Expanding the Use of Structuring : Formal Justification for Working on Subnets*

Claudia Ermel **Maike Gajewsky**
Institut für Kommunikation und Softwaretechnik
Technische Universität Berlin
e-mail: {lieske, magda }@cs.tu-berlin.de

keywords: algebraic high-level nets, structuring, verification, category theory

Abstract

We here present a new method for algebraic high-level nets with categorical structuring techniques. It allows deducing the structure of the composed net from the structure of its subnets with regard to the composition. This enables compositional working on subnets, as for instance verification, analysis or simulation. Thus, results being valid for subnets are still valid for the entire net without actually constructing it. The main theorem states under which assumptions the structure of a subnet is preserved in the composed net. We then give an example using this method for the verification of requirements with a case study of a medical information system.

1 MOTIVATION

In the area of software engineering it is common sense, that structuring techniques are indispensable for coping with complex systems. The system is decomposed into logically related parts, which is reflected by the component models. Thus comprehension of the complex system is reduced to comprehension of its components and their interaction. In the field of Petri nets the concept of structuring is responded by numerous techniques like for instance hierarchies ([Jen92]), refinement mechanisms for places and transistions ([Feh92]), categorical constructions ([MM90, PER95]), etc ([Kin95]). These all agree in the point, that the entire net is given implicitly by its component nets and their combination.

This concept is usually abandoned when it comes to examination of the model. In order to profit from the large variety of efficient analysis and verification techniques (see e.g. [Rei85]), the entire model actually has to be composed. This seems to undermine the use of structuring, especially when taking into account, that composition may easily lead to nets and data type specifications too complex to handle.

To overcome this conflict, usually component nets are examined and it is assumed that the entire model inherits its properties. This approach can inspire confidence in the entire model. But since the transfer of properties from a part to the whole is not yet well investigated, proving its correctness is problematic.

In many cases verification exploits the fact that a specific place is in the predomain (or postdomain) of a transition. These facts are typically used for the verification of safety properties. While these structural properties can easily be extracted from an unstructured, i.e. flat, net, they are not at all obvious in a structured net. Usually, the entire net actually has to be constructed.

We here pursue a different approach. In order to avoid actual composition of the entire net we infer the structure of the entire model from the structure of its components and the way they are put together. This is the main idea of this paper, where we concentrate on the relation of structuring and verification. This can be regarded as a first step to achieve compatibility between structuring and other techniques as simulation, analysis or verification. In this sense, the use of structuring is expanded. The main benefit of this approach is, that it permits profitting from structuring in the sense of identifying logically relevant parts of the entire net.

The main impulse for this paper was given as we tried to verify requirements for a large case study of a medical information system, which is modelled by algebraic high-level nets ([Erm96]). The entire Petri net model of this case study is not explicitly given, meaning that there is a multitude of component nets and instructions of how to compose them. We can easily identify those component nets, that are relevant for a requirement. Nevertheless, we have to construct the entire net in order to see their interrelation. Furthermore, composition may destroy some relevant property of a component net. Thus, we have been looking for preconditions of structuring operations, that guarantee preservation of this property in the context of the entire net. This leads to the following method: for proving an assertion a concerning

*This work is part of the joint research project "DFG-Forschergruppe PETRINETZ-TECHNOLOGIE" between H. Weber (coordinator), H. Ehrig (both from Technische Universität Berlin) and W. Reisig (Humboldt-Universität zu Berlin), supported by the German Research Council (DFG). In fact, it is motivated by one of the case studies used in this project.

the structure of the composed net we prove a for a component net and that the composition does not violate this assertion. Thus, compositional verification becomes possible. We illustrate this approach by verifying a requirement of the mentioned case study.
The paper is organized as follows: In section 2 we review the notion of algebraic high-level nets. Especially, the horizontal structuring technique *union*, which is realized by the categorical pushout construction is defined and explained.
We then formally substantiate structural properties of the pushout object, the result of applying *union* techniques in section 3. These properties depend only on the nets to be composed and the interface net. The main theorem states under which circumstances parts of the structure of the composed net are inherited from the corresponding component net.
In section 4 we introduce the case study of a medical information system that motivated this paper. We sketch the verification of a requirement of this system applying the main theorem of section 3.
A conclusion is drawn in section 5 and figures can be found in section 6.

2 REVIEW OF STRUCTURING TECHNIQUES FOR ALGEBRAIC HIGH-LEVEL-NETS

In the context of Petri nets, category theory has been used in literature to formulate structural properties of specific net classes, to study compositionality and the relation to other formal descriptions. One of the main issues for the practical use of categorical specification formalisms is the possibility of horizontal and vertical structuring. For the concept of algebraic high-level (AHL) nets as used in this paper, structuring techniques are formulated within the frame of high-level replacement systems [EL93]. Results from the theory of AHL-net transformations [PER95] comprise horizontal structuring techniques like *union* and *fusion*, concurrency properties of transformations like local confluence and parallelism, and compatibility of horizontal structuring with AHL net abstraction/refinement steps in the sense of software engineering.

This section contains basic definitions concerning the notion of, morphisms and the structuring technique *union*. We here refer to the notion of AHL-net given in [PER95]. Further information can be found for instance in [Vau86, KS91, Rei91, EPR94, Lil94].

