Stepwise Introduction and Preservation of Safety Properties in Algebraic High-Level Net Systems

J. Padberg, K. Hoffmann, and M. Gajewsky

Technical University Berlin
Institute for Communication and Software Technology
{padberg, hoffmann, gajewsky}@cs.tu-berlin.de

Abstract. Our approach of rule-based refinement[1] provides a formal description for the stepwise system development based on Petri nets. Rules with a left-hand and a right-hand side allow replacing subnets in a given algebraic high-level net system. The extension of these rules supports the preservation of system properties. In this paper we extend the preservation of safety properties significantly. We define rules, that introduce new safety properties. In our new approach we propose first the verification of properties at the moment they can be expressed and then their preservation further on. Hence, properties can be checked as long as the system is still small. Moreover, introducing properties allows checking these for the relevant subpart only. Changes that are required later on can be treated the same way and hence preserve the system properties. Hence, we have made a step towards a formal technique for the stepwise system development during analysis and design.

Keywords: algebraic high-level net systems, rule-based refinement, temporal logic, safety properties, safety preserving morphisms, safety introducing rules, safety preserving rules

1 Introduction

The need to ensure quality of software systems yields a demand for formal techniques for the assertion of system properties. The verification of a (sub)system usually takes place, after the system has been modeled. This is on the one hand costly, since it is a large and complex system. On the other hand subsequent changes require the repeated verification of the whole system once more. The approach we are suggesting solves this problem. We propose to introduce safety properties already during the stepwise development of the system model. Our approach is based on the rule-based refinement of Petri nets.

Petri nets have a long and successful story. The integration with data type descriptions has led in the 80's to powerful specification techniques (a summary can be found in [JR91]). These techniques together with an adequate

[1] This paper continues our research on rule-based refinement of algebraic high-level nets, we have first introduced at FASE 98 [PGE98].

T. Maibaum (Ed.): FASE 2000, LNCS 1783, pp. 249–265, 2000.

tool support, e.g. [JCHH91, Jen92, Gru91, DG98, OSS94] are widely applied in practical system development. Here, we use algebraic high-level net systems [Vau87, PER95, GPH00], an integration of place/transition nets (as in [Rei85] or algebraically in [MM90]) with algebraic specifications in the sense of [EM85]. Rule-based refinement of nets is obtained by the instantiation of high-level replacement systems [EHKP91] by algebraic high-level net systems. Rules are given as a span of morphisms, mapping the interface to the left-hand as well as to the right-hand side. The application of a rule to a net system is a transformation and yields again a net system. This target net system is roughly the original one, where one subnet is replaced by another. Refinement rules, moreover have an additional morphism from the left-hand side to the right-hand side. This morphism then ensures the preservation of system properties in the sense of [MP92].

In this paper safety properties are expressed in terms of certain temporal formulas over the markings of the net system. We introduce several morphisms of algebraic high-level net systems. Namely place preserving and transition gluing morphisms preserve safety properties. That is, if the safety properties hold in the domain of a morphisms then it holds in the codomain as well. We then present safety preserving rules yielding safety preserving transformations. Moreover, we present safety introducing rules. These add a new subnet together with a safety property over this subnet. Hence, new safety properties are introduced into the model and then preserved. The main result of this paper is that the introduced morphisms give rise to safety preserving rules and transformations in Theorems 1 and 2. Furthermore, rules for the introduction of new safety properties preserve the old safety properties and add new ones (see Theorem 3). Our approach allows the explicit introduction of safety properties as soon as they are expressible in the model. So once the corresponding system part is modeled the relevant safety properties can be expressed and verified. Hence, the verification takes place as soon as possible and more important as long as the model is fairly small. Moreover, we have the possibility to verify the safety properties for the relevant part only. Subsequently they are preserved throughout the remaining development process. In Concept 11 we describe this impact of our technical results.

The first step to a rule-based refinement preserving safety properties has been presented for algebraic high-level net systems at FASE 98 [PGE98]. We here give the new class of transition gluing morphisms. We show that these morphisms preserve safety properties and allow safety preserving rules (see Theorem 2). Moreover, our approach now comprises nets with an initial marking, i.e. net systems. Hence, we expand the notion of place preserving morphisms and ensure the desired properties (see Theorem 1). And we extend our approach to allow rules that explicitly introduce new safety properties. Details concerning the proofs and the lemmas in the appendix can be found in [GPH00].

This paper is organized as follows: First in Section 2 we illustrate our notions by an example. Then (Section 3) we give the formal definitions and state our results. At last we summarize our results and discuss future work.

2 Example: Medical Information System

In this section we motivate the notions and results of this paper in terms of a small example inspired by a case study [Erm96] of the medical information system called Heterogeneous Distributed Information Management System (HDMS). On the one hand the aim of this section is to introduce the basic notions of the model (algebraic high-level net system), temporal logic formulas, and rule-based refinement on an intuitive level. On the other hand we demonstrate how safety properties are preserved using place preserving and transition gluing morphisms. These safety properties have to be proven only once for the rule introducing them. Our notions concerning safety introducing and preserving rules capture the intuitive understanding of such development steps, that cannot violate given safety properties. In Section 3 these notions are given formally.

An algebraic high-level net system consists — roughly speaking — of a Petri net with inscriptions of an algebraic specification defining the data type part of the net system and an initial marking. The tokens are data elements of a *Spec*-Algebra [EM85]. Figure 1 shows a sample net system, the subsystem blood count `BC`. The net system is inscribed with terms over the specification `BC-Spec` sketched below. The idea of the net system `BC` is to model the following situation at the hospital: The blood test of a patient arrives at the laboratory, denoted by the transition *Taking blood pressure*. Then the blood count is carried out, if demanded. Finally, the doctor is notified about the measured values. All these activities are represented as transitions in the net system `BC`. Furthermore we give the initial marking of `BC`.

