Multi-scale process model description by generalized hierarchical CPN models

E. Németh, K. M. Hangos

Research Report SCL-002/2004

Contents

List of Figures

Abstract

In this paper we show how to extend coloured Petri nets (CPNs) with a hierarchy concept. The basic idea behind using hierarchical CPNs is to allow the modeller to construct a large model by using a number of small CPNs which are related to each other in a well-defined way. We discuss seven different ways to relate submodels to each other, and they are illustrated by examples.

The hierarchy constructions can be used for a simple countercurrent heat exchanger.

1 Introduction

This paper purposes a generalized definition of hierarchical coloured Petri nets (CPNs) for multi-scale process modelling purposes. It shows how a set of subnets (called pages) can be related to each other in such a way that they together constitute a single model [1]. The basic idea is to allow the modeller to describe a set of submodels which all contribute to a larger model, in which the submodels interact with each other in a well-defined way. This idea is well-known from other kind of artificial languages. The purpose is to break down the complexity of the large model by dividing it into a number of submodels.

The idea is most easily explained by proposing the notion of a substitution node, which is a place or a transition or a combination of a place and a transition related to a submodel. Usually, the submodel totally replaces the substitution node and the surrounding arcs.

The paper describes 7 different ways to relate submodels to each other. These 7 hierarchy constructions are not at all independent and it is often possible to choose between them, in order to fulfil a certain modelling goal. Thee proposed 7 hierarchy constructions are as follows:

- substitution transition,
- substitution place,
- substitution place-transition pair,
- invocation transition,
- invocation place,
- fusion place,
- fusion transition.

The concepts and notions are illustrated by using a simple process example: a countercurrent heat exchanger model.

2 Coloured Petri nets (CPNs)

In this section, we summarize the basic notations found in the literature about coloured Petri nets and generalized hierarchical coloured Petri nets. For first, the general non-hierarchical coloured case will be considered, while the second part concerns with the various hierarchical Petri nets.

2.1 Coloured Petri nets

Coloured Petri nets (CPNs) [2] belong to the area of discrete event system methodology. A CPN is well known for its capability in modelling discrete event systems.

The structure of a Petri net is a bipartite directed graph describing the structure of a discrete event system, while the dynamics of the system is described by the execution of the Petri net. A Petri net is coloured if the tokens are distinguishable. According to the formal definition of CPNs [3] a coloured Petri net model is a nine-tuple

$$CPN = (\Sigma, P, T, A, N, C, G, E, IN)$$

satisfying the following requirements:

(i) Σ is a finite set of non-empty types, called *colour sets*

(ii) P is a finite set of *places*

(iii) T is a finite set of *transitions*

(iv) A is a finite set of *arcs* such that $P \cap T = P \cap A = T \cap A = \emptyset$

(v) $N : A \to P \times T \cup T \times P$ is a *node function*

(vi) $C : P \to \Sigma$ is a *colour function*

(vii) G is a *guard function*. It is defined from T into expressions such that
$\forall t \in T : [Type(G(t)) = Bool \wedge Type(Var(G(t))) \subseteq \Sigma]$

(viii) E is an *arc function*. It is defined from A into expressions such that
$\forall a \in A : [Type(E(a)) = C(p(s))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$
where $p(a)$ is the place of $N(a)$ and C_{MS} denotes the set of all multi-sets over C

(ix) IN is an *initialization function*. It is defined from P into expressions such that
$\forall p \in P : [Type(IN(p)) = C(p(s))_{MS} \wedge Var(IN(p)) = \emptyset]$

where:
$Type(expr)$ denotes the type of an expression,
$Var(expr)$ denotes the set of variables in an expression,
$C(p)_{MS}$ denotes a multi-set over $C(p)$.

A *binding* of a transition t is a function b defined on $Var(t)$, such that:

(i) $\forall v \in Var(t) : b(v) \in Type(v)$,

(ii) $G(t)_{<b>}$ denotes the evaluation of the guard expression $G(t)$ in the binding b.

A *token element* is a pair (p, c) where $p \in P$ and $c \in C(p)$. A *binding element* is a pair (t, b) where $t \in T$ and $b \in B(t)$. By $B(t)$ denotes the set of all bindings for t. The set of all token elements is denoted by TE while the set of all binding elements is denoted by BE.

A *marking* is a multi-set over TE while a *step* is a non-empty and finite multi-set over BE. The *initial marking* M_0 is the marking which is obtained by evaluating the initialization expressions.

A *transition is enabled* if each of its input places contain the multi-set specified by the input arc inscription (possibly in conjunction with the guard), and the guard evaluates to true. When a transition is enabled it may *occur*, and this means that the tokens are removed from the input places and added to the output places of the occurring transitions. The number and colour of the tokens are determined by the arc expressions, evaluated for the occurring bindings.

A *finite occurrence sequence* is a sequence of markings and steps: $M_1[Y_1\rangle M_2[Y_2\rangle M_3 \dots M_n[Y_n\rangle M_{n+1}$, such that $n \in \mathbb{N}$. A marking M'' is *reachable* from a marking M' if and only if exists a finite occurrence sequence having M' as start marking and M'' as end marking, i.e., if and only if for some $n \in \mathbb{N}$ there exists a sequence of steps $Y_1 Y_2 \dots Y_n$ such that: $M'[Y_1 Y_2 \dots Y_n\rangle M''$. We then also say that M'' is reachable from M' in n step. The set of markings which are reachable from M' is denoted by $[M'\rangle$.

2.2 Generalized hierarchical coloured Petri nets

In this subsection we shall see how non-hierarchical CPNs can be extended to hierarchical nets, i.e., how it is possible to construct a large CPN by combining a number of smaller nets by different several hierarchical constructions.

2.2.1 Introduction to hierarchical coloured Petri nets

CPNs [3, 4] are capable of describing the dynamic behaviour of process systems and handle the hierarchy. The basic idea behind using hierarchical CPNs [1, 5] is to allow the modeller to construct a large model by using a number of small CPNs which are related to each other in a well-defined way. At one level, we want to give a simple description of the modelled activity without having to consider internal details about how it is carried out. Moreover, we want to be able to integrate the detailed specification with more crude descriptions and this integration must be done in such a way that it is meaningful to speak about the behaviour of the *complete* net.

Pages and their instances We want to relate individual CPNs to nodes, which are members of other CPNs, and this means that our description will contain a *set* of non-hierarchical CPNs, which we shall call *pages*. A *diagram* is a set of related non-hierarchical CPNs, called *pages*.

A page may have many different *page instances*. These page instances will have their own private markings, which are independent of the markings of the other instances (in a similar way that each procedure call has its own private copies of the local variables in the procedure).

Each hierarchical inscription in a CPN tells us the identity of the *subpage*, i.e. the page which contains the detailed description of the activity modelled by the corresponding substitution node. Each substitution node is said to be a *supernode* (of the corresponding subpage) while the page of substitution node is a superpage (of the corresponding subpage).