Definition 1 (Algebraic High-Level Net)
An algebraic high-level net, AHL-net for short, $N = (SPEC, P, T, pre, post, cond, A)$ consists of an algebraic specification $SPEC = (S, OP, E)$ in the sense of [EM85], sets P and T (places and transitions, respectively), functions $pre, post : T \to (T_{OP}(X) \times P)^{\oplus}$ assigning to each transition t an element of the commutative monoid over the cartesian product of terms $T_{OP}(X)$ with variables in X and the set of places P and a function $cond : T \to \mathcal{P}_{fin}(EQNS(SIG))$ assigning to each transition t a finite set of equations over $SIG = (S, OP)$, the signature of $SPEC$, and a $SPEC$-algebra A. N can be represented by the following diagram

$$\mathcal{P}_{fin}(EQNS(SIG)) \xleftarrow{cond} T \underset{post}{\overset{pre}{\rightrightarrows}} (T_{OP}(X) \times P)^{\oplus}$$

□

Remark: An element a of the commutative monoid $Z^{\oplus}$ over some alphabet $Z = \{z_1, z_2, \ldots z_n\}$ can be represented as the linear sum $a = \Sigma_{i=1}^{n}\ a_i z_i$ with coefficients $a_i \in \mathbb{N}$. Addition of elements $a, b \in Z^{\oplus}$ is defined as componentwise addition of coefficients and $a \leq b$ iff for all coefficients $a_i \leq b_i$ holds $(i \leq n)$. Finally, $z_j \in a$ iff $a_j > 0$.
Remark: The notion of commutative monoids corresponds to multisets.

Definition 2 (Morphisms between AHL-nets)

A morphism $f : N_1 \to N_2$ between two AHL-nets $N_i = (SPEC_i, P_i, T_i, pre_i, post_i, cond_i, A_i), i = 1, 2$ is given by $f = (f_{SPEC}, f_P, f_T, f_A)$, where

- $f_{SPEC} : (SIG_1, E_1) \to (SIG_2, E_2)$ is a specification morphism with $f^{\sharp}_{SPEC}(E_1) \subseteq E_2$, where $f^{\sharp}_{SPEC}$ is the extension of f_{SPEC} to terms and equations.
- $f_P : P_1 \to P_2$ and $f_T : T_1 \to T_2$ are functions on the sets of places, resp. transitions.
- $(f_{SPEC}, f_A) : A_1 \to A_2$ is a generalized homomorphism in the category $\underline{\underline{\textbf{GALG}}}$ of generalized algebras, and $f_A : A_1 \to V_{f_{SPEC}}(A_2)$ is an isomorphism in $\underline{\underline{\textbf{Cat}(SPEC_1)}}$, the category of $SPEC_1$-algebras (for details see [PER95]).

such that the following diagram commutes componentwise (for *pre*- and *post*-function)

$$\begin{array}{ccccc}
\mathcal{P}_{fin}(EQNS(SIG_1)) & \xleftarrow{cond_1} & T_1 & \underset{post_1}{\overset{pre_1}{\rightrightarrows}} & (T_{OP}(X) \times P_1)^{\oplus} \\
\downarrow{\scriptstyle \mathcal{P}_{fin}(f^{\sharp}_{SIG})} & = & \downarrow{\scriptstyle f_T} & = & \downarrow{\scriptstyle (f^{\sharp}_{SIG} \times f_P)^{\oplus}} \\
\mathcal{P}_{fin}(EQNS(SIG_2)) & \xleftarrow{cond_2} & T_2 & \underset{post_2}{\overset{pre_2}{\rightrightarrows}} & (T_{OP}(X) \times P_2)^{\oplus}
\end{array}$$

Morphisms between AHL-nets are called AHL-morphisms. □

In [PER95] it has been shown, that the category $\underline{\underline{\textbf{AHL}}}$ of algebraic high-level-nets, consisting of objects $N = (SPEC, P, T, pre, post, cond, A)$ and morphisms $f = (f_{SPEC}, f_P, f_T, f_A)$ is cocomplete. Thus, pushouts (see def. 3) exist which realize the structuring technique *union*. Intuitively, *union* is the gluing of two nets sharing a common subnet. We first present the definition of pushout, then explain its meaning.

Definition 3 (Pushouts of AHL-Nets)
Given three AHL-nets I, N_1, N_2 and AHL-morphisms $m : I \to N_1$, $m' : I \to N_2$, called *matches*. The pushout $(N_1 \to N_3 \leftarrow N_2)$ of $(N_1 \leftarrow I \to N_2)$ over the *interface net* I is given by the following diagram:

$$\begin{tikzcd} I \arrow[r, "m"] \arrow[d, "m'"'] \arrow[dr, phantom, "(\mathbf{1})"] & N_1 \arrow[d, "g"] \\ N_2 \arrow[r, "g'"'] & N_3 \end{tikzcd}$$

such that the following conditions hold:

Commutativity: $m \circ g = m' \circ g'$, that is $(\mathbf{1})$ commutes.

Universal Property: For a net H and morphisms $h_1 : N_1 \to H$ and $h_2 : N_2 \to H$ with $h_1 \circ m = h_2 \circ m'$ there exists a unique morphism $h : N_3 \to H$ with $h_1 = h \circ g$ and $h_2 = h \circ g'$.

The AHL-morphisms $g : N_1 \to N_3$ and $g' : N_2 \to N_3$ are called *gluing morphisms* and the nets N_1 and N_2 are called *component nets*. □

Remark : The pushout properties imply

Commutativity: This ensures, that images of nodes from the interface net I in N_1 and N_2 are identified (glued) in N_3.

Universal Property: The universal property guarantees, that nothing except the images of I is glued. Furthermore, nothing is added, that has no correspondence in either N_1 or N_2.

Both properties together make sure that *exactly* the images of nodes of I are identified.
The pushout net N_3 roughly consists of the interface net I and the disjoint union of those parts of N_1 and N_2, that are not in the image of the matches. An example of a *union* is depicted in figure 1. The interface net consists of one place which is mapped by the morphisms m and m' into the component nets N_1 and N_2. The composed net N_3 is constructed by joining N_1 and N_2 and gluing of the two places that are images of $place0$. The underlying algebraic specification $SPEC$ remains unchanged in this case.
This example captures the situation where two places in different nets are logically the same. In other approaches this identification is done by naming conventions and often excessively used.