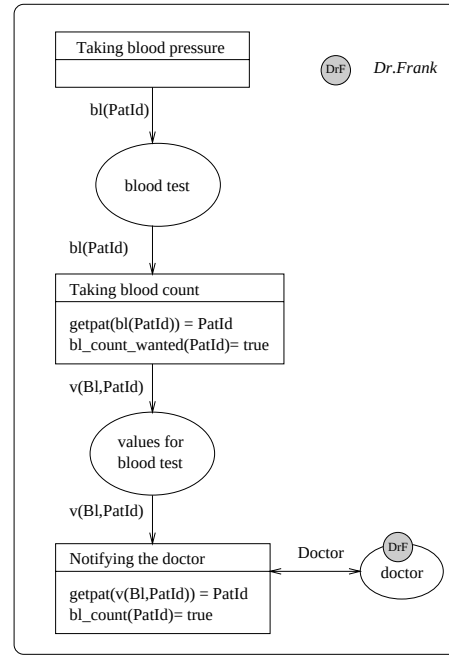


Figure 1: Start Net System `BC`

We merely state the sorts and operations of `BC-Spec` used explicitly in the subsequent initial markings.

sorts: Name, Patient, PatId, Pat Record, Blood Test, ...

opns : patient: Name, Sex, Adress, PatId → Patient
getpat: Patient → PatId
mk_patrecord: Patient, Presc → PatRecord
bl: PatId → Blood Test

We here consider the A-quotient term algebra (see [EM85]), the algebra generated according to the specification over carrier sets for names, doctors,

resources etc. Assuming the carrier sets $A_{Name} = \{Smith, Miller, \dots\}$ and $A_{Doctor} = \{Dr.\ Frank, Dr.\ Brown \dots\}$ we have the following initial marking $M_{\texttt{BC}}$ given by $(Dr.\ Frank, \mathbf{doctor})$. Dr. Frank is at the ward, which is represented by the token on the place **doctor**.

We are now going to enhance the model with the blood value measurement of a patient at the ward. We achieve this by first adding patients and their ward documents and the taking of blood pressure and blood test to the start net system `BC`, i. e. the application of the rule r_{int}, depicted in Figure 2.

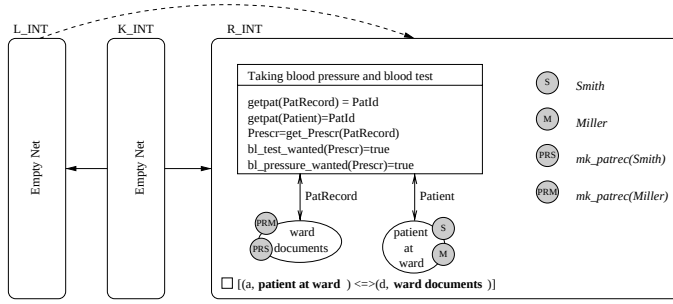


Figure 2: Safety Introducing Rule r_{int}

In general, rules are split into a deleting part L, an adding part R and an interface K which is preserved. The rule r is then given by $r = (L \xleftarrow{u} K \xrightarrow{v} R)$ where u and v are specific morphisms. The application of a rule to an algebraic high-level net system informally means replacing a subnet specified by the left-hand side of the rule with a net system specified by the right-hand side. More precisely, the rule r is applied to a net system G via an occurrence morphism $g : L \to G$. Then the deletion step and the construction step is defined using two pushouts (for formal definition see Appendix A). In our example the application of the rule r_{int} to the net system `BC` yields the net system `BVM1`[2], which consists of the right-hand side of the rule r_{int} and the start net system `BC`. The initial marking of the net system on the right-hand side of r_{int} is given by patients and their patient records, that is we have

$$(Smith, \mathbf{patient\ at\ ward}) \oplus (mk\_patrec(Smith, \dots), \mathbf{ward\ documents}) \oplus$$
$$(Miller, \mathbf{patient\ at\ ward}) \oplus (mk\_patrec(Miller, \dots) \mathbf{ward\ documents})$$

The net system `R_INT` on the right-hand side of the rule r_{int} in Figure 2 is provided with an additional safety property. Such rules can be shown to be safety introducing (see Definition 9 and Theorem 3). Safety introducing rules are provided with additional safety properties for the right-hand side net system. This safety property is introduced to the resulting net system for each application of the rule. For the correct treatment of a patient we have to ensure the following: "whenever some patient is at the ward the corresponding patient record is within

[2] Due to space limitation we do not give the intermediate net systems `BVM1`, `BVM2`, or `BVM3` explicitely here.

the ward documents". This safety property is not relevant before application of r_{int} and even not expressible. After application of r_{int} however it becomes crucial. First of all it has to be true on the level of rules, that is we have to verify it for the net system R_INT. The safety property obviously holds in R_INT. As this kind of rule preserves safety properties (see Theorem 3), it also holds in the net system BVM1 resulting of the application of r_{int} to BC. In Definition 6 we define temporal logic formula over nets systems and their validity with respect to markings. Intuitively, a temporal logic formula states facts about markings and is given in terms of data elements of tokens on places. That is, the static formula $(a, \textbf{patient at ward}) \Longleftrightarrow (d, \textbf{ward documents})$ is true for a marking M where a patient a is at the ward if and only if the corresponding patient record d is at the ward. This is given formally by $getpat(a) = getpat(d)$ for $a \in A_{Patient}$ and $d \in A_{PatRecord}$. The always operator in the safety property $\Box[(a, \textbf{patient at ward}) \Longleftrightarrow (d, \textbf{ward documents})]$ states that this is true for all reachable markings.

In Figure 3 we give the rule r_{tg-tbp} that identifies the transitions *Taking blood pressure* and *Taking blood pressure and blood test*. Applying r_{tg-tbp} to the net system BVM1 yields the net system BVM2, that contains no longer isolated elements. As the morphism from L_TG-TBP to R_TG-TBP is transition gluing the safety property is preserved. Moreover the safety property holds in BVM2 as tg-rules preserve safety properties (see Theorem 2).

