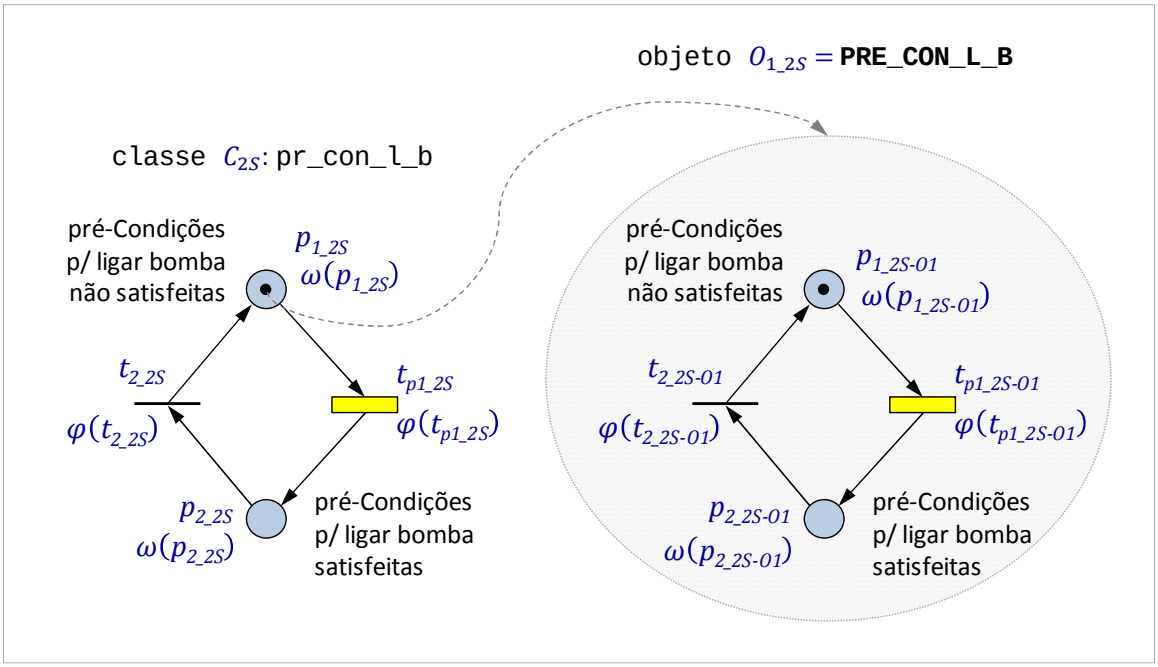


Figura 3.23 – `Classe` C_{2S} = `pr_con_op_b` com a instanciação do `objeto` O_{1_2S} = **PR_CON_L_B** - pré-condições p/ ligar bomba.

Onde:

$\varphi(t_{p1_2S-O1}) = i_1 \wedge \neg i_6$ (pré-condições para ligar bomba - satisfeitas)

$\varphi(t_{2_2S-O1}) = \neg i_1 \vee i_6$ (pré-condições para ligar bomba - não satisfeitas)

$\omega(p_{i_2S-O1}) = (vo_2) \Longrightarrow$

$\omega(p_{1_2S-O1}) = (0)$

$\omega(p_{2_2S-O1}) = (1)$

i_1: Chave em operação remota

i_6: Bomba não disponível

vo_2: Sinalizar pré-condições para ligar bomba - satisfeitas

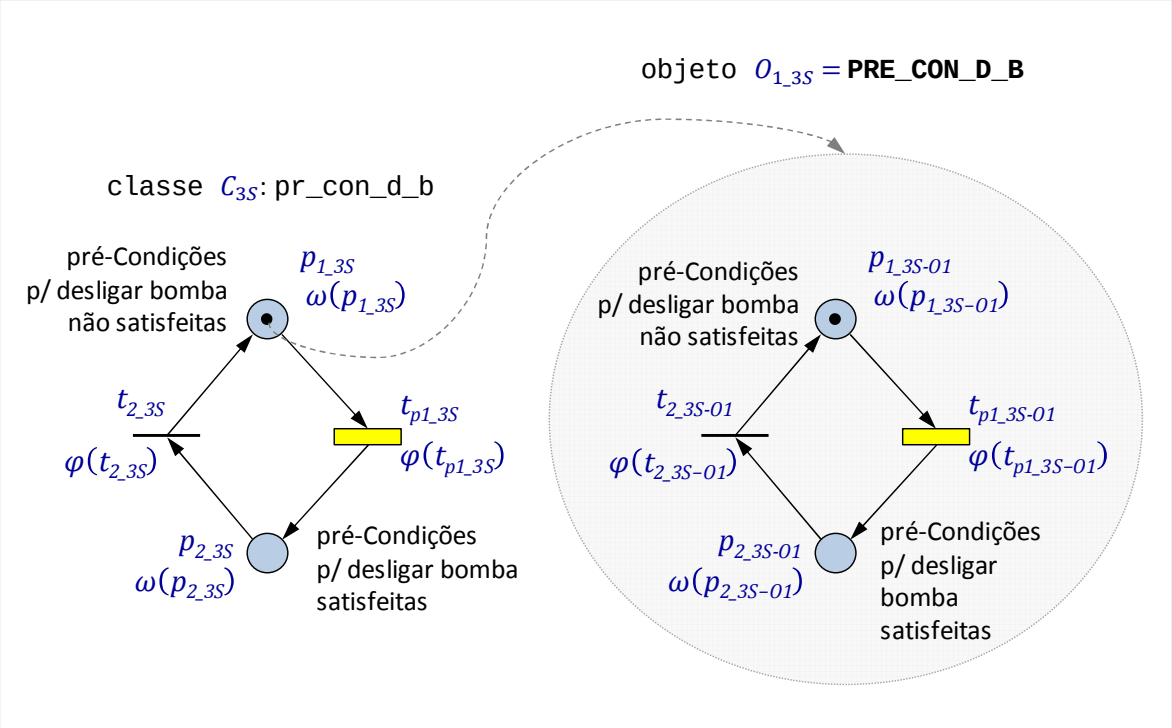


Figura 3.24 – Classe C_{3S} = pr_con_d_b com a instanciação do objeto O_{1_3S} = **PR_CON_D_B** - pré-condições p/ desligar bomba.