Definition 4 (Union of AHL-Nets)
The *union* of the AHL-nets N_1, N_2 over an interface net I and matches $m : I \to N_1$ and $m' : I \to N_2$ is given by the pushout of $N_1 \xleftarrow{m} I \xrightarrow{m'} N_2$. □

We will consequently use "pushout" and "*union*" as synonyms.

As we now concentrate on the structuring of nets in contrast to the structuring of data-types, we here consider AHL-nets over a fixed data-type $SPEC$. Accordingly we omit the algebra A, because it only influences the firing-modes of transitions, not the static structuring of nets. Correspondingly, the notion of AHL-morphisms $f = (f_{SPEC}, f_P, f_T, f_A)$ given in definition 2 is reduced. This leads to the following definition.

Assumption 5 (Fixed Specification)
An AHL-net N over a fixed algebraic specification $SPEC$ is given by
$N = (SPEC, P_N, T_N, pre_N, post_N, cond_N)$, where its components are defined analogously to defintion 1. A morphism between two AHL-nets $N_i = (SPEC, P_i, T_i, pre_i, post_i, cond_i), i = 1, 2$ is given by $f = (f_P, f_T)$, such that the following diagram commutes componentwise:

$$\begin{tikzcd} \mathcal{P}_{fin}(EQNS(SIG)) \arrow[d, "\mathcal{P}_{fin}(id_{SIG})"'] \arrow[dr, phantom, "="] & T_1 \arrow[l, "cond_1"'] \arrow[r, shift left, "pre_1"] \arrow[r, shift right, "post_1"'] \arrow[d, "f_T"] \arrow[dr, phantom, "="] & (T_{OP}(X) \times P_1)^{\oplus} \arrow[d, "(id_{SIG} \times f_P)^{\oplus}"] \\ \mathcal{P}_{fin}(EQNS(SIG)) & T_2 \arrow[l, "cond_2"'] \arrow[r, shift left, "pre_2"] \arrow[r, shift right, "post_2"'] & (T_{OP}(X) \times P_2)^{\oplus} \end{tikzcd}$$

▼

Remark: AHL-nets and -morphisms over a fixed specification $SPEC$ are a specialization of general AHL-nets and -morphisms over arbitrary specifications as defined in section 2. Therefore, all notions given in the previous section hold as well for AHL-nets and -morphisms over a fixed specification.

3 FORMAL JUSTIFICATION FOR WORKING ON COMPONENT NETS

Using algebraic high-level nets for the specification of a system allows application of *union* as structuring technique. For verification of properties, however, the entire net has to be composed. If, for instance, the assertion is relevant for verification, that a specific place is only in the postdomain of exactly one transition, there seems no getting round the actual composition of the entire net. Even if this property holds in a component net, the application of *union* may destroy it as exemplified for $place0$ in figure 1. Therefore, we are going to examine the object resulting from *union*. Due to the pushout construction, we have a sequence of gluing morphisms embedding component nets into the entire net. Gluing morphism thus allows us to follow one specific component net during its embedding into the composed net. So we can focus on this net and the morphisms leading to the entire net. Doing so, we profit from the concept of structuring, that is decomposition of a system into logically related subsystems, in so far as we can easily identify the logically relevant component net.

The following definitions 6, 7 and 8 prepare the notion of *structural change*, from a component of a pushout diagramm to the pushout object. We first define, what the structural change refers to.

Definition 6 (Subnet)
Let $N = (SPEC, P_N, T_N, pre_N, post_N, cond_N)$ be an AHL-net over a fixed specification. We call $M = (SPEC, P_M, T_M, pre_M, post_M, cond_M)$ a *subnet* of N, written $M \subseteq N$, if :

1. $P_M \subseteq P_N$
2. $T_M \subseteq T_N$
3. $pre_M(t) \leq pre_N(t)\, \forall t \in T_M$
4. $post_M(t) \leq post_N(t)\, \forall t \in T_M$
5. $cond_M = cond_N|_{T_M}$

A subnet with maximal $pre-$ and $post-$functions, i.e. satisfying the condition: $\forall p \in P_M, t \in T_M\ :\ a \cdot (term, p) \in pre_N(t) \Rightarrow a \cdot (term, p) \in pre_M(t)$, with $a \in \mathbb{N}$, is called a *full subnet*. □

Remark: A full subnet is uniquely determined by the sets P_M and T_M: pre_M and $post_M$ are the corresponding restrictions of pre_N and $post_N$ to the codomain T_M and domain $(T_{OP}(X) \times P_M)^{\oplus}$ and $cond_M = cond_N|_{T_M}$.
From a subnet $M \subseteq N$ there is not necessarily a morphism to N. Nevertheless, we have to use the above notion instead of inclusion, because inclusion is too coarse. The notion of subnet is needed in order to locate that part of a net, that is possibly changed by a *union* operation. The maximal subnet with that property is the environment (of a match). Therefore, the remaining subnet is minimal, meaning that at least that part is preserved.

Definition 7 (Environment)
Let $M \subseteq N$. The $environment(M)$ of M is the full subnet K $(M \subseteq K \subseteq N)$, induced by the sets:

- $P_K = P_M \cup \{p\ |\exists t \in T_M\ :\ (term, p) \in pre_N(t)$ or $(term, p) \in post_N(t)\}$ und
- $T_K = T_M \cup \{t\ |\exists p \in P_M\ :\ (term, p) \in pre_N(t)$ or $(term, p) \in post_N(t)\}$

Note, that the $environment(M)$ contains M itself. □

Definition 8 (Isomorphic Environment)
Let N, N', M, M' be nets and $f : N \rightarrow M$ a morphism. A subnet $N' \subseteq N$ has an isomorphic environment to $M' \subseteq M$ w.r.t. f if the restriction of f to the domain $environment(N')$ and codomain $environment(M')$ is an isomorphism.
There is *a (structural) change* of the subnet N' if $environment(N')$ is not isomorphic to $environment(M')$. □

We are now ready to locate the structural change from one net to the *union* net. More precisely, we will formulate conditions on the interface net and its match, that guarantee preservation of a subnet. Thus, without constructing the *union* net, we are able to specify parts of it, that already are identical in the starting net.