Page hierarchy To give an overview of the set of pages, we use a *page hierarchy graph*. This is a directed graph which contains a node for each page and an arc for each direct superpage-subpage relationship. Each node is inscribed by the name of the corresponding page, while each arc is inscribed with the names of the corresponding substitution or invocation nodes. Ellipse shape indicates that all supernodes must be places, box shape that they must be transitions, hexagon shape that they must be place-transition pair and rounded box shape that there is no restriction. Global fusion sets are indicated in the page hierarchy, but page and instance fusion sets are not represented in the page hierarchy, because they involve only a single page.

Substitution nodes The idea of *substitution nodes* is to allow the user to relate a node (and its surrounding arcs) to a more complex CPN, called *subnet*, which usually gives a more precise and detailed description of the activity represented by the substitution nodes.

Each subnet has a number of places called *port nodes* and they constitute the interface with which the subnet communicates with its surroundings. A substitution node has some input nodes and some output nodes called *input* and *output socket nodes*, respectively. To specify the relationship between a substitution node and its subnet, we must describe how the port nodes of the subnet are related to the socket nodes of the substitution node. This is done by providing a *port assignment*. When a port node is assigned to a socket node, the two nodes become identical. Through the input and output ports the subnet can be in communication with its surroundings.

Invocation nodes In contrast to substitution nodes, the invocation nodes are not substituted by their subpage. This means that they can occur and each of their occurrences triggers the creation of a new instance of the subpage. These subpage instances are executed concurrently with the other page instances in the model, until some specified exit condition is reached.

The termination of a recursive substitution is usually triggered by execution of the last statement or by an explicit exit statement. The execution is terminated the first time an exit transition occurs or an exit place receives a token.

Each invocation node can have any page as a subpage. This means that the invocation hierarchy is allowed to contain circular (i.e. recursive) dependencies while the substitution hierarchy is demanded to be acyclic (to avoid infinite substitution).

Fusion sets The main idea behind *fusion* is to allow the modeller conceptually to fold a set of nodes into a single node without graphically having to represent them as a single object. A fusion is obtained by defining a fusion set containing an arbitrary number of places or an arbitrary number of transitions. The nodes that participate in such a *fusion set* may belong to a single page or to several different pages. There are three different kind of fusion sets: *global fusion sets* are allowed to have members from many different pages, *page fusion sets* and *instance fusion sets* only have members from a single page. A page fusion unifies all the instances of its nodes (independently of the page instance at which the node instance appear). An instance fusion set only identifies node instances that belong to the same page instance.

2.2.2 Formal definition of generalized hierarchical coloured Petri nets

A hierarchical CPN consists of a set of subnets. Each subnet $s \in S$ is a non-hierarchical CPN, i.e., a tuple:

$$(\Sigma_s, P_s, T_s, A_s, N_s, C_s, G_s, E_s, IN_s) \ .$$

In here we must to note that a substitution or invocation node(s) should *not* be considered as a normal node(s). These are special elements signifying with special tags.

When we talk about the elements of the entire hierarchical CPN, we use the following notations:

- $\Sigma = \bigcup_{s\in S} \Sigma_s$, $P = \bigcup_{s\in S} P_s$, $T = \bigcup_{s\in S} T_s$, $A = \bigcup_{s\in S} A_s$, where it should be noted that the sets of colour sets usually have common elements, while the sets of net elements (P_s, T_s, A_s) are required to be disjoint.
- $X = P \cup T$ is the set of *nodes*.
- $X \in [X \rightarrow X_S]$ maps each node x to the set of its *surrounding nodes*, i.e., the nodes that are connected to x by an arc: $X(x) = \{x' \in X | \exists a \in A : [N(a) = (x, x') \vee N(a) = (x', x)]\}$.
- $In \in [X \rightarrow X_S]$ maps each node x to the set of its *input nodes*, i.e., the nodes that are connected to x by an arc: $In(x) = \{x' \in X | \exists a \in A : N(a) = (x', x)\}$.
- $Out \in [X \rightarrow X_S]$ maps each node x to the set of its *output nodes*, i.e., the nodes that are connected to x by an arc: $Out(x) = \{x' \in X | \exists a \in A : N(a) = (x, x')\}$.
- $ST(x_1, x_2), x_1 \in P, x_2 \in T$ is the *socket type* function, which maps from pairs of socket nodes and substitution nodes into $\{in, out, i/o\}$.

$$ST(x_1, x_2) = \begin{cases} in & if\ x_1 \in (In(x_2) - Out(x_2)), \\ out & if\ x_1 \in (Out(x_2) - In(x_2)), \\ i/o & if\ x_1 \in (In(x_2) \cup Out(x_2)) \end{cases}$$

A generalized hierarchical coloured Petri-net is then a tuple

$$HCPN = (S, SN, SA, PN, PT, PA, FS, FT, PP)$$

satisfying the following requirements:

(i) S is a finite set of *pages (subnets)* such that:
- Each page $s \in S$ is a non-hierarchical CPN: $(\Sigma_s, P_s, T_s, A_s, N_s, C_s, G_s, E_s, IN_s)$.
- The sets of net elements are pairwise disjoint: $\forall s_1, s_2 \in S : [s_1 \neq s_2 \Rightarrow (P_{s_1} \cup T_{s_1} \cup A_{s_1}) \cap (P_{s_2} \cup T_{s_2} \cup A_{s_2}) = \emptyset]$.

(ii) $SN \subseteq T \cup P \cup (T \times P) \cup (P \times T)$ is a set of *substitution nodes.*

(iii) SA is a *page assignment* function. It is defined from SN into S such that: [1]
- When we allow invocation nodes: page is a subpage of itself: $\{s_0 s_1 \dots s_n \in S^* | n \in \mathbb{N}_+ \wedge s_0 = s_n \wedge \forall k \in \mathbb{N}_+ : s_k \in SA(SN_{s_{k-1}})\} \neq \emptyset$.
- When we do NOT allow invocation nodes: No page is a subpage of itself: $\{s_0 s_1 \dots s_n \in S^* | n \in \mathbb{N}_+ \wedge s_0 = s_n \wedge \forall k \in 1 \dots n : s_k \in SA(SN_{s_{k-1}})\} = \emptyset$.

(iv) $PN \subseteq P \cup T$ is a set of *port nodes.*

(v) PT is a *port type* function. It is defined from PN into $\{in, out, i/o\}$.

[1] S^* denotes all finite sequences with element from S.