Onde:

$\varphi(t_{p1_3S-O1}) = i_1$ (pré-condições para desligar bomba - satisfeitas)

$\varphi(t_{2_3S-O1}) = \neg i_1$ (pré-condições para desligar bomba - não satisfeitas)

$\omega(p_{i_3S-O1}) = (vo_3) \Longrightarrow$

$\omega(p_{1_3S-O1}) = (0)$

$\omega(p_{2_3S-O1}) = (1)$

i_1: Chave em operação remota

vo_3: Sinalizar pré-condições para desligar bomba - satisfeitas

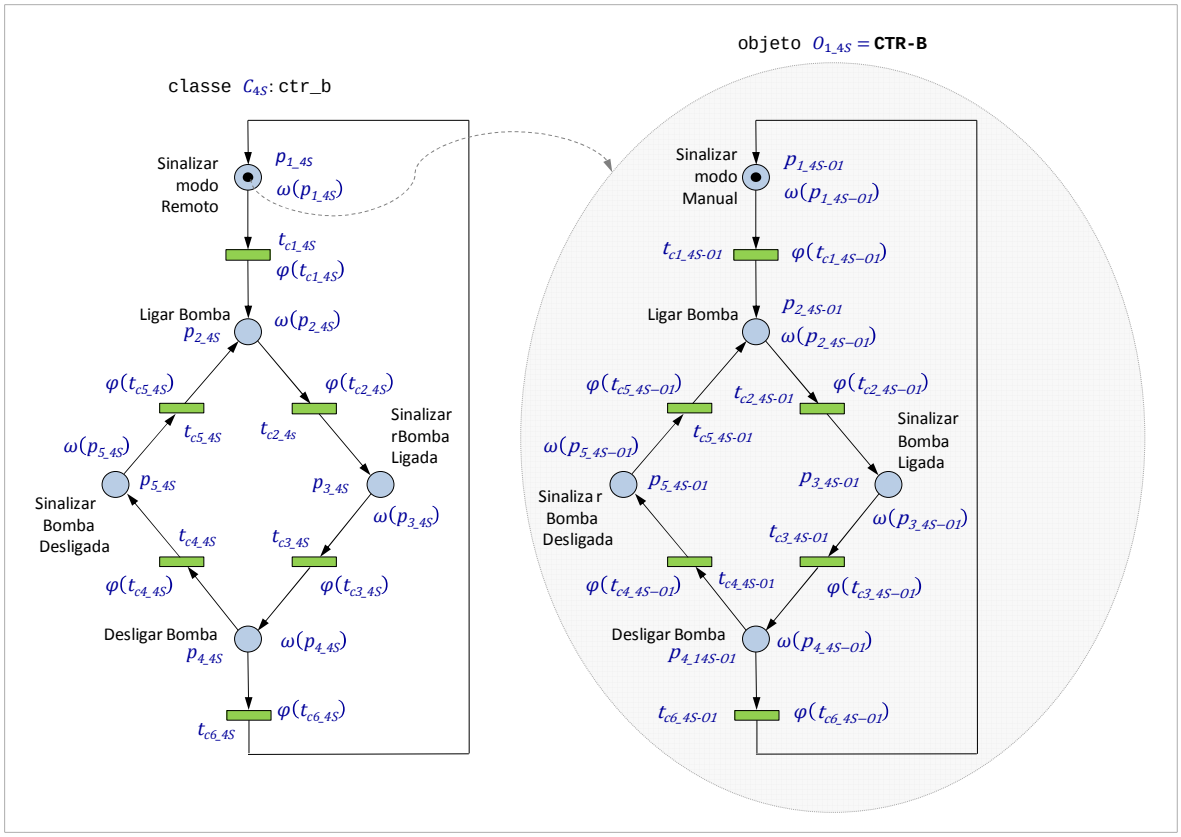


Figura 3.25 – `Classe` C_{4S} = `ctr_b` com a instanciação do `objeto` O_{1_4S} = **CTR_B** - pré-condições p/ desligar bomba.

Onde:

$\varphi(t_{c1_4S-O1}) = i_1 \wedge \neg i_6$ (pré-condições p/ operar bomba)

$\varphi(t_{c2_4S-O1}) = i_1 \wedge \neg i_6 \wedge vi_1 \wedge \neg i_7$ (condições de bomba ligada)

$\varphi(t_{c3_4S-O1}) = \neg i_6 \wedge vi_1 \wedge \neg i_7$ (pré-condições p/ desligar bomba)

$\varphi(t_{c4_4S-O1}) = i_1 \wedge \neg i_6 \wedge vi_1 \wedge \neg i_7$ (condições de bomba desligada)

$\varphi(t_{c5_4S-O1}) = vi_1 \wedge \neg i_7$ (pré-condições p/ ligar bomba)

$\varphi(t_{c6_4S-O1}) = \neg i_6 \wedge vi_1 \wedge \neg i_7$ (modo de operação remota selecionado)