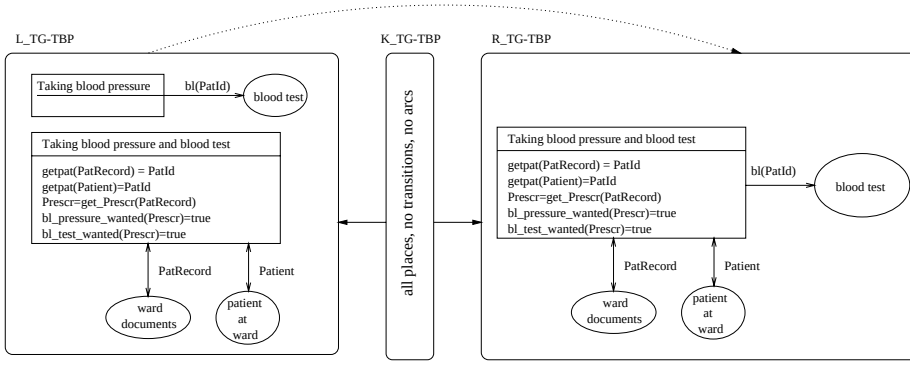


Figure 3: Transition Gluing Rule r_{tg-tbp}

In order to introduce the planning of the next measurement we apply rule r_{pp} depicted in Figure 4 to the net system BVM2. This yields the net system BVM3, which consists of the net system BVM2 extended by the net system for planning of the next measurement. Rule r_{pp} describes the insertion of the place **values for hypertension test** and the subsequent transition and the place **schedule**. This describes that the values of the hypertension test influence the schedule of the next measurement. As the rule r_{pp} does not change the environment of places, it is called *place preserving* (see Definition 4). Applying this rule we again preserve the safety property. Especially, the introduced safety property is

propagated to the resulting net system. The preservation of safety properties by place preserving rules is stated in Theorem 1.

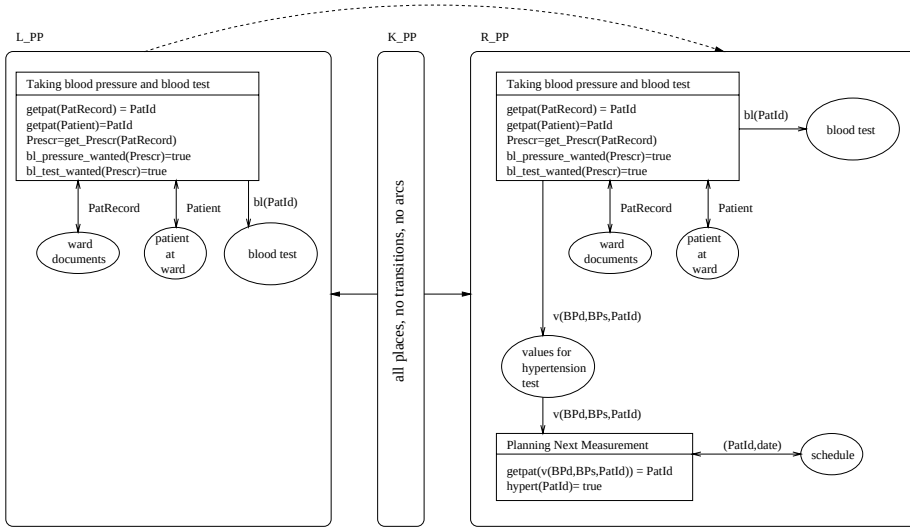


Figure 4: Place Preserving Rule r_{pp}

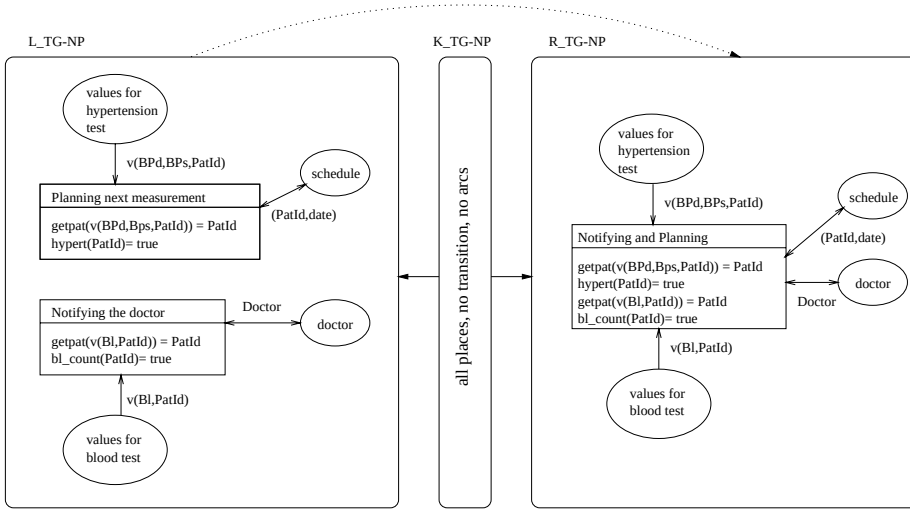


Figure 5: Transition Gluing Rule r_{tg-np}

Finally we model that notifying the doctor and planning of the next measurement are done synchronously. Both values for hypertension test and for blood test have to be available. This is achieved by the rule r_{tg-np} in Figure 5 that glues

the corresponding transitions. Since it is transition gluing it is compatible with the safety property (as stated in Theorem 2). This means, that in the derived net system BVM in Figure 6 still the safety property $\Box[(a, \textbf{patient at ward}) \Longleftrightarrow (d, \textbf{ward documents})]$ holds.

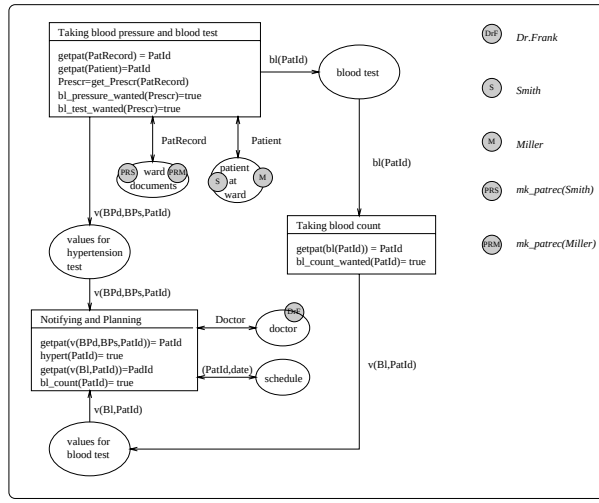


Figure 6: Final Net System BVM

The main result of this paper is that we can introduce new safety properties, which have to be proven for the right-hand side of the safety introducing rules only. They are preserved by place preserving and transition gluing rules. In this way we can extend our example net system BVM by adding further safety properties. The given rules satisfy sufficient conditions to ensure the propagation of safety properties. This is a great advantage for software development as safety properties can be proven in a subsystem of the whole model.