(vi) PA is a *port assignment* function. It is defined from SN into binary relations such that:
- Socket nodes are related to port nodes :
  $\forall x \in SN : PA(x) \subseteq X(x) \times PN_{SA(x)}$.
- Socket nodes are of the correct type:
  $\forall x \in SN : \forall (x_1, x_2) \in PA(x) : [PT(x_2) \in \{in, out, i/o\} \Rightarrow ST(x_1, x) = PT(x_2)]$.
- Related nodes have identical colour sets, equivalent initialization expressions, equivalent guards, equivalent arc expressions:
  (a) $\forall t \in SN \cup T : \forall (p_1, p_2) \in PA(t) : [C(p_1) = C(p_2) \wedge I(p_1)_{<>} = I(p_2)_{<>}]$.
  (b) $\forall p \in SN \cup P : \forall (t_1, t_2) \in PA(p) : [G(t_1) = G(t_2) \wedge$
  $$\begin{cases} A(t_1, p) = A(t_2, p) & if\ PT(t_2) = in, \\ A(p, t_1) = A(p, t_2) & if\ PT(t_2) = out, \\ PT(t_2) = in \wedge PT(t_2) = out & if\ PT(t_2) = i/o. \end{cases} \quad].$$
  (c) $\forall (p, t) \in SN$ and $\forall (t, p) \in SN$: for t the (a) is required and for p the (b) is required.

(vii) $FS \subseteq P_s$ is a finite set of *fusion sets* such that:
- Members of fusion set have identical colour sets and equivalent initialization expressions:
  $\forall fs \in FS : \forall p_1, p_2 \in fs : [C(p_1) = C(p_2) \wedge I(p_1)_{<>} = I(p_2)_{<>}]$.

(viii) FT is a *fusion type* function. It is defined from fusion sets into $\{global, page, instance\}$ such that:
- Page and instance fusion sets belong to a single page:
  $\forall fs \in FS : [FT(fs) \neq global \Rightarrow \exists s \in S : fs \subseteq P_s]$.

(ix) $PP \in S_{MS}$ is a multi-set of *prime pages.*

2.2.3 Instances

A page $s \in S$ may have many different page instances. The set of *page instances* of a page $s \in S$ is the set SI_s of all triples $(s^*, n^*, x_1 x_2 \dots x_m)$ that satisfy the following requirements:

(i) $s^* \in PP \wedge n^* \in 1 \dots PP(s^*)$.

(ii) $x_1 x_2 \dots x_m$ is a sequence of substitution nodes, with $m \in \mathbb{N}$, such that:
- $m = 0 \Rightarrow s^* = s$
- $m > 0 \Rightarrow (x_1 \in SN_{s^*} \wedge [k \in 2 \dots m \Rightarrow x_k \in SN_{SA(x_{k-1})}] \wedge SA(x_m) = s)$.

Page instances where the third component is the empty sequence are said to be *prime* page instances, while all others are *secondary* page instances.

When a page has several page instances, these each have their own instances of the corresponding places, transitions and arcs. However, it should be noted that substitution nodes and their surrounding arcs do not have instances because they are replaced by instances of the corresponding direct subpages.

The set of *place instances* of a page $s \in S$ is the set PI_s of all pair (p, id) that satisfy the following requirements:

(i) $p \in P_s \setminus SN_s$.

(ii) $id \in SI_s$.

Some of the place instances are related to each other, because of the fusion sets and because of the port assignments. Two place instances (p_1, id_1) and (p_2, id_2) are related by a fusion set $fs \in FS$ iff the following conditions are fulfilled:

- The two original places must both belong to fs, i.e. $p_1, p_2 \in fs$.
- When fs is an instance fusion set, the two place instances must belong to the same page instance, i.e. $id_1 = id_2$.
- When fs is a global fusion set or a page fusion set, there is no restriction on the relation between id_1 and id_2.

Analogously, two place instances (p_1, id_1) and (p_2, id_2) are related by the port assignment of a substitution transition $t \in SN$ iff the following conditions are fulfilled:

- The two original places must be related by the port assignment, i.e. $(p_1, p_2) \in PA(t)$.
- The page instance $id_2 = (s_2, n_2, tt_2)$ of the port node p_2 must not be a prime page instance because of the existance of the substitution transition t on the page instance $id_1 = (s_1, n_1, tt_1)$ of the socket node p_1. This means that the two page instance must originate from the same prime page instance, i.e. that $(s_1, n_1) = (s_2, n_2)$- Moreover, id_2 must have the same sequence of substitution nodes as id_1, expect that t has been added, i.e. $tt_1^{\wedge}t = tt_2$, where $^{\wedge}$ denotes concatenation of sequences.

The set of *transition instances* of a page $s \in S$ is the set TI_s of all pair (t, id) that satisfy the following requirements:

(i) $t \in T_s \setminus SN_s$.

(ii) $id \in SI_s$.

The set of *arc instances* of a page $s \in S$ is the set AI_s of all pair (a, id) that satisfy the following requirements:

(i) $a \in A_s \setminus A(SN_s)$.

(ii) $id \in SI_s$.

Each place instance, transition instance and arc instance is said to *belong* to the page instance in its second component.

We define token elements, binding elements, markings, steps, initial markings, reachability and occurrence sequences analogously to the corresponding concepts for non-hierarchical CPNs.

2.3 Examples for the CPN hierarchies

In the following we show some illustrative examples for the different types of CPN hierarchies. We have omitted most of the other net inscriptions, e.g. the initial markings, since we are focusing more on the net structure than on the details of colour sets, arc expressions and guards, etc.

2.3.1 Substitution nodes

The relationship between each of the substitution nodes and the corresponding subpages is defined by the inscription next to the HS tag (HS = Hierarchy + Substitution). This inscription tells the name of the subpage and it describes how each of the nodes surrounding the compound node is assigned to one of the border nodes of the subpage. The interfaces of subnets are defined by the B-tags (B = Border) and the inscription (*in, out, i/o*) next to them.

Substitution transition The page in the left part of Fig. 1 represents a supernet, and the subnet of the substituted transition T can be see in the right part. Below it is shown how the hierarchical CPN in Fig. 1 is represented as a many-tuple. To save space (and time) we don't give the tuple-definition of the individual pages.

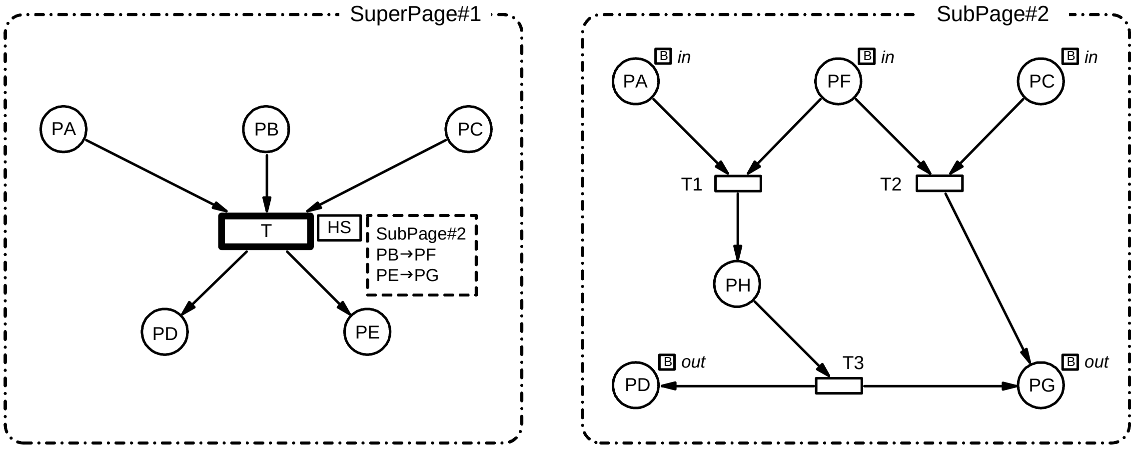

Figure 1: Substitution transition

The tuple $HCPN = (S, SN, SA, PN, PT, PA, FS, FT, PP)$ of the hierarchical CPN shown in Fig. 1 is defined as below:

(i) $S = \{SuperPage\#1, SubPage\#2\}$

(ii) $SN = \{T@SuperPage\#1\}$

(iii) $SA(x) = SubPage\#2 \quad if\ x = T@SuperPage\#1.$

(iv) $PN = \{PA@SubPage\#2, PF@SubPage\#2, PC@SubPage\#2, PD@SubPage\#2, PG@SubPage\#2\}$

(v) $PT(x) = \begin{cases} in & if\ x \in \{PA@SubPage\#2, PF@SubPage\#2, PC@SubPage\#2\} \\ out & if\ x \in \{PD@SubPage\#2, PG@SubPage\#2\}. \end{cases}$

(vi) $PA(x) = \{\ (PA@SuperPage\#1, PA@SubPage\#2), (PB@SuperPage\#1, PF@SubPage\#2), (PC@SuperPage\#1, PC@SubPage\#2), (PD@SuperPage\#1, PD@SubPage\#2), (PE@SuperPage\#1, PG@SubPage\#2)\ \} \quad if\ x = T@SuperPage\#1.$

(vii) $FS = \emptyset$

(viii) $FT(fs) = global \quad for\ all\ fs \in FS$

(ix) $PP = 1`SuperPage\#1.$

Substitution place The page in the left part of Fig. 2 represents a supernet, and the subnet of the substituted place can be seen in the right part. Below we show how the hierarchical CPN in Fig. 2 is represented as a many-tuple. To save space (and time) we don't give the tuple-definition of the

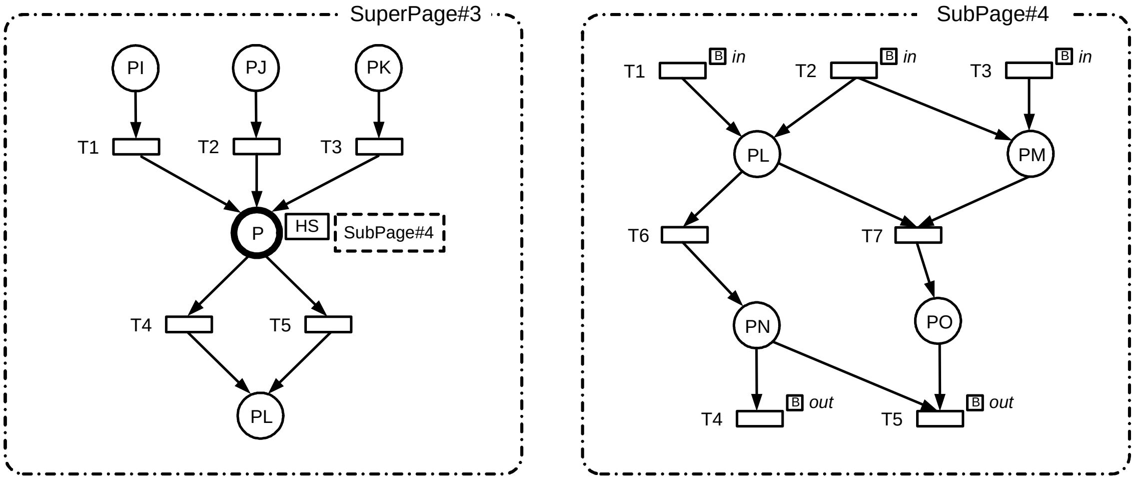

Figure 2: Substitution place

individual pages.

The tuple $HCPN = (S, SN, SA, PN, PT, PA, FS, FT, PP)$ of the hierarchical CPN shown in Fig. 2 is defined as below:

(i) $S = \{SuperPage\#3, SubPage\#4\}$

(ii) $SN = \{P@SuperPage\#3\}$

(iii) $SA(x) = SubPage\#4 \quad if\ x = P@SuperPage\#3.$

(iv) $PN = \{T1@SubPage\#4, T2@SubPage\#4, T3@SubPage\#4, T4@SubPage\#4, T5@SubPage\#4\}$

(v) $PT(x) = \begin{cases} in & if\ x \in \{T1@SubPage\#4, T2@SubPage\#4, T3@SubPage\#4\} \\ out & if\ x \in \{T4@SubPage\#4, T5@SubPage\#4\}. \end{cases}$

(vi) $PA(x) = \{\ (T1@SuperPage\#3, T1@SubPage\#4), (T2@SuperPage\#3, T2@SubPage\#4),$
$(T3@SuperPage\#3, T3@SubPage\#4), (T4@SuperPage\#3, T4@SubPage\#4),$
$(T5@SuperPage\#3, T5@SubPage\#4)\ \} \quad if\ x = P@SuperPage\#3.$

(vii) $FS = \emptyset$

(viii) $FT(fs) = global \quad for\ all\ fs \in FS$

(ix) $PP = 1`SuperPage\#3.$

Substitution place-transition pair The page in the left part of Fig. 3 presents a supernet, and the subnet of the substituted place-transition pair can be seen in the right part. The hexagon represents the substituted place-transition pair as a composite node. The order of the place and the transition in the pair depends on the input and output node types (i.e. which are places and which are transitions). Below it is shown how the hierarchical CPN in Fig. 3 represented as a many-tuple.