$\omega(p_{i_4S-O1}) = (o_1, o_2, o_3, o_4) \Longrightarrow$

$\omega(p_{1_4S}) = (0,\ 0,\ 0,\ 0)$: início

$\omega(p_{2_4S}) = (1,\ 0,\ 0,\ 0)$: ligar bomba

$\omega(p_{3_4S}) = (0,\ 1,\ 0,\ 0)$: sinalizar bomba ligada

$\omega(p_{4_4S}) = (0,\ 0,\ 1,\ 0)$: desligar bomba

$\omega(p_{5_4S}) = (0,\ 0,\ 0,\ 1)$: sinalizar bomba desligada

i_1: Chave em operação remota

i_6: Bomba não disponível

i_7: Botão de emergência acionado

vi_1: Sistema pronto para operar (via rede)

o_1: Ligar bomba

o_2: Sinalizar bomba ligada

o_3: Desligar bomba

o_4: Sinalizar bomba desligada

vo_1: Sinalizar pré-condições para operar bomba - satisfeitas

vo_2: Sinalizar pré-condições para ligar bomba - satisfeitas

vo_3: Sinalizar pré-condições para desligar bomba – satisfeitas

3.5.3.3 Fusão das *Transições* dos `Objetos` do Objeto de Controle - Passo 3.

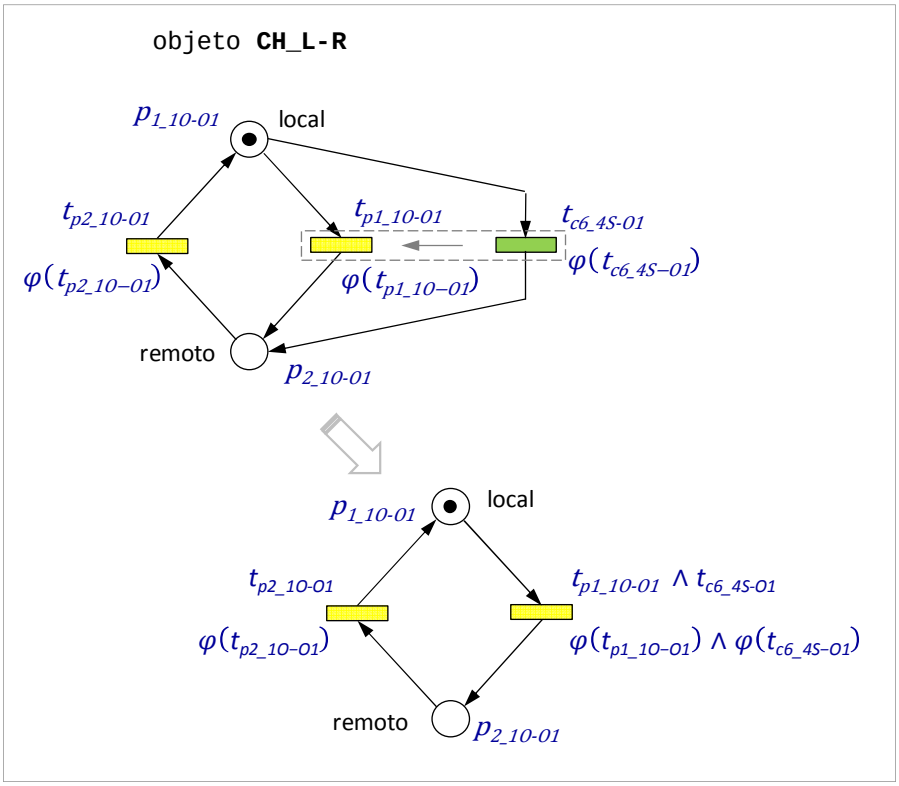


Figura 3.26 – Fusão e renomeação das *transições* do `objeto` O_{1_10} = **`CH_L-R`** (chave local/remoto).