3 Safety Introducing and Preserving Rules

In this section we present our important conceptual result concerning a method for handling safety properties during software development. This result is supported by our main technical results stating the introduction and preservation of safety properties by special kinds of rules (For a review of rules and transformations see Appendix A). We formally define algebraic high-level (AHL) net systems given on an intuitive level in Section 2. For AHL net systems we introduce different notions of morphisms needed for safety preservation and their integration with rules. Based on the notion of safety property we also define safety preserving morphisms and rules. In our main theorems we show that the previously defined rules are in fact safety preserving.

A detailed version of our notions, results and proofs can be found in [GPH00], but would definitely exceed the scope of the paper.

Definition 1 (Algebraic High-Level Net System). *An algebraic high-level (AHL) net system is given by*

$$N = (SPEC, A, P, T, pre, post, cond, \widehat{m})$$

with

- $SPEC \in |\mathbf{SPEC}|$ *: an algebraic specification with* $SPEC = (\Sigma, E)$ *(see [EM85]),*
- $A \in |\mathbf{Alg}(\mathbf{SPEC})|$ *a* $SPEC$ *algebra,*
- P *: the set of places,*
- T *: the set of transitions,*
- $pre, post : T \to (T_{OP}(X) \times P)^{\oplus}$
  the pre- and postcondition functions of T*, defining for each transition with adjacent arcs the arc inscriptions and the weight, (see Def. 15 for* $\oplus$*-operator)*
- $cond : T \to \mathcal{P}_{fin}(EQNS(\Sigma))$
  the function that maps each transition to a finite set of conditions over the signature with variables representing the firing conditions, and
- $\widehat{m} \in (A \times P)^{\oplus}$ *the initial marking (see Def. 15).*

In contrast to [PGE98] AHL net systems comprise an initial marking. Examples of AHL net systems are depicted in the figures in Section 2. AHL net systems are provided with distinguishable tokens. These are denoted by tuples (a, p), where $a \in A$ is a data element and $p \in P$ is the place it resides on. Tokens are moved over the set of places by firing of transitions. This firing behaviour is given by adding and subtracting of tokens to and from a marking. This is achieved using the corresponding monoid operations (see Def. 16 in Appendix B), the tokens and a consistent assignment of variables.

Subsequently, we introduce various notions of morphisms for AHL net systems. These notions are based on the restriction of the initial marking, given in Definition 18 in Appendix B, and the pre- and post set, see Definition 17. All morphisms are specializations of so called loose morphisms. These give rise to the category **QAHLSys** which is of fundamental importance in the proofs of our main theorems.

Definition 2 (AHL Net System Morphisms and Categories). *Given two AHL net systems* $N_i = (SPEC_i, A_i, P_i, T_i, pre_i, post_i, cond_i, \widehat{m_i})$ *for* $i = 1, 2$ *then an algebraic high-level (AHL) net system morphisms* $f : N_1 \to N_2$ *given by* $f = (f_\Sigma, f_A, f_P, f_T)$ *with*

- $f_\Sigma : SPEC_1 \to SPEC_2$ *is a specification morphism,*
- $f_A : A_1 \to V_{f_\Sigma}(A_2)$ *is a homomorphism in* $\mathbf{Alg}(\mathbf{SPEC_1})$*,*
- $f_P : P_1 \to P_2$ *maps places to places in* **Set***, and*
- $f_T : T_1 \to T_2$ *maps transitions to transitions in* **Set***,*

and the following abbreviations

- $f_{insc} := (f_\Sigma \times f_P)^{\oplus} : (T_{OP1}(X) \times P_1)^{\oplus} \to (T_{OP2}(X) \times P_2)^{\oplus}$
- $f_M := ((f_\Sigma, f_A) \times f_P)^{\oplus} : (A_1 \times P_1)^{\oplus} \to (A_2 \times P_2)^{\oplus}$
- $f_C := \mathcal{P}_{fin}(f_\Sigma^{\#})$

for the induced mappings of arc inscriptions, markings, respectively conditions, are called

loose

- *for any* $t \in T_1$ *we have* $f_{insc}(pre_1(t)) \leq pre_2(f_T(t))$
  $f_{insc}(post_1(t)) \leq post_2(f_T(t))$ no "old" arcs are lost
- $f_C \circ cond_1 \subseteq cond_2 \circ f_T$ possibly adding transition guards
- $f_M(\widehat{m_{1|p}}) \leq \widehat{m_{2|f_P(p)}}$ placewise lesser initial marking of N_1
- $A_1 \cong V_{f_\Sigma}(A_2)$ compatibility of models as f_A is an isomorphism

This class of morphisms gives rise to the category **QAHLSys**.

transition preserving

if they are loose and

- $f_{insc} \circ pre_1 = pre_2 \circ f_T$
  $f_{insc} \circ post_1 = post_2 \circ f_T$
  no "new" arcs adjacent to "old" transitions are allowed
- $f_C \circ cond_1 = cond_2 \circ f_T$ compatibility of transition guards

This class of morphisms gives rise to the category **AHLSys**.

place preserving

if it is a loose morphism and the following place preserving conditions hold:

- $f_\Sigma : SPEC_1 \to SPEC_2$ *is persistent, that is for the corresponding free functor* $F_{f_\Sigma} : \mathbf{Alg(SPEC_1)} \to \mathbf{Alg(SPEC_2)}$ *we have* $V_{f_\Sigma} \circ F_{f_\Sigma} \cong ID$ *(see [EM85]).*
- $\bullet(f_P(p)) = (f_\Sigma \times f_T)^{\oplus}(\bullet p)$
  and $(f_P(p))\bullet = (f_\Sigma \times f_T)^{\oplus}(p\bullet)$ *for all* $p \in P_1$
  (see Def. 17 for pre and post sets)
  no "new" arcs adjacent to "old" places are allowed
- f_T, f_P, *and* f_Σ *are injective*
- $\widehat{m_{2|f_P}} = f_M(\widehat{m_1})$ no "new" tokens on "old" places are allowed
- $f_C \circ cond_1 = cond_2 \circ f_T$ compatibility of transition guards

transition gluing

if it is a loose morphism and the following holds:

- f_P *and* f_Σ *are isomorphisms*
- $f_M(\widehat{m_1}) = \widehat{m_2}$ no "new" tokens
- f_T *is surjective s. t. for* $t_2 \in T_2$
  $pre_2(t_2) = \sum_{t \in f_T^{-1}(t_2)} f_{insc}(pre_1(t))$, *and analogously for post,*
  summing up all arcs from the source transitions
- $cond_2(t_2) = \bigcup_{t \in f_T^{-1}(t_2)} f_C \circ cond_1(t)$ *for* $t_2 \in T_2$
  summing up all conditions from the source transitions

Note, that these notions are novel and especially designed for safety preservation. In fact, place preserving and transition gluing morphisms preserve safety properties, because basically the environments of places are preserved. Moreover, the treatment of the initial marking, namely placewise comparison, yields cocompleteness.