Lemma 9 (Change of Environment)
Let $I \xrightarrow{m} N$ be a match of a *union*. The gluing morphism $N \xrightarrow{g} M$ induces at most a change of the subnet $environment(m(I))$. On the remaining part $N \setminus environment(m(I))$ g is an inclusion.
Proof: Exactly the nodes of the image of the interface net $m(I)$ are glued in M (see remark to definition 3). Therefore, in the environment of the glued nodes $environment(m(I))$ there can be structural changes. Other nodes are not affected by the gluing. □

Lemma 10 (Gluing of Transitions)
Let $I \xrightarrow{m} N$ be a match of a *union*. If a transition of N is glued by the resulting gluing morphism $N \xrightarrow{g} M$, it has a pre-image in I and pre- and post-domain also are in I.
Proof: Follows by remark to definition 3 and the condition for morphisms (see assumption 5). □

This leads to the main theorem below. It states that the structure of a subnet is preserved by a sequence of gluing morphisms, if none of the interface nets matches any intermediate image. This implies, that all gluing morphisms are inclusions on that subnet.

Theorem 11 (Change by Sequence)
Let $N_0 \xrightarrow{g_1} N_1 \xrightarrow{g_2} \cdots \xrightarrow{g_m} N_m$ be a composition of gluing morphisms. A place $p \in N_0$ has an isomorphic environment as its image $g_m \circ g_{m-1} \circ \cdots \circ g_1(p)$ if all of its images inbetween are not in the matches of the interface nets.
Proof: Follows by induction over the number of applications of *union*-operation.

Induction base $(m = 1)$ **:** We have to show, that places that are not in the image of I_1 have an isomorphic environment in N_0 and in N_1.
With lemma 9 and 10 the environment of a place can only change, if it has a pre-image in I. This is an equivalent formulation.

Induction step $m - 1 \rightsquigarrow m$ **:** Let $N_0 \xrightarrow{g_1} N_1 \xrightarrow{g_2} \cdots \xrightarrow{g_m} N_m$ be a composition of gluing morphisms, $p \in N_0$ and all of its intermediate images are not in the images of the interface nets.
$p_{m-1} = g_{m-1} \circ g_{m-2} \circ \cdots \circ g_1(p)$ has the same

environment as $p_m = g_m \circ g_{m-1} \circ \cdots \circ g_1(p)$, if p_{m-1} is not in the image of I_m (see induction base). By induction hypothesis, p_{m-1} has the same environment as p and by transitivity p_m has the same environment as p.

$\square$

This theorem has the following impact: a structural property of the composed net is valid if it is valid in the corresponding component net and construction does not violate it. Thus, actual composition of the entire can be avoided.

4 VERIFYING SYSTEM PROPERTIES IN A MEDICAL INFORMATION MANAGEMENT SYSTEM

In this section we will present the case study and the verification of one requirement, that motivated this paper. The formal calculus for this verification is based on "assertions" (see e.g. [JV87]). An invariant over assertions then is valid if it is valid under the initial marking and each firing of any transition leads to a state in which it also holds.

The Case Study

The *Heterogeneous Distributed Information Management System* (HDMS), has been a large project including the whole reorganisation of the medical and management data of the German Cardiac Center Berlin, Deutsches Herz-Zentrum Berlin (DHZB), a clinical center which is dedicated to the treatment of all kinds of cardiac diseases. The kernel of HDMS, the folder of patients' data, is modelled by algebraic high-level nets in our case study [Erm96]. The kernel is an abstraction of user interfaces, programming environment, operating systems, data bases, integrated software-components, etc. For brevity, we will call this abstraction "HDMS" instead of "kernel of HDMS" in the following.

The following documents of HDMS are described by algebraic specifications in the sense of [EM85]:

- The *document for personal data* containing all patient-relevant information.
- The *temperature chart* (TC) contains the data that is taken daily by the nursing staff like blood pressure, temperature, pulse, medication, etc.
- The *laboratory report* includes various blood and urine analysis results.
- The *clinical finding* of a cardiac catheterization includes all data of the cardiac catheterization concerning radiography, blood circulation values etc.

These documents are involved in the subsequent routines in the DHZB :

- the *admittance of patients*, where the patients personal data are written down and the hospital treatment contract is signed;
- the *treatment*, where the patients vital values are measured daily and put down into the patients temperature chart;
- the *laboratory activities*, including various kinds of blood and urine analysis, the results being noted in the laboratory report;
- the *cardiac catheterization* (CC), the main surgical procedure of the DHZB, which will be explained later on.
- the *discharge of patients*, where the final diagnosis and the patient's discharge date and referral are noted on the patient's admittance card.

The entire algebraic Petri net of this case study is too large and complex to be represented as a unstructured net. Therefore, suitable horizontal structuring techniques (developed in [PER95]), namely *union* (denoted by the forking lines as shown in figure 2) and *fusion*, the gluing of component nets within a given net, are applied.

Note, that only the leaves of the tree in figure 2 and the interface nets for the *union* and *fusion* operations are explicitly modelled.

Example: The Cardiac Catheterization

As example we explain one of the most important routines at the DHZB, the cardiac catheterization of patients. The routine is modelled by the net CC-C, represented as a node in figure 2. It is composed of the nets CC-P and CC-E, represented by the leaves above it. In figure 3 we show one of its component, namely the net modelling the CC-Examination (CC-E).