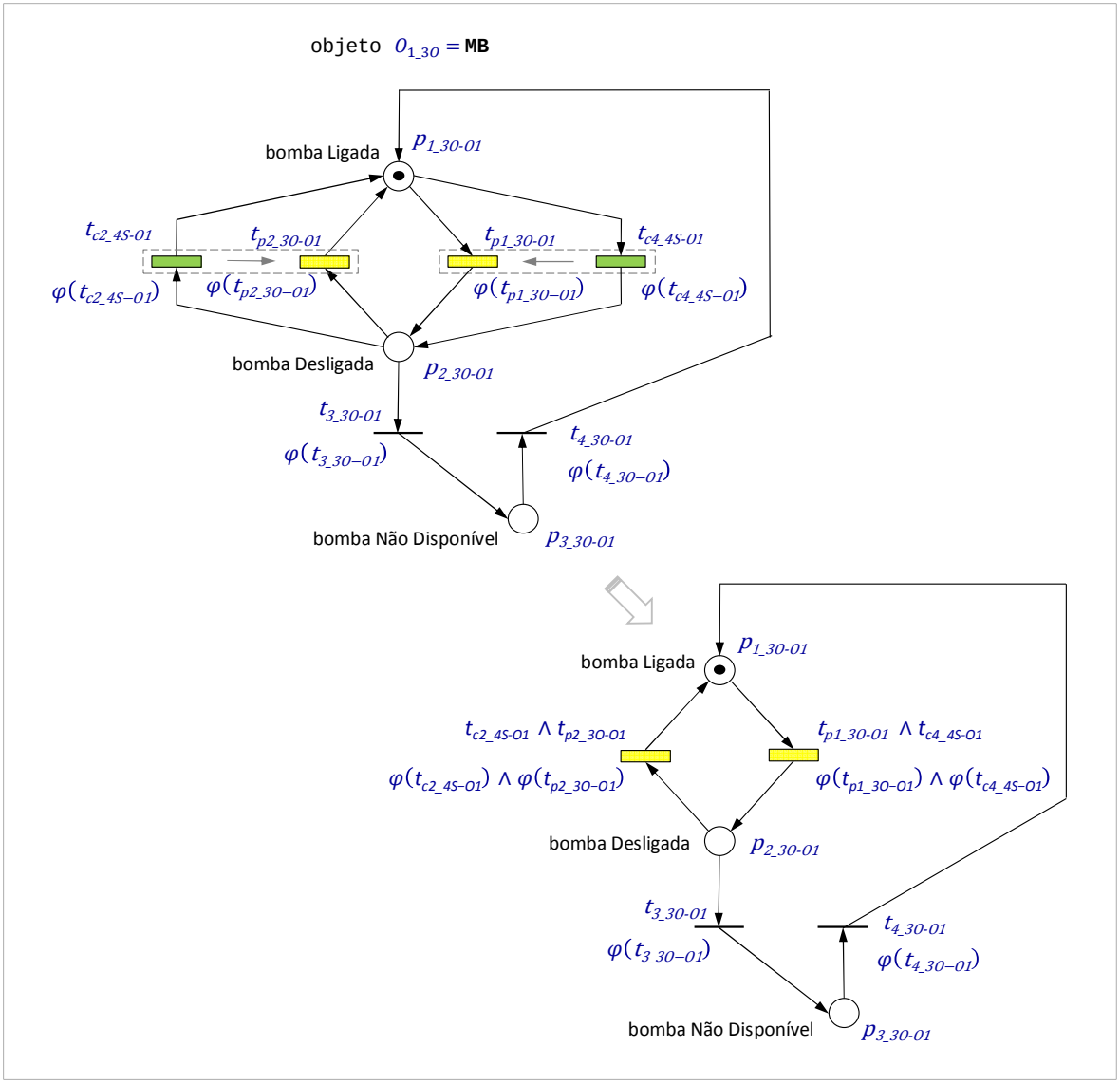


Figura 3.27 – Fusão e renomeação das *transições* do `objeto` O_{1_3O} = **MB** – bomba.

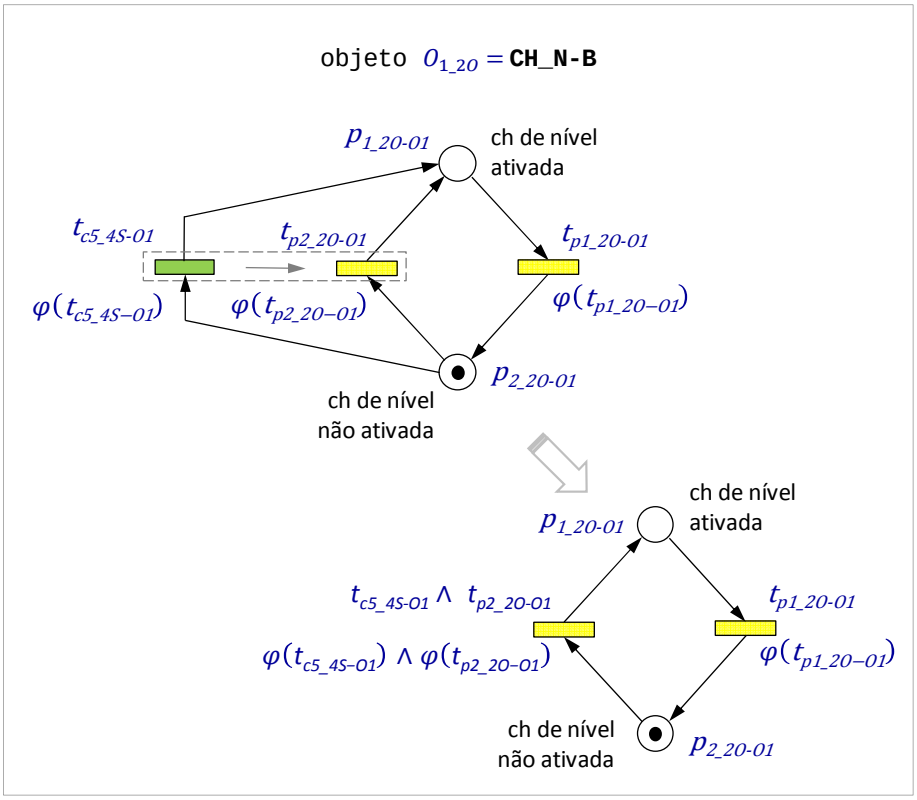


Figura 3.28 – Fusão e renomeação das *transições* do objeto O_{1_20} = **CH_N-B** - chave de nível baixo.

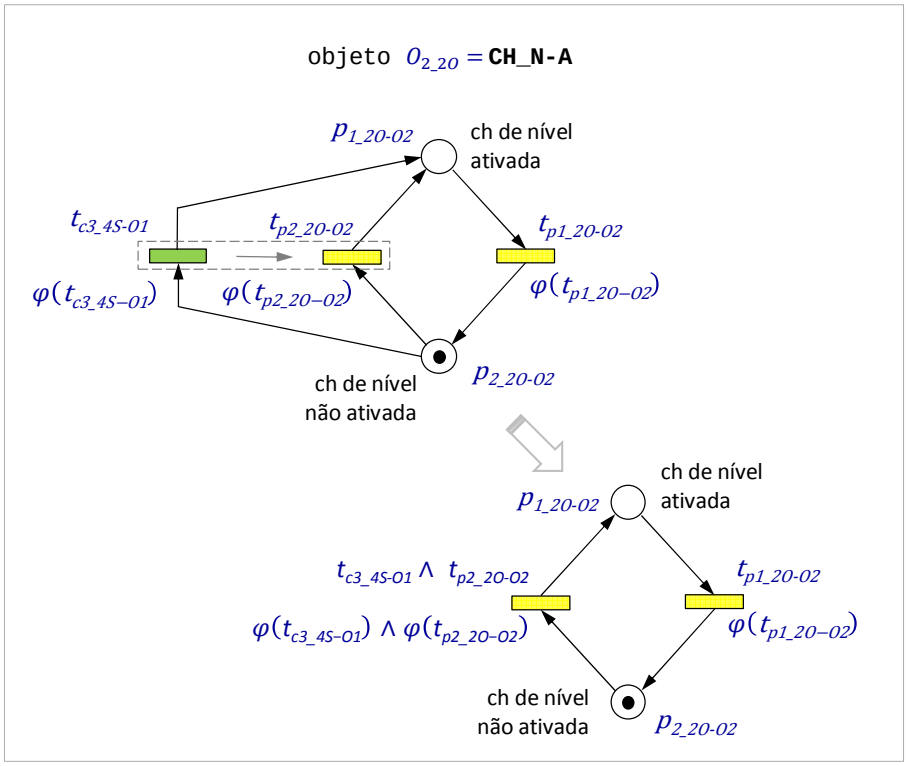


Figura 3.29 – Fusão e renomeação das *transições* do objeto O_{2_20} = **CH_N-A** - chave de nível alto.