Example 1 (Morphisms in Example). Transition preserving morphisms are depicted in Figures 2,3, 4, and 5 by $\rightarrow$ as there are no transitions in the interfaces of the rules. The morphisms in Figure 2 and 4 denoted by $\twoheadrightarrow$ are place preserving, because new arcs are adjacent only to new places. Figures 3 and 5 depict transition gluing morphisms by $\dashrightarrow$.

For the integration of place preserving and transition gluing morphisms with rules in Definition 4 and 5 we use strict morphisms yielding an HLR system, see [EHKP91, EGP99].

Definition 3 (Strict Morphisms). *Given a transition preserving AHL net system morphism* $f : N_1 \to N_2$ *with* $f = (f_\Sigma, f_A, f_P, f_T)$ *and* $N_i = (SPEC_i, A_i, P_i, T_i, pre_i, post_i, cond_i, \widehat{m_i})$ *for* $i = 1, 2$. *f is called*

1. *marking strict*
   if $f_M(\widehat{m_1}_{|p}) = \widehat{m_2}_{|f_P(p)}$ *the initial marking of the source net is placewise equal to the target net.*
2. *strict*
   if it is marking strict, and moreover f_P, f_T, *and* f_Σ *are injective, and* f_Σ *is strict in* **SPEC**.

Definition 4 (PP-Rules). *A pair* (r, f) *is called pp-rule, if* $r = (L \xleftarrow{u} K \xrightarrow{v} R)$ *is a rule with strict transition preserving morphisms* u, v *and a place preserving morphism* $f : L \to R$ *with* $f \circ u = v$.

Definition 5 (TG-Rules). *A pair* (r, f) *is called tg-rule, if* $r = (L \xleftarrow{u} K \xrightarrow{v} R)$ *is a rule with strict transition preserving morphisms* u, v *and a transition gluing morphism* $f : L \to R$ *with* $f \circ u = v$.

Example 2 (Rules in Example). According to Example 1, the rules in Figure 2 and 4 are pp-rules. Figures 3 and 5 denote tg-rules.

The next definition concerns safety properties based on formulas, see also [PGE98], and their translation via (arbitrary) morphisms. Subsequently, Definition 7 specifies safety preserving morphisms.

Definition 6 (Safety Property, Formulas, Translations).

1. *The set of static formulas* $\mathcal{F}$ *is given inductively: For* $\lambda(a, p) \in (A \times P)^\oplus$ *we have the atoms* $\lambda(a, p) \in \mathcal{F}$. *Furthermore* $(\varphi_1 \in \mathcal{F} \Longrightarrow \neg\varphi_1 \in \mathcal{F})$, *and* $(\varphi_1 \in \mathcal{F}, \varphi_2 \in \mathcal{F} \Longrightarrow \varphi_1 \wedge \varphi_2 \in \mathcal{F})$.
   The validity of formulas is given w. r. t. the marking of a net. Let $m \in (A \times P)^\oplus$ *be a marking of N then:* $m \models_N \lambda(a, p)$ *iff* $\lambda(a, p) \leq m$, *and* $m \models_N \neg\varphi_1$ *iff* $\neg(m \models_N \varphi_1)$, *and* $m \models_N \varphi_1 \wedge \varphi_2$ *iff* $(m \models_N \varphi_1) \wedge (m \models_N \varphi_2)$.
2. *Let* φ *be a static formula over N. Then* $\Box\varphi$ *is a* **safety property**. *The safety property* $\Box\varphi$ *holds in N under m iff* φ *holds in all states reachable from m:*
   $$m \models_N \Box\varphi \Longleftrightarrow \forall m' \in \lceil m \rangle : m' \models_N \varphi$$
   For the initial marking $\widehat{m}$ *we also write* $N \models \Box\varphi$ *instead of* $\widehat{m} \models_N \Box\varphi$.
3. *The* **translation** $\mathcal{T}_f$ *of formulas over* N_1 *along a morphism* $f = (f_\Sigma, f_A, f_P, f_T) : N_1 \to N_2$ *to formulas over* N_2 *is given for atoms by*
   $$\mathcal{T}_f(\lambda(a, p)) = \lambda(f_A(a), f_P(p))$$
   The translation of formulas is given recursively by $\mathcal{T}_f(\neg\varphi) = \neg\mathcal{T}_f(\varphi)$, $\mathcal{T}_f(\varphi_1 \wedge \varphi_2) = \mathcal{T}_f(\varphi_1) \wedge \mathcal{T}_f(\varphi_2)$ *and* $\mathcal{T}_f(\Box\varphi) = \Box\mathcal{T}_f(\varphi)$.

Definition 7 (Safety Preserving Morphism). *A morphism $f : N_1 \to N_2$ is called safety preserving, iff for all safety properties $\Box\varphi$ we have $N_1 \models \Box\varphi \Longrightarrow N_2 \models \mathcal{T}_f(\Box\varphi)$.*

Definition 8 captures the notion of safety preserving rules. There must be a safety preserving morphism both on the level of the rule as well as on the level of its application, i. e. the transformation.

Definition 8 (Safety Preserving Rule and Transformation). *Given a rule $r = (L \leftarrow K \rightarrow R)$ in an HLR system and an arbitrary morphism $f : L \to R$. The pair (r, f) is a safety preserving rule, iff*

1. *f is a safety preserving morphism*
2. *given a net N_1, and an occurrence $g_1 : L \to N_1$, the direct transformation $N_1 \overset{r}{\Longrightarrow} N_2$ yields a safety preserving morphism $\overline{f} : N_1 \to N_2$.*

We denote the direct transformation by $N_1 \overset{(r,f)}{\Longrightarrow} N_2$ and call it safety preserving transformation, short sp-transformation.