The tuple $HCPN = (S, SN, SA, PN, PT, PA, FS, FT, PP)$ of the hierarchical CPN shown in Fig. 3 is defined as below:

(i) $S = \{SuperPage\#5, SubPage\#6\}$

(ii) $SN = \{(T1, P1)@SuperPage\#5, (T2, P2)@SuperPage\#5\}$

(iii) $SA(x) = SubPage\#6 \quad if\ x \in \{(T1, P1)@SuperPage\#5, (T2, P2)@SuperPage\#5, T2@SubPage\#8\}.$

(iv) $PN = \{P3@SubPage\#6, P4@SubPage\#6, T5@SubPage\#6, T6@SubPage\#6\}$

(v) $PT(x) = \begin{cases} in & if\ x \in \{P3@SubPage\#6, P4@SubPage\#6\} \\ out & if\ x \in \{T5@SubPage\#6, T6@SubPage\#6\}. \end{cases}$

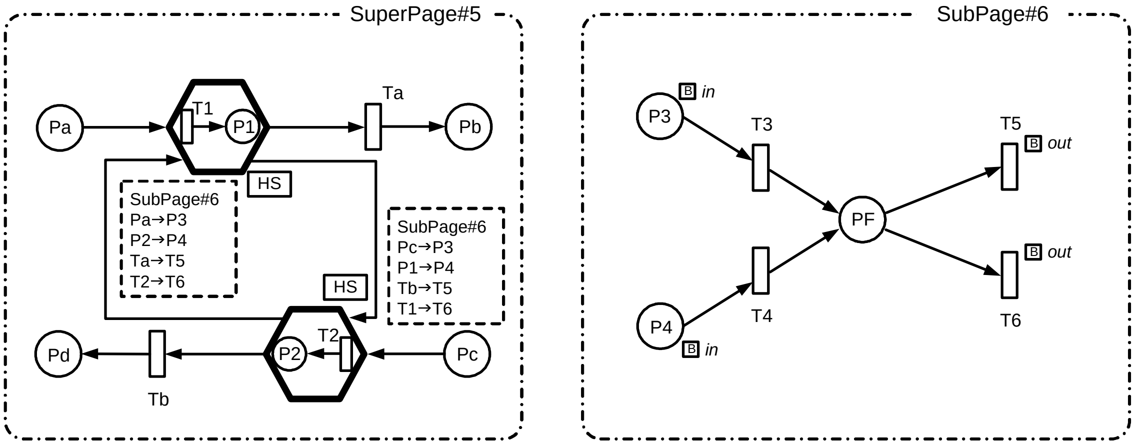

Figure 3: Substitution place-transition

$$\text{(vi)}\quad PA(x) = \begin{cases} \{(Pa@SuperPage\#5, P3@SubPage\#6), (P2@SuperPage\#5, P4@SubPage\#6) \\ \quad (Ta@SuperPage\#5, T5@SubPage\#6), (T2@SuperPage\#5, T6@SubPage\#6)\} \\ \qquad \qquad if\ x = (T1, P1)@SuperPage\#5 \\ \{(Pc@SuperPage\#5, P3@SubPage\#6), (P1@SuperPage\#5, P4@SubPage\#6) \\ \quad (Tb@SuperPage\#5, T5@SubPage\#6), (T1@SuperPage\#5, T6@SubPage\#6)\} \\ \qquad \qquad if\ x = (T2, P2)@SuperPage\#5 \end{cases}$$

(vii) $FS = \emptyset$

(viii) $FT(fs) = global \quad for\ all\ fs \in FS$

(ix) $PP = 1`SuperPage\#5.$

In here we note that we do not allow a substitution node to be a neighbor of another substitution node because then it would be impossible to construct an equivalent non-hierarchical CPN by the methods defined above. To solve cases when two substitution nodes come too close together, they can always be separated by inserting an extra place and an extra transitions between them.

2.3.2 Invocation nodes

The invocation nodes are distinguishable by the HI-tags (HI = Hierarchy + Invocation) and the inscription next to them specifies the subpage and the port assignment.

Invocation transition The page in the left part of Fig. 4 presents a supernet, and the subnet of the invocation transition can be seen in the right part. Below we show how the hierarchical CPN in Fig. 3 is represented as a many-tuple.

The tuple $HCPN = (S, SN, SA, PN, PT, PA, FS, FT, PP)$ of the hierarchical CPN shown in Fig. 4 is defined as below:

(i) $S = \{SuperPage\#7, SubPage\#8\}$

(ii) $SN = \{T@SuperPage\#7, T1@SubPage\#8, T2@SubPage\#8\}$

(iii) $SA(x) = SubPage\#8 \quad if\ x \in \{T@SuperPage\#7, T1@SubPage\#8, T2@SubPage\#8\}.$

(iv) $PN = \{Start@SubPage\#8, Stop@SubPage\#8\}$

$$\text{(v)}\quad PT(x) = \begin{cases} in & if\ x \in \{Start@SubPage\#8\} \\ out & if\ x \in \{Stop@SubPage\#8\}. \end{cases}$$

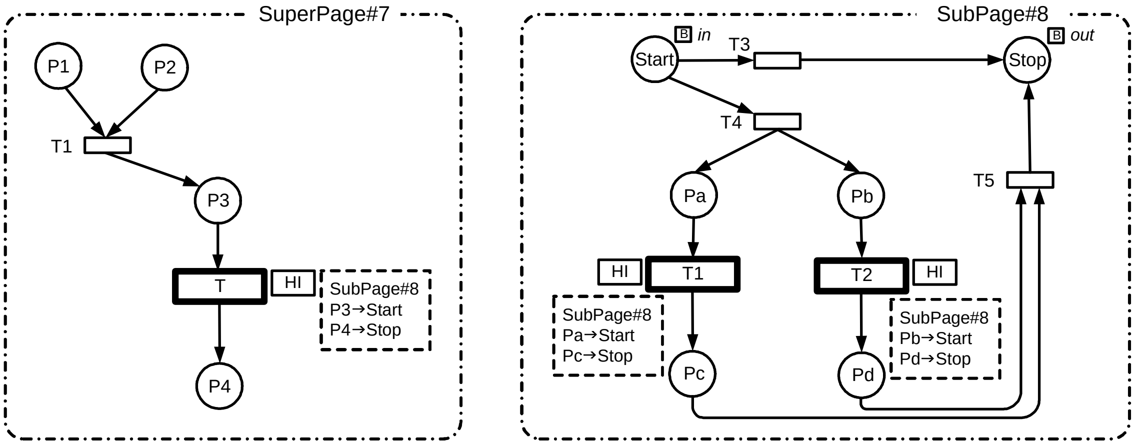

Figure 4: Invocation transition

$$\text{(vi)}\quad PA(x) = \begin{cases} \{(P3@SuperPage\#7, Start@SubPage\#8), & \\ \;(P4@SuperPage\#7, Stop@SubPage\#8)\} & if\ x = T@SuperPage\#7 \\ \{(Pa@SubPage\#8, Start@SubPage\#8), & \\ \;(Pc@SubrPage\#8, Stop@SubPage\#8)\} & if\ x = T1@SubPage\#8 \\ \{(Pb@SubPage\#8, Start@SubPage\#8), & \\ \;(Pd@SubrPage\#8, Stop@SubPage\#8)\} & if\ x = T2@SubPage\#8 \end{cases}$$

(vii) $FS = \emptyset$

(viii) $FT(fs) = global \quad for\ all\ fs \in FS$

(ix) $PP = 1`SuperPage\#7.$

Invocation place Exactly the same set of concepts as above for the invocation transition applies to the invocation place.