Let us shortly explain the idea of the net CC-E in figure 3. The catheterization examination is a sequence of different examination steps (modelled by transitions) consisting of taking cardiac blood pressure and investigating flow velocity, stenoses or narrowings of vessels. The firing conditions of these transitions make sure that the measured values are written down in the correct protocol, belonging to the patient under examination. The involved people (doctor, patient) as well as the resources and machines (i.e. X-Ray, pressure meter, thermo medium, ...) employed during the examination are denoted as arc inscriptions. Arc inscriptions are terms to the underlying algebraic specification DHZB-STAY-Spec. For instance, the token $Patient$ is a variable of the sort $Patient$ which is specified as a tuple of a patient's personal data. The term $ch\_proto(Proto, X - ray)$ is an operation adding a new element $(X - ray)$ to the current protocol $Proto$. The doctor is present during the whole examination. After the examination the patient returns to the ward and the used machines and catheters are returned to their places.

Verification

In HDMS there are requirements concerning safety and security, the data, the processes, and others. Some of these are very general, for example that at any time each patient can only be once in the hospital or that identity numbers for patient and his prescribed treatment have to be the same. Other requirements directly refer to the treatment of patients in the hospital, for instance that advised remedies have to be compatible with each other or that the heart must not be catheterized without an intensive examination of the blood in advance.
In order to illustrate the use of our main theorem, we are going to verify the latter requirement. In the HDMS model it refers to two distinct component nets, Differential Blood Count and CC-Examination. Additionally, these component nets are situated in different subtrees of the structuring tree (see figure 2), so its verification has some complexity.

Requirement 1
"A cardiac catheter examination can only be carried out for a patient whose differential blood analysis has been completed."

The proof is organized as follows: First we formalize this requirement with respect to our HDMS model leading to formula (1). Generally, we formalize requirements using temporal logics (see e.g. [WVV+96]). For example, the expression "$\Box\varphi$" (read: φ is always valid) denotes that φ is an invariant of the system. Applying our main theorem, we can equivalently reformulate it, which leads to formula (2). Then, we investigate a part of the entire model. The result is used for the actual verification using the technique of assertions at the end of this section.

The **Requirement 1** refers to the process of differential blood examination in net Differential Blood Count (see figure 2) and that of a cardiac catheterization examination. The first process affects the *laboratory report* registering the differential blood examination of a patient. The relevant algebraic operation concerning the blood analysis is "$getdiff$", which references to the constant value no_Diff by default for all patients entering in the DHZB. The value no_Diff is overwritten in a patient's record document by a concrete differential blood analysis result when the analysis for this patient has been finished and cannot be set back to no_Diff any more. The process of a cardiac catheterization examination is modelled in the net CC-Examination (CC-E) in figure 3. The structure of this net is roughly a sequence. The requirement thus states that for any state between start and end of the examination, the patient's differential blood analysis must be completed. In Petri net terms this means that the marking of each place belonging to the sequence only may include $Patient$ tokens to which the operation registering the differential blood analysis result in the respective $PatientDocument$ has been applied before. This corresponds to an entry in the laboratory report, which can be checked by the operation $getdiff$.
Using terms from our net structure (place names, arc expressions, variables) and the algebraic specification (sorts, operations and equations), the requirement $\Box\varphi$ is formalized as follows:

$$\begin{aligned}
\Box(\ &(\textsf{Patient before CC-E}(Patient) \\
&\longrightarrow\ getdiff(Patient) \neq no_Diff) \wedge \\
&(\textsf{Patient under X-ray}(Patient) \\
&\longrightarrow\ getdiff(Patient) \neq no_Diff) \wedge \\
&(\textsf{Patient after press. taken}(Patient) \\
&\longrightarrow\ getdiff(Patient) \neq no_Diff) \wedge \\
&(\textsf{Patient after blood flow exam.}(Patient) \\
&\longrightarrow\ getdiff(Patient) \neq no_Diff) \wedge \\
&(\textsf{Patient after thermo injection}(Patient) \\
&\longrightarrow\ getdiff(Patient) \neq no_Diff)\quad) \qquad (1)
\end{aligned}$$

In φ, states of our system are evaluated. The premises $\textsf{Place}(Variable)$ mean: on the place Place there exist tokens being terms of the sort of $Variable$. The conclusion is an equation from the algebraic specification that must be satisfied for all these tokens.

The formula (1) can be equivalently reformulated for the following reason: Regarding the net CC-E in figure 3 locally, it seems clear that we have a sequence of transitions where each transition is activated only if the transition above (its predecessor) has been firing before. As none of the transitions change the entry referenced by $getdiff$, it is sufficient to state our requirement (1) only for the first place in the sequence Patient before CC-E, as all the following places will "inherit" the tokens of sort $Patient$ that have been verified already.
The problem arising now, is that we have not constructed the entire system net, so we do not know how the net CC-E is embedded into the entire model. It might well be that the sequential structure is destroyed by some composition operation. See, for instance, the example in figure 1.
Therefore, we have to show that in the entire model none of the places of { Patient under X-ray, Patient after press. taken, Patient after blood flow exam., Patient after thermo injection } lies in the postdomain of some transition that is not already depicted in net CC-E.
For the formal proof we need theorem 11, which can be specialized in the case of HDMS, because all morphisms are inclusions. Thus, for a sequence of

structuring steps $N_0 \xrightarrow{g_1} N_1 \xrightarrow{g_2} \cdots \xrightarrow{g_m} N_m$ a place $p \in N_0$ has an isomporphic environment as in N_m if it is not contained in any of the interface nets.

Lemma 12 (Starting a CC Examination)
Every CC examination (CC-E) begins only with a marking on the place Patient before CC-E.
Proof : According to theorem 11 we only have to consider the *union* operations composing the net CC-E with other component nets. As we can see in figure 2 we have to search the interface nets of four *union* operations for one of the places of {Patient under X-ray, Patient after press. taken, Patient after blood flow exam., Patient after thermo injection}. None of the interface nets contains these places, and by the above specialization of theorem 11 we conclude that the sequential structure is preserved in the entire net. Under the assumption that the initial marking does not mark any intermediate place of { Patient under X-ray, Patient after press. taken, Patient after blood flow exam., Patient after thermo injection } we have the supposition. □

We now can modify the formalization of **Requirement 1** in formula (1), leading to the eqivalent, but less complex formula:

$$\Box(\ \text{Patient before CC-E}(Patient) \longrightarrow getdiff(Patient) \neq no\_Diff \ ) \quad (2)$$

For the actual verification of formula (2) we now examine how the entry place Patient before CC-E is marked. Again we consider the composition operations concerning the net CC-E.