Generally, it is difficult to determine in advance whether the application of a rule yields a special kind of morphism. However, the theory of $\mathcal{Q}$-transformation introduced in [Pad96, Pad99] and reviewed in Appendix A yields sufficient criteria. These are employed in the proofs of the subsequent theorems. Proofs can can be sketched here only due to space limitation, for details see [GPH00].

Theorem 1 (PP-Rules are Safety Preserving [PGE98]).

Proof. (Sketch) For condition 1 of Definition 8 the corresponding proof of Theorem 3.5 (Place Preserving Morphisms Preserve Safety Properties) in [PGE98] is lifted to AHL net systems which involves a detailed treatment of the initial marking. Analogously, for condition 2 the proof of Theorem 4.2 (Rule-Based Refinement Preserves Safety Properties) in [PGE98] has to be extended.

Theorem 2 (TG-Rules are Safety Preserving).

Proof. (Sketch) Condition 1 of Definition 8 is due to the fact that transition gluing morphism reflect firing. Condition 2 can be shown by showing the Q-conditions of Definition 14 in Appendix A. For pushouts in **QAHLSys** it can be proven by contradiction that their components are pushouts in the underlying categories. Furthermore, the initial marking and the set of transition conditions of the pushout object is given by the supremum of the (translated) markings and conditions of the source net systems. These facts are basically used for the proof of preservation of pushouts and inheritance of transition gluing morphisms. Analogously, inheritance of transition gluing morphisms under coproducts uses the componentwise construction and corresponding facts about the initial marking and transition conditions.

Our last main theorem concerns rules that introduce new safety properties. In fact, these are safety preserving as well. The introduction of safety properties by rules is given by safety properties satiesfied by the AHL net system R and preserved by the occurence of R in the AHL net system N_2. Hence then N_2 also satisfied this property. In this sense it is introduced.

Definition 9 (Safety Introducing). *Given a rule $r = (L \leftarrow K \rightarrow R)$ and a safety property $\Box\varphi$ over R s. t. $R \models \Box\varphi$. If $o : R \rightarrow N_2$ is safety preseving for all transformations $N_1 \overset{r}{\Longrightarrow} N_2$, r is called safety introducing rule.*

Definition 10 (SI-Rule). *A rule of the form $r = (\emptyset \leftarrow \emptyset \rightarrow R)$ is called si-rule.*

Example 3 (SI-Rule in Example). Figure 2 depicts an si-rule. The safety property $\Box[(a, \textbf{patient at ward}) \Longleftrightarrow (d, \textbf{ward documents})]$ is introduced.

Theorem 3 (SI-Rules are Safety Introducing and Preserving).

Proof. (Sketch) The application of an si-rule to a net N yields the coproduct $N + R$ and an occurence $o : R \rightarrow N + R$ which is a coproduct injection and therefore place preserving. These morphisms preserve safety properties. Furthermore, the unique morphism $\emptyset : \emptyset \rightarrow R$ (from the left-hand side to the right-hand side) is place preserving. Thus si-rules are a special case of pp-rules and Theorem 1 yields the stated proposition.

Main Conceptual Result 11 (Stepwise Development of Safe Models). *The concept of introducing safety properties (see Definition 10) enables development of an AHL net system hand-in-hand with its safety properties. The safety property has to be proven only once for the right-hand side of the rule. Subsequent refinement by si-rules, pp-rules, or tg-rules does not only preserve the explicitly introduced, but also all implicit safety properties (see Theorems 1, 2, and 3). Hence, employing these kinds of rules in the development of a system supports the development of safe models.*

4 Conclusion

In this paper we have presented our approach of rule-based refinement based on the introduction and preservation of safety properties. The technical results concern algebraic high-level net systems. For those we present safety preserving morphisms, rules and transformations. Place preserving morphisms preserve safety properties, since the adjacency of the places is not changed. Transition gluing morphisms preserve safety properties since the glued transitions has the union of the pre and post domains of the mapped transitions. Both classes of morphisms satisfy the $\mathcal{Q}$ conditions (see Apendix A). Thus, we obtain safety preserving rules and transformations. Safety introducing rules add a new isolated

subnet, that satisfies some new safety property. Moreover, these rules preserve safety properties as the corresponding morphism is place preserving.

Conceptually, we have extended our approach of rule-based refinement. Rule-based refinement is a formal description of stepwise development of system model. This technique is especially useful in early phases of the software development process. Each step is given by an application of rules. These rules introduce new safety properties or preserve those that already hold in the system. These steps can even be taken in parallel provided some independence conditions are satisfied (for these results see [Pad96, Pad99]). Moreover, the compatibility with horizontal structuring techniques (as given in [Pad99]) requires detailed knowledge of the transformation as well as the structuring technique. This can be expressed in independence conditions as they are given in our approach. The applicability fo safety introducing as well as preserving rules is the same as for usual rules in net transformation systems. The structure of the rules may prohibit the application of a rule to a given net. This is the case if its application could violate either the net or the safety property.

Future work concerns further system properties. Rule-based refinement has to be extended to comprise further system properties. Our approach is designed to include any system property provided it can be expressed using morphisms. Hence, we are going to investigate various morphism classes for certain properties. Liveness is one of those properties that are most interesting for this line of research.

References

[DG98] W. Deiters and V. Gruhn. Process Management in Practice - Applying the FunSoft Net Approach to Large-Scale Processes. *Automated Software Engineering*, 5:7–25, 1998.

[EGP99] H. Ehrig, M. Gajewsky, and F. Parisi-Presicce. *High-Level Replacement Systems with Applications to Algebraic Specifications and Petri Nets*, volume 3: Concurrency, Parallelism, and Distribution, chapter 6, pages 341–400. World Scientific, Handbook of Graph Grammars and Computing by Graph Transformations edition, 1999.

[EHKP91] H. Ehrig, A. Habel, H.-J. Kreowski, and F. Parisi-Presicce. Parallelism and concurrency in High Level Replacement Systems. *Math. Struc. in Comp. Science*, 1:361–404, 1991.

[EM85] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1: Equations and Initial Semantics*, volume 6 of *EATCS Monographs on Theoretical Computer Science*. Berlin, 1985.

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

[GPH00] Maike Gajewsky, Julia Padberg, and Kathrin Hoffmann. Safety Introducing and Preserving Rules for Algebraic High-Level Net Systems. Technical report, Technical University Berlin, 2000. to appear.

[Gru91] V. Gruhn. *Validation and Verification of Software Process Models*. PhD thesis, Universität Dortmund, Abteilung Informatik, 1991.