2.3.3 Fusion sets

Place fusion This idea is illustrated in Fig. 5 where the left CPN has a fusion set called A. This fusion set contains the fusion set members $A1$ and $A2$ which are distinguishable by the FP-tags (FP = Fusion + Page). $FusA$ is a page fusion set and this means that it is allowed to have only fusion set members from a single page in the diagram. Let us assume that this page has only one instance. Then the equivalent CPN net is shown in Fig. 5. It is obtained by merging $A1$ into $A2$. Intuitively, this semantics means that the places $A1$ and $A2$ share the same marking.

Now let us consider the case where the page of $FusA$ has more than one page instances. Then we have two possibilities. Either we can merge all instances of all fusion set members into a single conceptual node FI-tags (FI = Fusion + Instance), or we can merge them into a node for each instance with FP-tags.

Finally we allow global fusion sets with FG-tags (FG = Fusion + Global). This allows fusion set members from all pages in the diagram and all instances of these nodes are merged into a single conceptual node. This means that a page fusion set is a special case of global fusion set.

The part of the tuple $HCPN = (S, SN, SA, PN, PT, PA, FS, FT, PP)$ of the hierarchical CPN shown in Fig. 1 is defined as below:

(i) $S = \{\ldots, SubPage\#10, \ldots\}$

$\vdots$

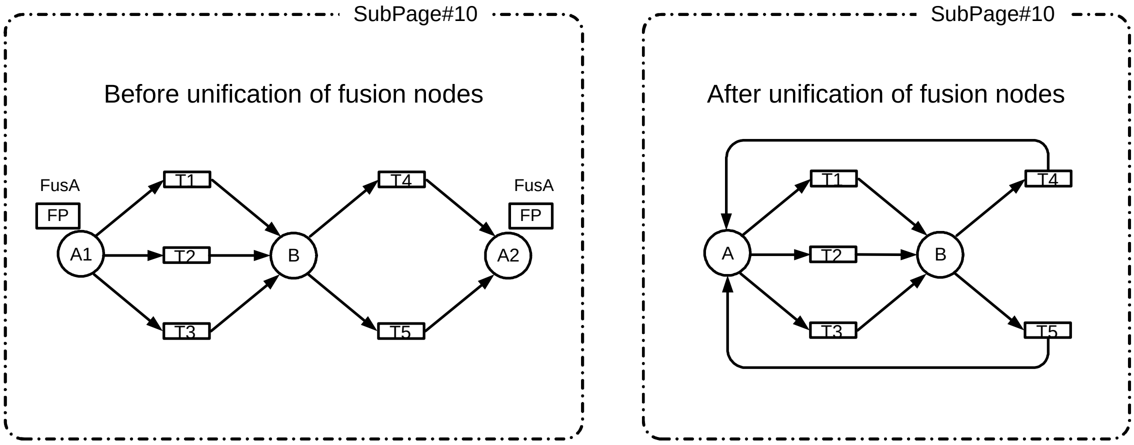

Figure 5: Fusion sets

(vii) $FS = \{\{A1@SubPage\#10, A2@SubPage\#10\}\}$

(viii) $FT(fs) = SubPage\#10 \quad for\ all\ fs \in FS$

$\vdots$

The members of a fusion set must be comparable to each other. For places this means that they must have the same colour set and the same initial marking. It also means that they either all must be ordinary places, all be substitution places or all be invocation places, and in the two latter cases they must all have the same subpage. Fusion of substitution places are useful, when we want to apply the same instance at several locations in the diagram.

Transition fusion Exactly the same set of concepts as above for place fusion applies to transition fusion. For transitions we do not demand that the guards are identical. Instead, we form the conjunction of the guards. The members of a transition fusion set must either all be ordinary transitions, all be substitution transitions or all be invocation transitions. In the two latter cases they must all have the same subpage. In addition, it is not allowed to use global and page fusion for transitions, which appear on subpages of invocation transitions (or on subpages of such pages).

3 Case study

In the following example a simple multi-scale process model will be used to demonstrate the top-down model building by using generalized hierarchical coloured Petri net models.

3.1 The countercurrent heat exchanger: models of various levels of detail

3.1.1 The single cell model of the HE

Consider a countercurrent heat exchanger (HE) shown in Fig. 6, where the cold liquid stream is being heated by a hot liquid stream. On the top level the heat exchanger is modelled by a single pair of perfectly stirred lumps forming a so called heat exchanger cell. Each cell consists of two perfectly stirred tanks with in- and outflows. The two tanks are connected by a heat transfer area between them.

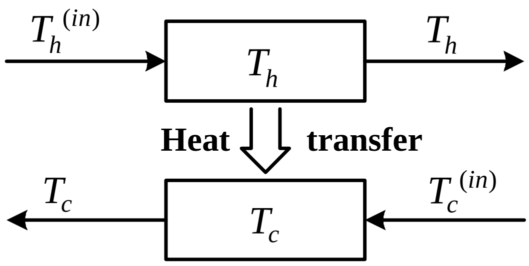

Figure 6: The diagram of the single cell countercurrent heat exchanger

Modelling assumptions

- The overall mass (volume) of the liquids on both sides is constant.
- No diffusion takes place.
- No heat lost to the surroundings.
- Heat transfer coefficients are constant.
- Specific heats and densities are constant.
- Both liquids are in plug flow.
- The heat exchanger is described as a CSTR pair of hot and cold liquid volumes.

Balance volumes We consider perfectly mixed balance volumes with equal holdups for each of the hot and cold sides. The subscripts h and c denote the hot and cold sides respectively.

Model Equations:

Variables

$$(T_h(t)\ , T_c(t)) \qquad ,\ 0 \leq t \tag{1}$$

where $T_h(t)$ and $T_c(t)$ is the hot and the cold side temperature in the tank and t is time.

Energy balances for the hot side

$$\frac{dT_h(t)}{dt} = \frac{F_h}{V_h}\left(T_h^{(in)} - T_h\right) - \frac{KA}{c_{p_h}\rho_h V_h}\left(T_h(t) - T_c(t)\right) \tag{2}$$

$T_h^{(in)}(t)$ is the hot liquid inlet temperature to the heat exchanger.

Energy balances for the cold side

$$\frac{dT_c(t)}{dt} = \frac{F_c}{V_c}\left(T_c^{(in)}(t) - T_c(t)\right) - \frac{KA}{c_{p_c}\rho_c V_c}\left(T_c(t) - T_h(t)\right) \tag{3}$$

$T_c^{(in)}(t)$ is the cold liquid inlet temperature to the heat exchanger.

Here F_h and F_c are the flowrates, V_h and V_c are the volumes, A is the heat transfer area, c_{p_h} and c_{p_c} are the specific heats, ρ_h and ρ_c are the densities, K is the heat transfer coefficient.

3.1.2 Cascade model of the HE

If we want to refine the model, the heat exchanger is divided into n equal parts, and it gives the cascade model of the HE (see in Fig. 7). The cascade model consists of heat exchanger (HE) cells shown in Fig. 8.