3.5.3.4 Fusão das *transições* dos `Objetos` do Sistema de Controle - Passo 4.

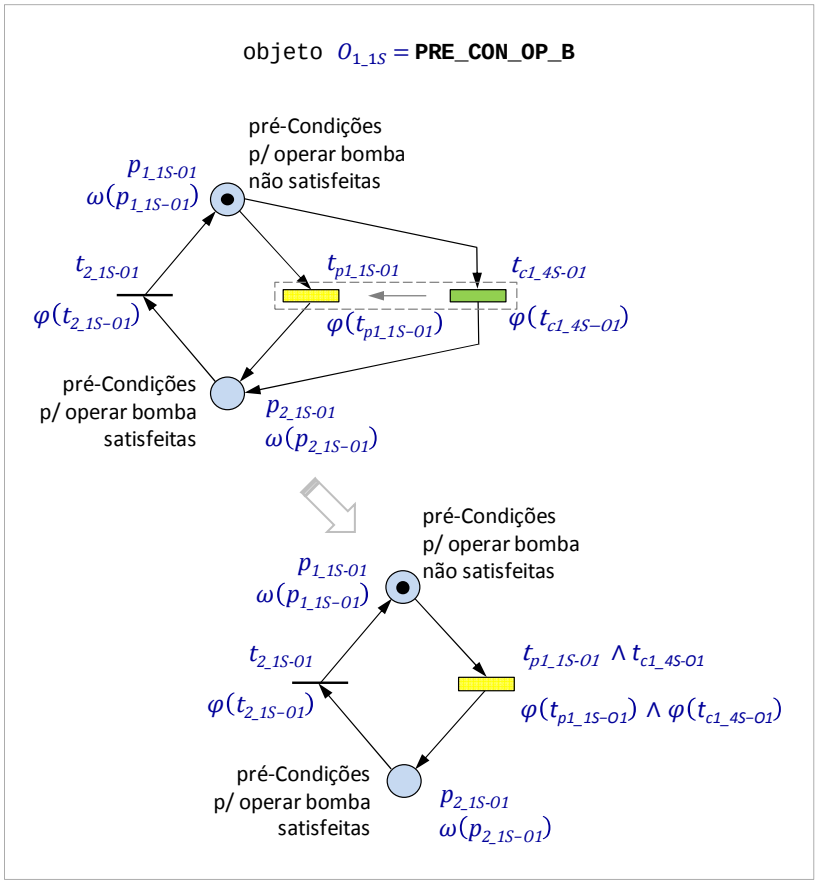


Figura 3.30 – Fusão e renomeação das *transições* do `objeto` O_{1_1S} = **PR_CON_OP_B**.

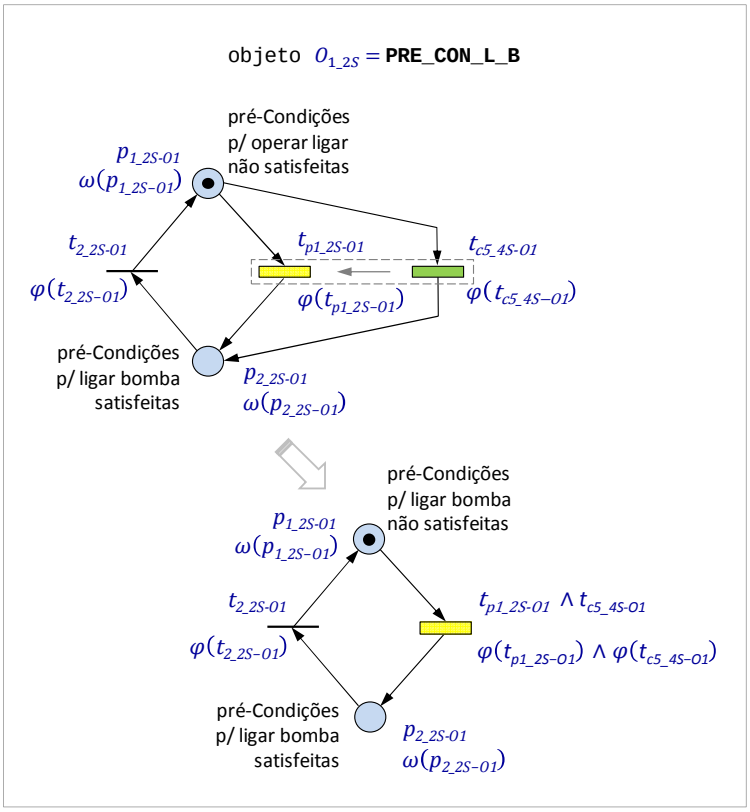


Figura 3.31 – Fusão e renomeação das *transições* do `objeto` O_{1_2S} = **PR_CON_L_B**.