[JCHH91] K. Jensen, S. Christensen, P. Huber, and M. Holla. *Design/CPN. A Reference Manual*. Meta Software Cooperation, 125 Cambridge Park Drive, Cambridge Ma 02140, USA, 1991.

[Jen92] K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*, volume 1: Basic Concepts. Springer Verlag, EATCS Monographs in Theoretical Computer Science edition, 1992.

[JR91] K. Jensen and G. Rozenberg, editors. *High-Level Petri-Nets: Theory and Application*. 1991.

[MM90] J. Meseguer and U. Montanari. Petri Nets are Monoids. *Information and Computation*, 88(2):105–155, 1990.

[MP92] Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems, Specification*. 1992.

[OSS94] A. Oberweis, G. Scherrer, and W. Stucky. INCOME/STAR: Methodology and Tools for the Development of Distributed Information Systems. *Information Systems*, 19(8):643–660, 1994.

[Pad96] J. Padberg. *Abstract Petri Nets: A Uniform Approach and Rule-Based Refinement*. PhD thesis, Technical University Berlin, 1996. Shaker Verlag.

[Pad99] Julia Padberg. Categorical Approach to Horizontal Structuring and Refinement of High-Level Replacement Systems. *Applied Categorical Structures*, 7(4):371–403, December 1999.

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

[PGE98] J. Padberg, M. Gajewsky, and C. Ermel. Rule-Based Refinement of High-Level Nets Preserving Safety Properties. In E. Astesiano, editor, *Fundamental approaches to Software Engineering*, pages 221–238, 1998. Lecture Notes in Computer Science 1382.

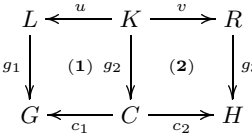
[Rei85] W. Reisig. *Petri Nets*, volume 4 of *EATCS Monographs on Theoretical Computer Science*. 1985.

[Vau87] J. Vautherin. Parallel System Specification with Coloured Petri Nets. In G. Rozenberg, editor, *Advances in Petri Nets 87*, pages 293–308, 1987. 266.

A Review of High-Level Replacement Systems

Here we briefly review the concepts of high-level replacement (HLR) systems in the sense of [EHKP91], a categorical generalization of graph grammars. High-level replacement systems are formulated for an arbitrary category $\mathbf{C}$ with a distinguished class $\mathcal{M}$ of morphisms.

Definition 12 (Rules and Transformations). *A* rule $r = (L \xleftarrow{u} K \xrightarrow{v} R)$ *in* $\mathbf{C}$ *consists of the objects L, K and R, called left-hand side, interface (or gluing object) and right-hand side respectively, and two morphisms $K \xrightarrow{u} L$ and $K \xrightarrow{v} R$ with both morphisms $u, v \in \mathcal{M}$, a distinguished class of morphisms in $\mathbf{C}$. Given a rule $r = (L \xleftarrow{u} K \xrightarrow{v} R)$ a* direct transformation $G \overset{r}{\Longrightarrow} H$*, from an object G to an object H is given by two pushout diagrams*



(1) *and* **(2)** *in the category* $\mathbf{C}$*. The morphisms $L \xrightarrow{g_1} G$ and $R \xrightarrow{g_3} H$ are called occurrences of L in G and R in H, respectively. By an occurrence of rule $r = (L \xleftarrow{u} K \xrightarrow{v} R)$ in a structure G we mean an occurrence of the left-hand side L in G.*

A transformation sequence $G \overset{*}{\Longrightarrow} H$, *short transformation, between objects* G *and* H *means* G *is isomorphic to* H *or there is a sequence of* $n \geq 1$ *direct transformations:* $G = G0 \overset{r_1}{\Longrightarrow} G1 \overset{r_2}{\Longrightarrow} \ldots \overset{r_n}{\Longrightarrow} Gn = H$.

Definition 13 (High-Level Replacement System). *Given a category* $\mathbf{C}$ *together with a distinguished class of morphisms* $\mathcal{M}$ *then* $(\mathbf{C}, \mathcal{M})$ *is called a HLR-category if* $(\mathbf{C}, \mathcal{M})$ *satisfies the HLR-Conditions (see [EGP99]).*

The main idea in the following definition is to enlarge the given HLR-category in order to include morphisms, that are adequate for refinement. The $\mathcal{Q}$-conditions [Pad99] state additional requirements, that an HLR-category has to satisfy for the extension to refinement morphisms.

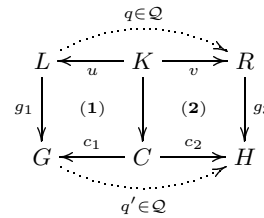
Definition 14 ($\mathcal{Q}$: Refinement Morphism [Pad99]). *Let* **QCat** *be a category, so that* $\mathbf{C}$ *is a subcategory* $\mathbf{C} \subseteq \mathbf{QCat}$ *and* $\mathcal{Q}$ *a class of morphisms in* **QCat**.

1. *The morphisms in* $\mathcal{Q}$ *are called* $\mathcal{Q}$*-morphisms, or refinement morphisms.*
2. *Then we have the following* $\mathcal{Q}$*-conditions:*
   Closedness: $\mathcal{Q}$ *has to be closed under composition.*
   Preservation of Pushouts: *The inclusion functor* $I : \mathbf{C} \to \mathbf{QCat}$ *preserves pushouts, that is, given* $C \overset{f'}{\longrightarrow} D \overset{g'}{\longleftarrow} B$ *a pushout of* $B \overset{f}{\longleftarrow} A \overset{g}{\longrightarrow} C$ *in* $\mathbf{C}$, *then* $I(C) \overset{I(f')}{\longrightarrow} I(D) \overset{I(g')}{\longleftarrow} I(B)$ *is a pushout of* $I(B) \overset{I(f)}{\longleftarrow} I(A) \overset{I(g)}{\longrightarrow} I(C)$ *in* **QCat**.
   Inheritance of $\mathcal{Q}$-morphisms under Pushouts: *The class* $\mathcal{Q}$ *in* **QCat** *is closed under the construction of pushouts in* **QCat**, *that is, given* $C \overset{f'}{\longrightarrow} D \overset{g'}{\longleftarrow} B$ *a pushout of* $B \overset{f}{\longleftarrow} A \overset{g}{\longrightarrow} C$ *in* **QCat**, *then* $f \in \mathcal{Q} \implies f' \in \mathcal{Q}$.
   Inheritance of $\mathcal{Q}$-morphisms under Coproducts: *The class* $\mathcal{Q}$ *in* **QCat** *is closed under the construction of coproducts in* **QCat**, *that is, for* $A \overset{f}{\longrightarrow} B$ *and* $A' \overset{f'}{\longrightarrow} B'$ *we have* $f, f' \in \mathcal{Q} \implies f + f' \in \mathcal{Q}$ *provided the coproduct* $A + A' \overset{f+f'}{\longrightarrow} B + B'$ *of* f *and* f' *exists in* **QCat**.
3. *A* $\mathcal{Q}$*-rule* (r, q) *is given by a rule* $r = L \overset{u}{\longleftarrow} K \overset{v}{\longrightarrow} R$ *in* $\mathbf{C}$ *and a* $\mathcal{Q}$*-morphism* $q : L \to R$, *so that* $K \overset{u}{\longrightarrow} L \overset{q}{\longrightarrow} R = K \overset{v}{\longrightarrow} R$ *in* **QCat**.