Lemma 13 (Transitions starting a CC Exam)

The place Patient before CC-E is only marked by a firing of the transition $\boxed{\text{initialize CC protocol}}$ in component net CC-P.
Proof : We find that the place Patient before CC-E is contained in one interface net for the composition of the nets CC-P and CC-E resulting in net CC-C as shown in figure 4. In the net CC-C the place Patient before CC-E lies in the postdomain of the transition $\boxed{\text{initialize CC protocol}}$. According to the specialization of theorem 11 we now have to search the three remaining interface nets used for the construction of the entire net. None of them contains the place Patient before CC-E, so that its environment in the entire net is the same as in the net (CC-C). □

The lemmata 12 and 13 now can be used for a formal verification using the technique of assertions[1] of the requirement 1, that is for the proof of the following theorem:

Theorem 14 (HDMS meets Requirement 1)
The composed net Actual State of a Patients Stay at the DHZB (the root of the net structure tree in figure 2) satisfies requirement 1.

Proof:
Adding a formal definition of the initial marking to HDMS the verification can be done in two steps, where $\Box\varphi$ is formula (2):

1. $M_0 \models \varphi$
   M_0 is the initial marking, and $M_0 \models \varphi$ means that φ is valid for the initial marking,
2. $\forall t \in T : M_i \models \varphi \longrightarrow M_{i+1} \models \varphi$
   with $M_i[\mathrm{t} > M_{i+1}$
   meaning that the firing of any transition keeps φ valid.

The validity of φ for M_0 can be deduced directly by the initial marking (in our case: no tokens of sort *Patient* in the net CC-E but enough tokens denoting doctors and resources to make a CC examination possible when a patient is referred to it). The proof of the induction steps in 2. is based on the result of lemma 13 that in the composed net the place Patient before CC-E is only in the postdomain of the transition $\boxed{\text{initialize CC protocol}}$ and not in the postdomain of any other transition. So this is the only transition in the entire net that could possibly render φ invalid, as we have no inverse operation for changing a valid *Diff*-entry into *noDiff*.
Therefore, an equivalent formulation of the induction steps in 2. is the following:

$$M_i \models \varphi \longrightarrow M_{i+1} \models \varphi \quad \text{with} \quad M_i[\boxed{\text{initialize CC protocol}} > M_{i+1}$$

We consider the firing condition of $\boxed{\text{initialize CC protocol}}$ in figure 4 and find that one of the equations requires $getdiff(Patient) \neq no\_Diff$. The proof therefore is completed here as the equation in the relevant transition's firing condition equals the requirement formalized in (2). That means, the transition can fire only if our requirement is satisfied for all tokens of type *Patient* in $\boxed{\text{initialize CC protocol}}$'s predomain. □

[1] For the mathematical foundations of algebraic net verification techniques see for example [CM88, WVV+96, KRVW96, Rei97].

5 CONCLUSION

In this paper, we have examined the relationship between structuring techniques and other Petri net techniques. For algebraic high-level nets we have shown that structuring and verification techniques, in particular, can be combined by extracting the relevant information for verification from the component nets and their composition. Our main theorem states under which assumptions the structure of a component net is preserved in the composed net. So, we can profit both from the concept of structuring as from verification techniques defined for algebraic high-level nets. We can identify the relevant component net a requirement refers to. Applying the main theorem verification of the entire net is possible without composing all component nets. Especially, for large systems, which often cannot be constructed for complexity reasons, this method faciliates verification. In this sense, we consider this work as a first step towards compatibility of structuring and verification.

But this approach is not restricted to the relation of structuring and verification. Other Petri net techniques like analysis, simulation, etc. are apt to profit from it, as well. In fact, it is useful for all techniques, that operate on the entire net. This will be subject to future work.

Furthermore we are confident that this approach is not restricted to only AHL-nets, but can be transferred to other Petri net classes that are tractable in a categorical setting. Provided that these categories have pushouts it is very likely that this method can be adopted.

Finally, the relation of verification and vertical structuring will be an intersting topic of our future work.

References

[CM88] K. M. Chandy and J. Misra, *Parallel program design: A foundation*, Addison-Wesley, 1988.

[EL93] H. Ehrig and M. Löwe, *The ESPRIT BRWG COMPUGRAPH Computing by Graph Transformations : A survey*, TCS 109, North-Holland, 1993, pp. 3 – 6.

[EM85] H. Ehrig and B. Mahr, *Fundamentals of algebraic specification 1: Equations and initial semantics*, EATCS Monographs on Theoretical Computer Science, vol. 6, Springer, Berlin, 1985.

[EPR94] H. Ehrig, J. Padberg, and G. Rozenberg, *Behaviour and realization construction for petri nets based on free monoid and power set graphs*, Tech. report, Technical University Berlin TR 94-15, 1994.

[Erm96] C. Ermel, *Anforderungsanalyse eines medizinischen Informationssystems mit Algebraischen High-Level-Netzen*, Tech. Report 96-15, TU Berlin, 1996.

[Feh92] R. Fehling, *Hierarchische Petri Netze*, Dr. Kovač, 1992.

[Jen92] Kurt Jensen, *Coloured Petri nets. basic concepts, analysis methods and practical use*, vol. 1, Springer, 1992.

[JV87] E. Jessen and R. Valk, *Rechnersysteme - Grundlagen der Modellbildung*, Studienreihe Informatik, Springer Verlag, Berlin, 1987.