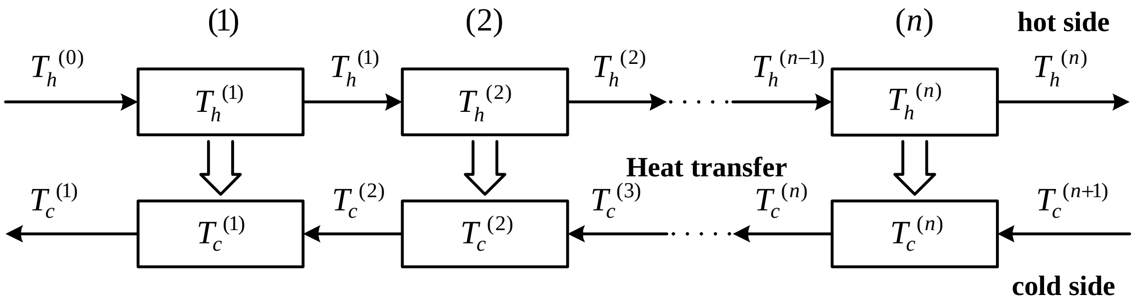

Figure 7: The cascade model of the heat exchanger

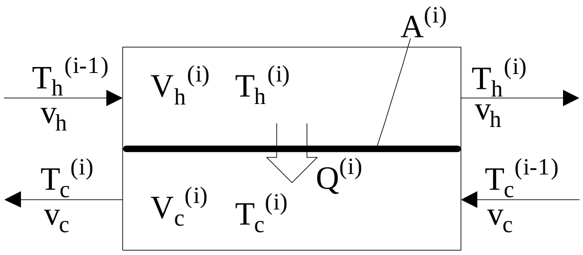

Figure 8: The diagram of a heat exchanger cell

Modelling assumptions Replace the last item of the modelling assumptions with the following item:

- The heat exchanger is described as a sequence of n CSTR pairs of hot and cold liquid volumes ($n = 3$).

Balance volumes We consider three perfectly mixed balance volumes with equal holdups for each of the hot and cold sides.

Model Equations:

Variables

$$\left(T_h^{(k)}(t)\ , T_c^{(k)}(t)\ ,\ k = 1, 2, 3\right) \qquad ,\ 0 \leq t \tag{4}$$

where $T_h^{(k)}(t)$ and $T_c^{(k)}(t)$ is the hot and the cold side temperature in the kth tank pair respectively and t is time.

Energy balances for the hot side

$$\frac{dT_h^{(k)}(t)}{dt} = \frac{F_h}{V_h}^{(k)} \left(T_h^{(k-1)} - T_h^{(k)}\right) - \frac{K^{(k)}A^{(k)}}{c_{p_h}^{(k)}\rho_h^{(k)}V_h^{(k)}} \left(T_h^{(k)} - T_c^{(k)}\right) \quad k = 1,2,3 \;\; , \; T_h^{(0)}(t) = T_h^{(i)}(t) \qquad (5)$$

$T_h^{(i)}(t)$ is the hot liquid inlet temperature to the heat exchanger.

Energy balances for the cold side

$$\frac{dT_c^{(k)}(t)}{dt} = \frac{F_c}{V_c}^{(k)} \left(T_c^{(k+1)} - T_c^{(k)}\right) - \frac{K^{(k)}A^{(k)}}{c_{p_c}^{(k)}\rho_c^{(k)}V_c^{(k)}} \left(T_c^{(k)} - T_h^{(k)}\right) \quad k = 1,2,3 \;\; , \; T_h^{(4)}(t) = T_c^{(i)}(t) \qquad (6)$$

$T_c^{(i)}(t)$ is the cold liquid inlet temperature to the heat exchanger. Note that the cold stream flows in the direction of descending volume indices.

Initial conditions

$$T_h^{(k)}(0) = f_1^{(k)} \quad , \quad k = 1,2,3 \qquad (7)$$

$$T_c^{(k)}(0) = f_2^{(k)} \quad , \quad k = 1,2,3 \qquad (8)$$

where the values of $f_1^{(k)}$, $k = 1,2,3$ and $f_2^{(k)}$, $k = 1,2,3$ are given.

3.2 The CPN models of the countercurrent heat exchanger

3.2.1 Single cell models

First consider a single cell hierarchical CPN model in Fig. 9. The *SuperPage#1* as prime page of the single cell HE model is divided into two main parts: the hot side and the cold side parts which are connected. The left part of Fig. 9 shows the top level of the one cell HE, the right part shows the subpages *HotCell#2* and *ColdCell#3* of the substitution place-transition pairs. Subnets denoted by hexagons contain the hot and cold side cell details, respectively. We use substitution place-transition pairs for hierarchical decomposition.

The hot or cold side balance volume is described by a node containing its state variable value, and the incoming transitions realized the applications of input data. The outgoing arcs serve the information to the surrounding about the balance volume.

3.2.2 Cascade models

The next step is to divide the single cell model into a n-cell model. The CPN realization can be seen in Fig. 10. The prime page of the cascade CPN model is the page *SuperPage_Cascade#3*, which is similar to the prime page of the single cell model, but the substitution subnets are changed. The new substitution subnets *SubPage_Casc_hot#4* and *SubPage_Casc_cold#5* in Fig. 10 describe the division of balance volumes of the single cell model. These inserted subnets do not describe the cell details. The hot and cold cell details are substituted to the corresponding substitution nodes by using subnets *HotCell#2* and *ColdCell#3* (see in Fig. 9).