The next fact states the class $\mathcal{Q}$ is also preserved under transformations.

Lemma 1 ($\mathcal{Q}$-Transformations [Pad99]). *Let* $\mathbf{C}$, **QCat**, $\mathcal{Q}$, *and* $I : \mathbf{C} \to \mathbf{QCat}$ *satisfy the* $\mathcal{Q}$*-conditions. Given a* $\mathcal{Q}$*-rule* (r, q) *and a transformation* $G \overset{r}{\Longrightarrow} H$ *in* $\mathbf{C}$

defined by the pushouts **(1)** *and* **(2)**, *then there is a unique* $q' \in \mathcal{Q}$, *such that* $q' \circ c_1 = c_2$ *and* $q' \circ g_1 = g_3 \circ q$ *in* **QCat**. *The transformation* $(G \overset{r}{\Longrightarrow} H, q' : G \to H)$, *or short* $G \overset{(r,q')}{\Longrightarrow} H$, *is called* $\mathcal{Q}$*-transformation.* $R \overset{g_3}{\longrightarrow} H \overset{q'}{\longleftarrow} G$ *is pushout of* $G \overset{g_1}{\longleftarrow} L \overset{q}{\longrightarrow} R$ *in* **QCat**.



B Algebraic High-Level Net Sytems

Definition 15 (Category CMON of Free Commutative Monoids). *Let P be a set. Then $(P^{\oplus}, \lambda, \oplus)$ is called the free commutative monoid generated by P, s.t. for all $u, v, w \in P^{\oplus}$ the following equations hold:*

- $\lambda \in P^{\oplus}$
- $v \oplus \lambda = \lambda \oplus v = v$,
- $u \oplus (v \oplus w) = (u \oplus v) \oplus w$
- $v \oplus w = w \oplus v$

Elements w of the free commutative monoid $P^{\oplus}$ are called linear sums. They can be represented as $w = \sum_{p \in P} \lambda_p \cdot p$ with coefficients $\lambda_p \in \mathbb{N}$ and $p \in P$. They can be considered as multisets.

Free commutative Monoids together with set-based monoid homomorphism define the category **CMON**.

Definition 16 (Operations of Free Commutative Monoids). *Free commutative monoids imply the operations $\oplus, \ominus, \leq$ and $\geq$ on linear sums. These are the obvious addition, subtraction and comparison of coefficients on linear sums.*
Let $M_1, M_2 \in P^{\oplus}$ with $M_1 = \sum_{p \in P} \lambda_p \cdot p$ and $M_2 = \sum_{p \in P} \lambda'_p \cdot p$, then

$M_1 \leq M_2$ *, if* $\forall p \in P : \lambda_p \leq \lambda'_p$
$M_1 \geq M_2$ *, if* $\forall p \in P : \lambda_p \geq \lambda'_p$
$M_1 \oplus M_2 = \sum_{p \in P} (\lambda_p + \lambda'_p) \cdot p$
$M_1 \ominus M_2 = \sum_{p \in P} (\lambda_p - \lambda'_p) \cdot p$, *if* $M_1 \leq M_2$

Lemma 2 (Induced Functions of Place Vector). *Let $m \in (A \times P)^{\oplus}$ Then we have $m : P \to A^{\oplus}$. Similarily, let $m \in (T_{OP}(X) \times P)^{\oplus}$. Then we have $m : P \to T^{\oplus}_{OP}(X)$.*

Definition 17 (Pre- and Post Set). *Given an AHL net system $N = (SPEC, A, P, T, pre, post, cond, \widehat{m})$ then we define*

$$\bullet p = \sum_{t \in T} post(t)(p) \cdot t \in (T_{OP}(X) \times T)^{\oplus}$$

and

$$p \bullet = \sum_{t \in T} pre(t)(p) \cdot t \in (T_{OP}(X) \times T)^{\oplus}$$

Definition 18 (Restrictions of Place Vector). *Given $m \in (A \times P)^{\oplus}$ and let $P' \subseteq P$, then there is $m_1 \in (A \times P')^{\oplus}$ and $m_2 \in (A \times (P \setminus P'))^{\oplus}$ such that $m = m_1 \oplus m_2$.*
We define $m_{|P'} = m_1$.
Moreover there are two important special cases:

1. *$P' = \{p\}$ denoted with $m_{|p}$*
2. *$P' = f_P(P_0)$ for some function $f_P : P_0 \to P$, denoted with $m_{|f_P}$*

Lemma 3 (Restrictions are Compatible with Monoid-Operations). *Let* $P' \subseteq P$ *and* $m \in (A \times P)^{\oplus}$ *and* $p \in P_1$ *. Then we have*

$$
\begin{array}{lrcl}
(1) & (m \oplus m')_{|P'} & = & m_{|P'} \oplus m'_{|P'} \\
(2) & (m \ominus m')_{|P'} & = & m_{|P'} \ominus m'_{|P'} \\
(3) & m \leq m' & \Longrightarrow & m_{|P'} \leq m'_{|P'} \\
(4) & f_M(m)_{|f_P} & = & f_M(m) \\
(5) & f_M(m_{1|f_P}) & = & m_{2|f_P(p)} \Longleftrightarrow m_1(p) = m_2(f_P(p))
\end{array}
$$