[Kin95] Ekkart Kindler, *Modularer entwurf verteilter systeme mit Petrinetzen*, Ph.D. thesis, Technische Universität M"unchen, Institut für Informatik, 1995, Erscheint demn"achst in: edition VERSAL, W. Reisig (Hrsg.), Dieter Bertz Verlag, Berlin.

[KRVW96] Ekkart Kindler, Wolfgang Reisig, Hagen Völzer, and Rolf Walter, *Petri net based verification of distributed algorithms: An example*, Informatik-Berichte 63, Humboldt-Universität zu Berlin, May 1996, Submitted.

[KS91] B. Krämer and H. W. Schmidt, *Types and modules for net specifications*, High-Level Petri Nets: Theory and Application (K. Jensen and G. Rozeberg, eds.), Springer, 1991, pp. 171–188.

[Lil94] J. Lilius, *On the structure of high-level nets*, Ph.D. thesis, Helsinki University of Technology, 1994.

[MM90] J. Meseguer and U. Montanari, *Petri nets are monoids*, Information and Computation **88** (1990), no. 2, 105–155.

[PER95] J. Padberg, H. Ehrig, and L. Ribeiro, *Algebraic high-level net transformation systems*, Mathematical Structures in Computer Science **5** (1995), 217–256.

[Rei85] W. Reisig, *Petri nets*, EATCS Monographs on Theoretical Computer Science, vol. 4, Springer-Verlag, 1985.

[Rei91] W. Reisig, *Petri nets and abstract data types*, Theoretical Computer Science **80** (1991), 1 – 34 (fundamental studies).

[Rei97] Wolfgang Reisig, *Elements of distributed algorithms: Modelling and analysis with Petri nets*, In preparation, 1997.

[Vau86] J. Vautherin, *Parallel specification with coloured Petri nets and algebraic data types*, Proc. of the 7th European Workshop on Application and Theory of Petri nets (Oxford, England), jul. 1986, pp. 5–23.

[WVV+96] R. Walter, H. Völzer, T. Vesper, W. Reisig, E. Kindler, and J. Freiheit, J.and Desel, *Memorandum: Petrinetzmodelle zur Verifikation Verteilter Algorithmen*, Informatik-Bericht 67, Humboldt-Universität zu Berlin, July 1996.