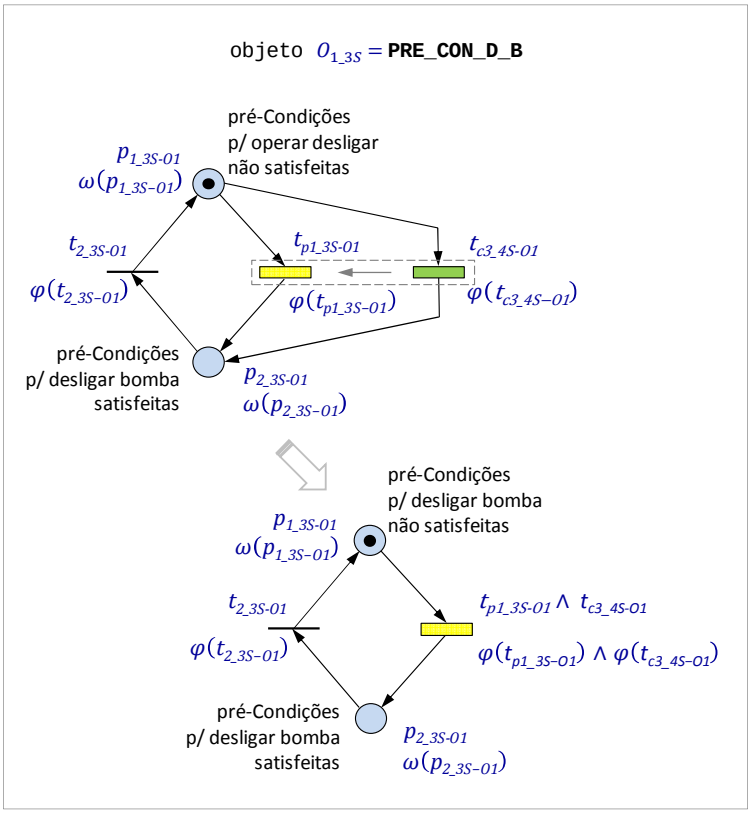


Figura 3.32 – Fusão e renomeação das *transições* do `objeto` O_{1_3S} = **PR_CON_D_B**.

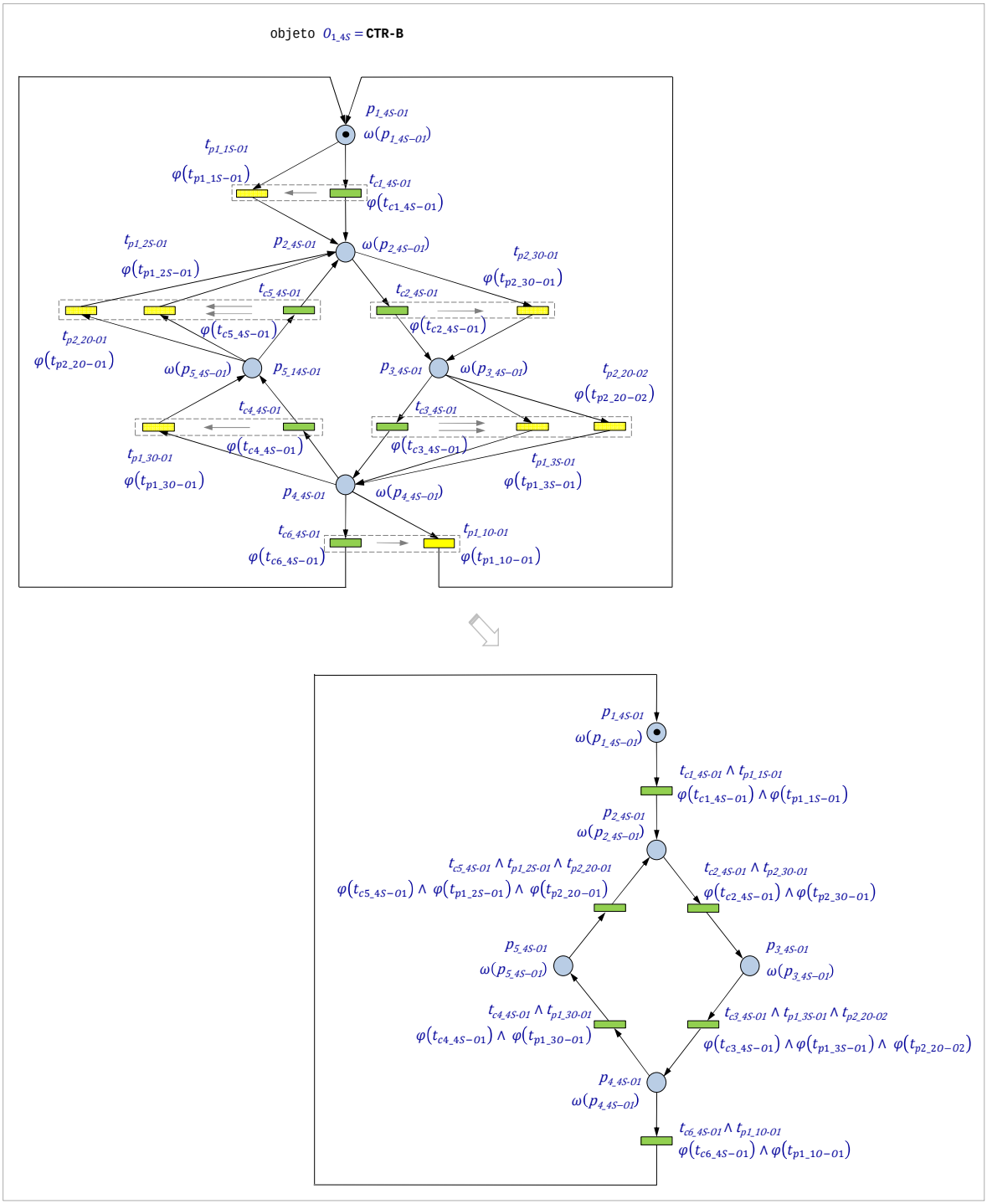


Figura 3.33 – Fusão e renomeação das *transições* do `objeto` O_{1_4S} = **CTR_B**.