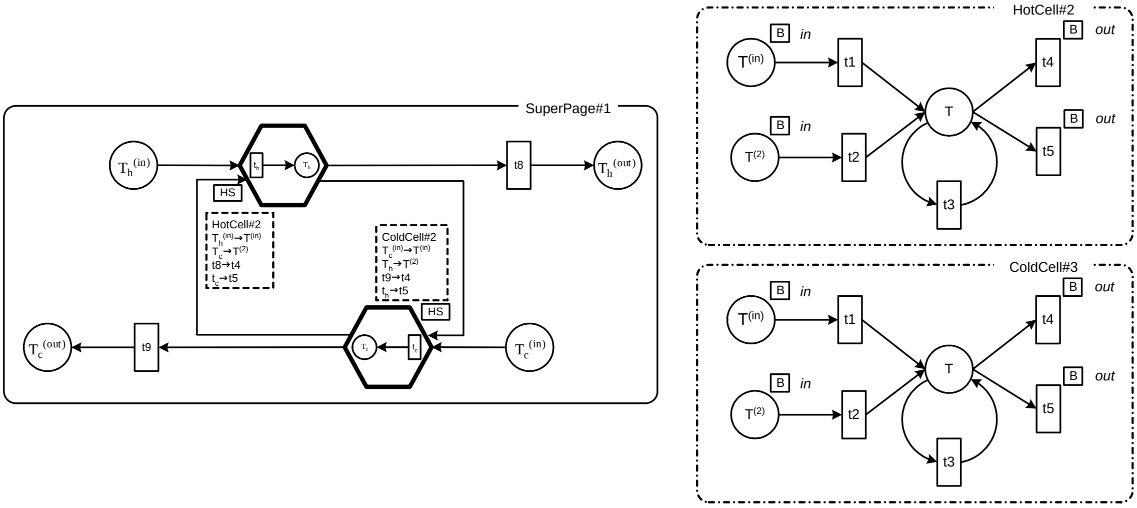

Figure 9: The hierarchical CPN model of one cell heat exchanger

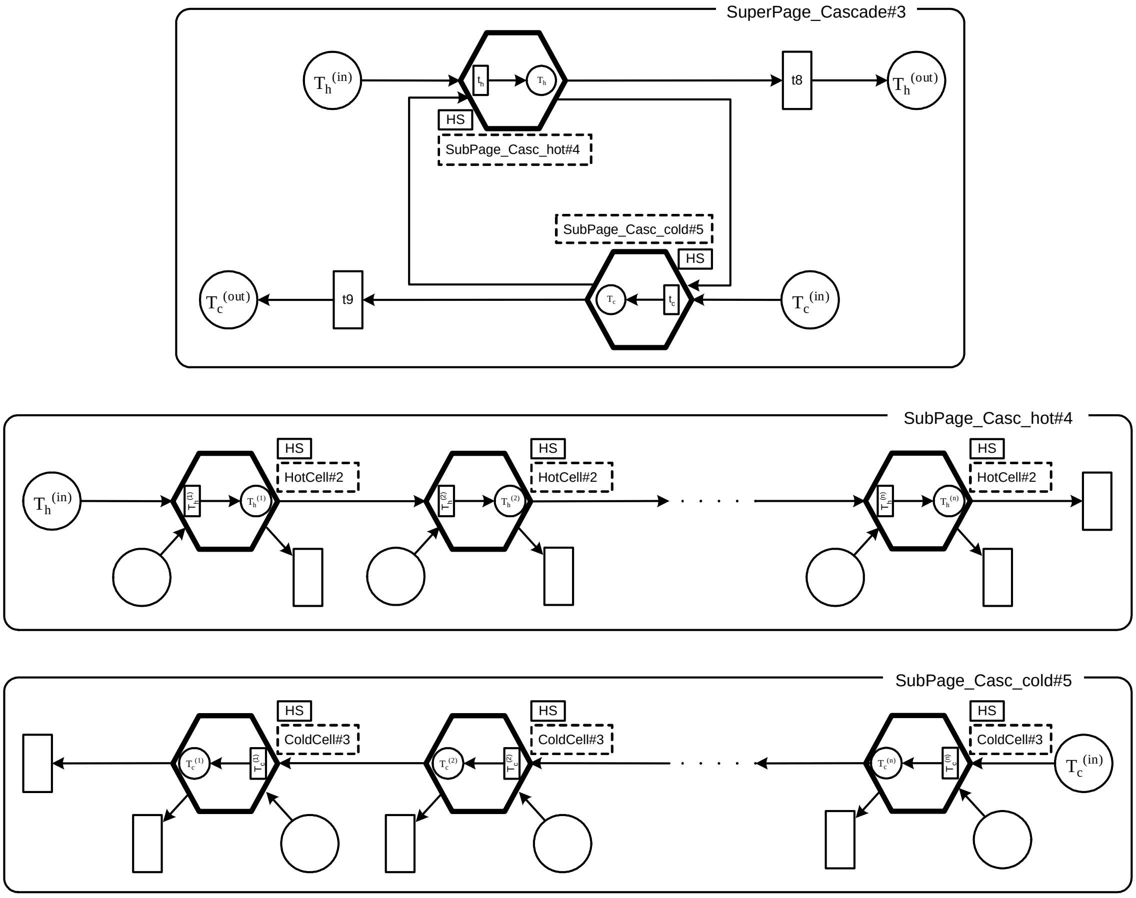

Figure 10: The hierarchical CPN model of the cascade heat exchanger

Finally, the whole CPN model of the cascade HE can be see in Fig. 11 after the substitutions of the corresponding subnets.

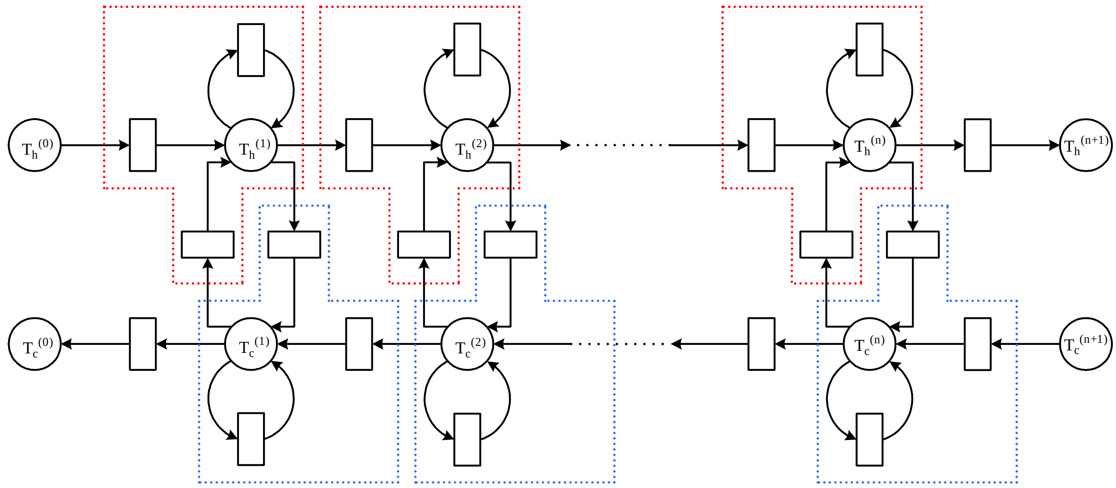

Figure 11: The CPN model of the cascade heat exchanger after the substitutions

4 Conclusion and future work

We have proposed extension of coloured petri nets (CPNs) by hierarchy concept in seven ways:

- substitution transition,
- substitution place,
- substitution place-transition pair,
- invocation transition,
- invocation place,
- fusion place,
- fusion transition.

Hierarchical CPN is formulated and illustrated by examples. The presented experiments are very difficult and rather complex, but we believe that our work will form the base for multi-scale process description for diagnostics.

References

[1] Huber P., K. Jensen, and R.M. Shapiro. Hierarchies in coloured petri nets. *Lecture Notes in Computer Science 483, Advances in Petri Nets 1990*, pages 313–341, 1990.

[2] Kurt Jensen and Grzegorz Rosenberg. *High-level Petri nets: Theory and Application.* Springer-Verlag, 1991.

[3] Kurt Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use Volume 1.* Springer-Verlag, 1992.

[4] Kurt Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use Volume 2.* Springer-Verlag, 1995.

[5] Lakos C. A. The role of substitution places in herarchical coloured petri nets. Technical Report TR93-7, Computer Science Department, University of Tasmania, 1993.