6 FIGURES

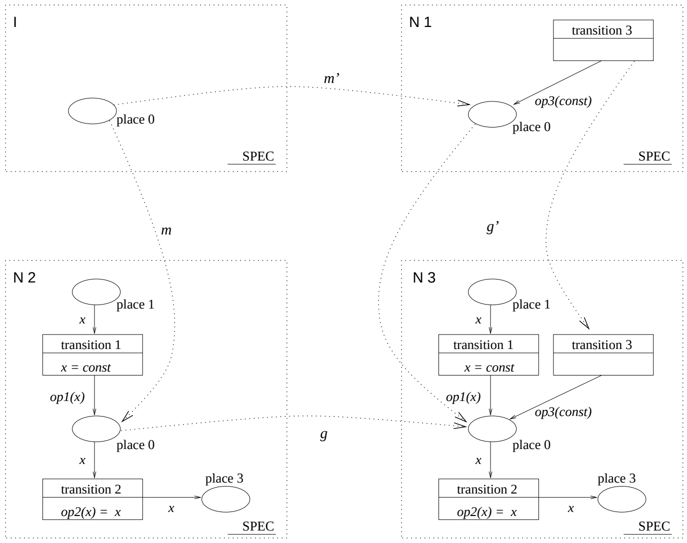

Figure 1: Union of two Nets over an Interface Net

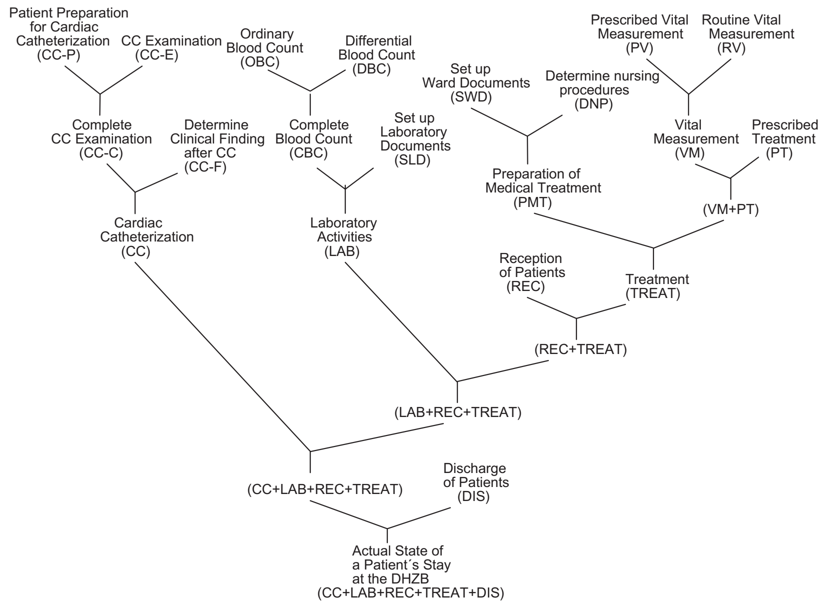

Figure 2: Horizontal Structuring of the HDMS-Model

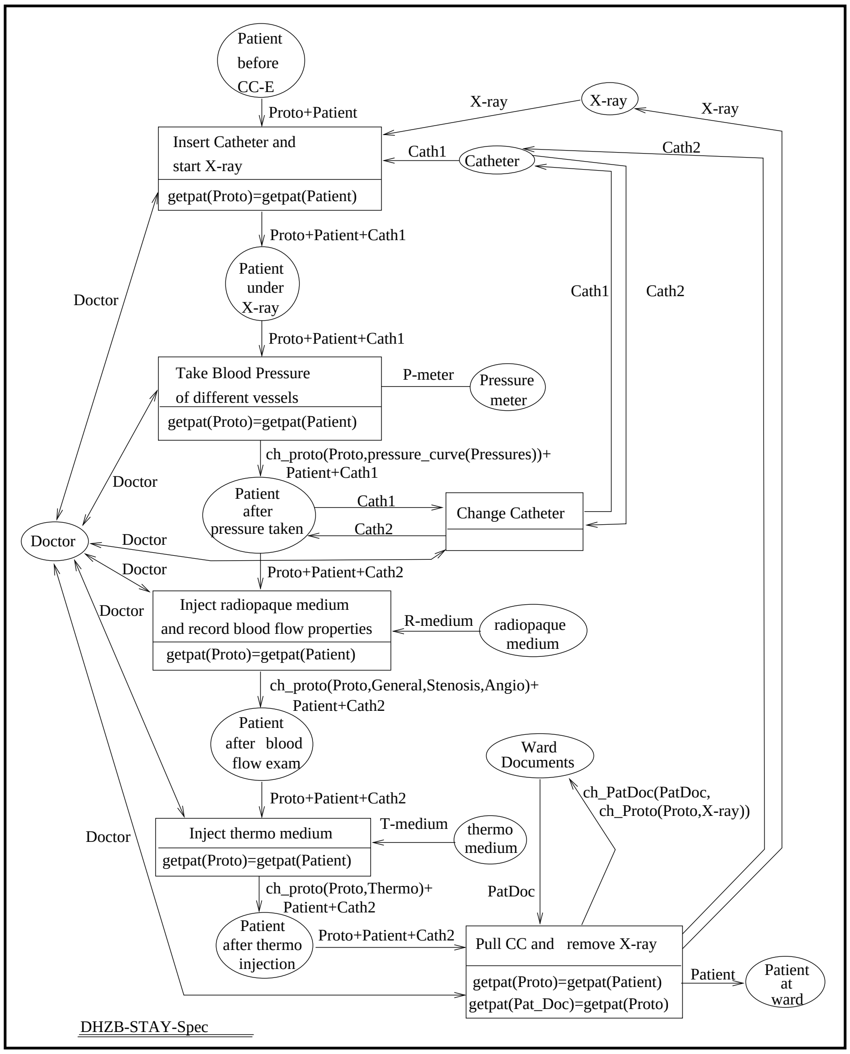

Figure 3: Cardiac Catheterization Examination (CC-E)

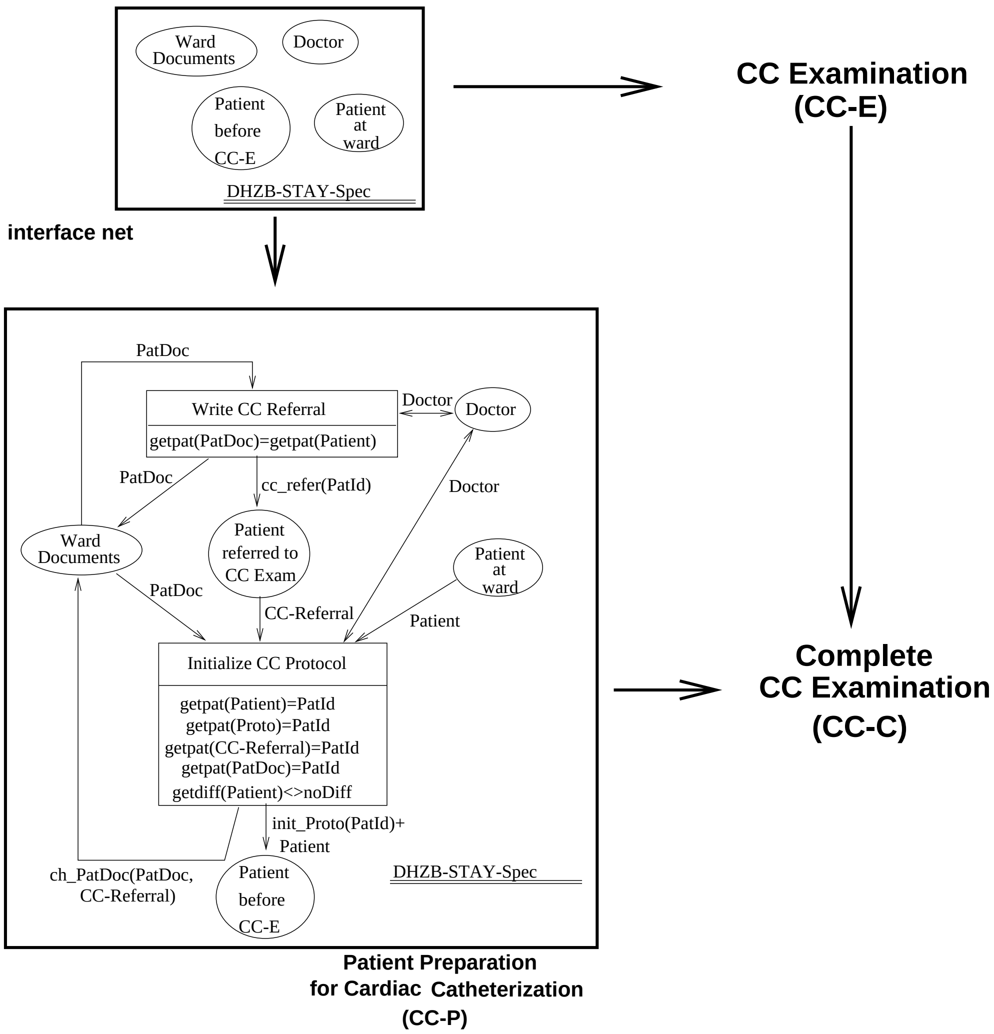

Figure 4: Composition of two Subnets via a common Interface Net