3.5.3.5 Modelo Integrado dos `Objetos` do Sistema de Controle – Passo 5.

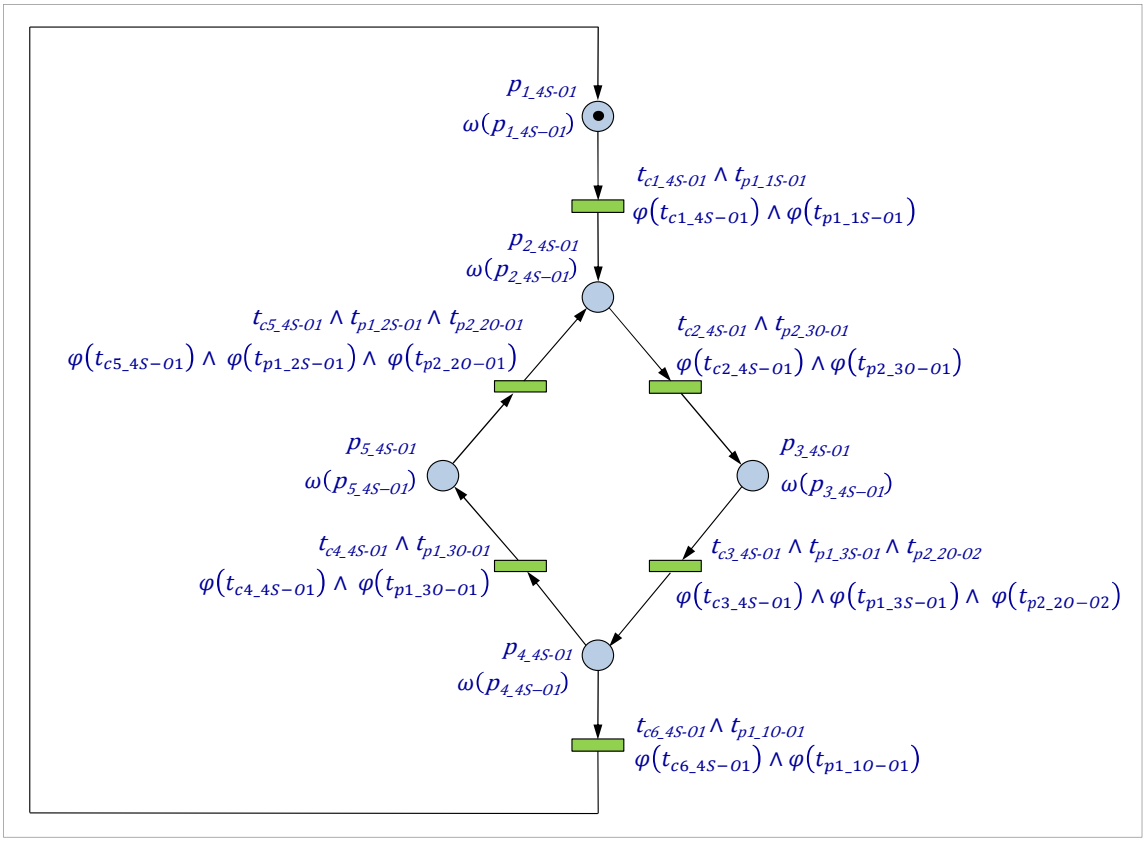


Figura 3.34 – Modelo integrado do `objeto` O_{1_4S} = **`CTR_B`** (controle liga/desliga bomba)

Onde:

$\varphi(t_{c1_4S-01}) \wedge \varphi(t_{p1_1S-01})$ =

$(i_1 \wedge \neg i_6) \wedge (vi_1 \wedge \neg i_7)$: pré-condições p/ operar bomba

$\varphi(t_{c2_4S-01}) \wedge \varphi(t_{p2_3O-01})$ =

$(i_1 \wedge \neg i_6 \wedge vi_1 \wedge \neg i_7) \wedge (i_4)$: condições de bomba ligada

$\varphi(t_{c3_4S-01}) \wedge \varphi(t_{p1_3S-01}) \wedge \varphi(t_{p2_2O-02})$ =

$(\neg i_6 \wedge vi_1 \wedge \neg i_7) \wedge (i_1) \wedge (i_3)$: pré-condições p/ desligar bomba

$\varphi(t_{c4_4S-01}) \wedge \varphi(t_{p1_3O-01})$ =

$(i_1 \wedge \neg i_6 \wedge vi_1 \wedge \neg i_7) \wedge (i_5)$: condições de bomba desligada

$\varphi(t_{c5_4S-01}) \wedge \varphi(t_{p1_2S-01}) \wedge \varphi(t_{p2_2O-01})$ =

$(vi_1 \wedge \neg i_7) \wedge (i_1 \wedge \neg i_6) \wedge (i_2)$: pré-condições p/ ligar bomba

$\varphi(t_{c6_4S-01}) \wedge \varphi(t_{p1_1O-01})$ =

$(\neg i_6 \wedge vi_1 \wedge \neg i_7) \wedge (i_1)$: modo de operação remota selecionado

$\omega(p_{i_4S-O1}) = (o_1, o_2, o_3, o_4) \Longrightarrow$

$\omega(p_{1_4S}) = (0,\ 0,\ 0,\ 0)$: início

$\omega(p_{2_4S}) = (1,\ 0,\ 0,\ 0)$: ligar bomba

$\omega(p_{3_4S}) = (0,\ 1,\ 0,\ 0)$: sinalizar bomba ligada

$\omega(p_{4_4S}) = (0,\ 0,\ 1,\ 0)$: desligar bomba

$\omega(p_{5_4S}) = (0,\ 0,\ 0,\ 1)$: sinalizar bomba desligada

i_1: Chave em operação remota

i_2: Chave de nível baixo - LSL

i_3: Chave de nível alto – LSH

i_4: Bomba ligada

i_5: Bomba desligada

i_6: Bomba não disponível

i_7: Botão de emergência acionado

vi_1: Sistema pronto para operar (via rede)

o_1: Ligar bomba

o_2: Sinalizar bomba ligada

o_3: Desligar bomba

o_4: Sinalizar bomba desligada

vo_1: Sinalizar pré-condições para operar bomba - satisfeitas

vo_2: Sinalizar pré-condições para ligar bomba - satisfeitas

vo_3: Sinalizar pré-condições para desligar bomba – satisfeitas

3.6 Considerações sobre o Capítulo

Apresentou-se um formalismo para a modelagem de sistemas, sob um foco de SDED, com base na RPIS e aplicação dos conceitos de orientação a `objetos`, onde um sistema pode ser representado como um agrupamento de `objetos` que se relacionam por meio de suas interfaces, de acordo com as características definidas nas `classes` que pertencem. Cada `objeto` pode ser entendido como a abstração de um equipamento, instrumento ou dispositivo real pertencente ao Objeto de Controle ou um programa ou parte de um programa que pertence ao Sistema de Controle, de acordo com seus `atributos`,

`operações` e `métodos` definidos por suas `classes` ou sub-redes. As sub-redes $RPIS_OC_{mO}$ e $RPIS_SC_{nS}$ se diferenciam principalmente no que diz respeito aos *lugares*, pois na $RPIS_OC_{mO}$ não está definido um mapeamento associando os *lugares* a saídas digitais, as quais pode alterar o *estado* de algum outro `objeto`; o *lugar*, neste caso, quando *marcado*, somente informa o *estado* atual do próprio `objeto`.

Distintamente, em um `objeto` do Sistema de Controle, $RPIS_SC_{nS}$, os *lugares* estão associados a um mapeamento de saídas, $\omega(p_{i_nS})$. Este mapeamento caracteriza uma ação de comando, o qual é um `atributo` do `objeto`, que pode também alterar o *estado* de outro `objeto`.

Um dos objetivos é que cada `classe` ou sub-rede possa ser construída separadamente, e depois, pelo procedimento de desdobramento possa se construir o modelo do sistema integrado por meio dos `objetos`, a partir de uma determinada *marcação*. Isto é válido, se os `objetos` e sua quantidade são conhecidos e instanciados quando da construção das sub-redes, como se considera neste trabalho. Conforme exemplificado nas figuras anteriores, os modelos integrados do sistema, para fins de análise, podem ser tratados como RPIS (os outros dados do `objeto` podem ser considerados que estão encapsulados), a qual é considerada uma RP Segura e Não Autônoma[53], onde a fusão de *transições* é interpretada como uma única *transição* que deve satisfazer as condições de *habilitação* das *transições* envolvidas, oriundas de diferentes `objetos`, por meio de uma conjunção lógica das funções de *disparo*, criando-se desta forma, uma sincronização entre os `objetos` envolvidos.

Alguns autores denominam este tipo de sincronização como sincronização tipo Recíproca ou *"Rendez-vous"* [93], conforme ilustrado na Figura 3.35, onde se considera que a fusão das *transições* indicadas se comporta como uma única *transição*. Ressalta-se que ambas as RP evoluem se e somente se os seus *pré-lugares* estão *marcados*, seus *pós-lugares* não estão *marcados* e as *transições* estão *habilitadas* de acordo com suas funções de *disparo*.

[53] Rede de Petri Não-Autônoma: descreve ou modela o funcionamento de um sistema, cuja evolução é condicionada por *eventos* externos (interage com o meio ambiente) e/ou pelo tempo. Uma RP Não-Autônoma é sincronizada e/ou temporizada. Diferente de uma Rede de Petri Autônoma, a qual descreve o funcionamento de um sistema em evolução de uma forma autônoma, ou seja, cujos instantes de *disparo* são desconhecidos ou não são indicados [93].

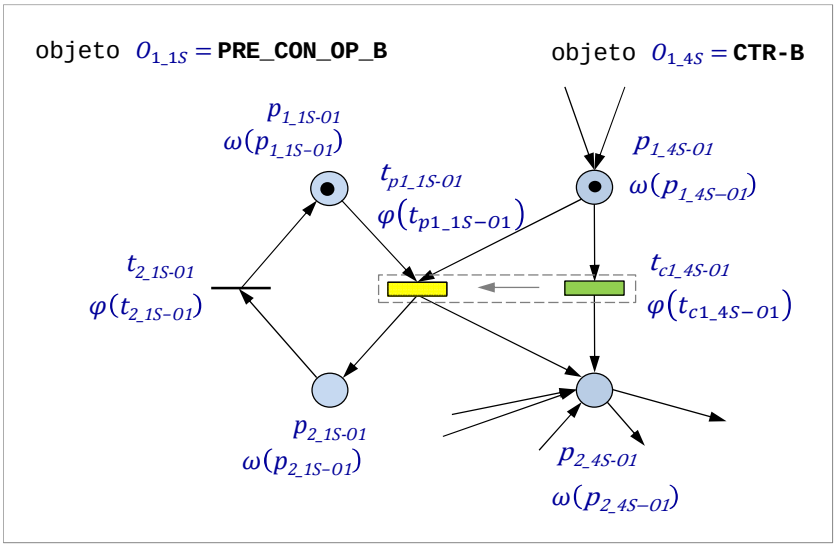


Figura 3.35 – Exemplo de Sincronização tipo Recíproca ou *"Rendez-vous"* (fusão das *transições* t_{p1_1S-01} e t_{c1_4S-01}, a qual é parte do `objeto` O_{1_4S} = **CTR_B** apresentado na Figura 3.33).

A fusão de *transições* por meio da lógica *and* das funções de *habilitação* possibilita alternativas para composição e restrições (quando necessárias) de lógicas no mecanismo de evolução de *estados* do modelo em $RPIS_OO$. Pode-se, por exemplo, utilizar sinais de entrada vindos de níveis hierárquicos de controle superiores ou inferiores propiciando, desta forma, condições de intertravamentos e sincronismos entre os mesmos.

Destaca-se também que as entradas e saídas digitais são as mesmas utilizadas no programa do CLP, independentemente da linguagem de implantação, podendo desta forma ser aproveitadas.

Algumas vantagens do formalismo proposto são a construção modular das `classes` e `objetos`, a reutilização dos modelos e a sua análise com técnicas já desenvolvidas na área, com algumas adaptações. Destaca-se ainda, que a mesma estrutura em `classes` e `objetos`, e os mesmos sinais digitais de entrada e saída utilizados na modelagem, podem ser utilizados na elaboração do programa do CLP, em CFC ou em DIPMCS, ou seja, em se tratando de um SSOO, grande parte do trabalho realizado nas Fases 1, 2 e 3 pode ser aproveitada na Fase 4 – Realização, tendo como referência a Figura 1.1, do Capítulo 1, o que poderá ser visto no Apêndice A – Exemplo de Aplicação. O próximo capítulo apresenta o método proposto para análise dos modelos integrados obtidos, tanto para os `objetos` do Objeto de Controle como para os `objetos` integrados do Sistema de Controle.