Duo-Internal Labeled Graphs with Distinguished Nodes: a Categorial Framework for Graph Based Anticipatory Systems *

P. Blauth Menezes

II/ UFRGS - Caixa Postal 15064, 91501-970, Porto Alegre, RS, Brazil

Fax: +55 51 319-1576, e-mail: blauth@inf.ufrgs.br

Abstract

A categorial framework for structured graph based systems with or without distinguished nodes or labeling on both arcs and nodes is proposed. Requirements for the existence of limits and colimits in the resulting categories are set. In this context, unrestricted and bicomplete categories of graph based systems such as Petri Nets, Labeled Transition Systems, Nonsequential Automata, etc., are easily defined. Then it is shown how limits and colimits can be interpreted as structuring and anticipatory properties of systems. The proposed framework called duo-internalization generalizes the notion of internal graphs allowing that nodes and arc may be objects from different categories. The results about limits and colimits of (reflexive) duo-internal (labeled) graphs (with distinguished nodes) are, for our knowledge, new.

Keywords. Graph based systems, anticipatory systems, duo-internal graphs, graph transformation, category theory

1 Introduction

Graph based systems such as Petri Nets (Reisig, 1985), Transition Systems (Milner, 1989), Asynchronous Transitions Systems (Bednarczyk, 1988) and Nonsequential Automata (Menezes & Costa, 1995) are some of the models for concurrency developed and used in many applications. Several categorial frameworks for graph based systems have been proposed for expressing the semantics of concurrent systems mainly in the so called true concurrency approach as in (Meseguer & Montanari, 1990), (Sassone *et al*, 1993) and (Menezes *et al*, 1998).

An important justification (among others) for the use of category theory is that of structuring, in the sense that most of the graph based systems are not equipped with compositional operations. A step toward structuring is provided in (Winskel, 1984) and (Winskel, 1987) where the categorial constructions of product and coproduct stand for parallel and nondeterministic composition operation of nets, respectively. In some categorial graph based systems such as Petri nets and Nonsequential Automata, if an initial marking/state is added, the resulting category may not have coproducts. Initial

** This work is partially supported by: FAPERGS (Project QaP-For), CNPq (Projects HoVer-CAM, GRAPHIT) and CAPES (Project TEIA) in Brazil.*

marking/state are used for defining the operational semantics for concurrent languages (see, for instance, (Degano *et al*, 1988), (Degano & Montanari, 1987), (Winskel, 1984), (Olderog, 1987) and (Glabbeek & Vaandrager, 1987)). To solve this problem, some restrictive solutions where proposed: restrictions on categories and morphisms (Winskel, 1987); restrictions on initial marking/state (Meseguer & Montanari, 1990); simulation of the coproduct construction using a functorial operation based on fibration technique (Menezes & Costa, 1996).

However, it may be the case that colimits (or coproducts in special) are needed for unrestricted graph based systems for any reason. An interesting example is the use of graph transformation using the so called double pushout approach (Ehrig, 1979). In this case, graph transformations extended for graph based systems may have several interpretation such as systems refinement, dynamic specification or anticipation (as proposed in (Menezes, 1999)) of systems and modeling system behavior (as a token game in Petri nets).

In this paper we propose a generalized categorial framework for defining structured graph based systems with or without distinguished nodes or labeling on both arcs and nodes. In this context, we show some requirements for preserving limits and colimits properties from the structuring categories. Therefore, *unrestricted* and *bicomplete* categories of graph based systems such as Petri nets, Transition Systems, Nonsequential Automata, etc., are easily defined. Then we show how limits and colimits can be interpreted as structuring and anticipatory properties of systems. The anticipatory properties are inspired by (Menezes, 1999), but in a different framework.

The proposed framework is based on internal graphs, generalized in order to allow that nodes and arcs may be objects from different categories. Initially, a (small) graph can be defined as a quadruple $G = \langle V, T, \partial_0, \partial_1 \rangle$ where V is a set of nodes, T is a set of arcs $\partial_0, \partial_1 : T \to V$ are two functions called source and target which associate for each arc the corresponding source and target nodes, respectively. As stated in (Corradini, 1990) and (Asperti & Longo, 1991) a graph G can be considered as a diagram in the category *Set* where V and T are sets and ∂_0 and ∂_1 are (total) functions. Moreover, graph morphisms are commutative diagrams in *Set*. This means that *Set* plays the role of "universe of discourse" of the category *Gr* (of graphs): it is defined internally to the category *Set*. This suggests a generalization of graphs as diagrams in an arbitrary universe (base) category. This approach is known as *internalization* and can be extended for reflexive graphs and categories (for categories, see (Corradini, 1990) and (Menezes, 1997)). However, nodes and arcs may be objects of different categories, provided that there are functors from the categories of nodes and arcs to the base category. This notion we call *duo-internalization*.

Graph based systems usually are (structured) graphs with some special features such as labeling and distinguished nodes. A distinguished node is a node specially identified and its interpretation may vary according to the application. For instance, common interpretations are initial, final, abort or "colored" state/markings. The proposed approach for distinguished nodes was first sketched in (Menezes, 1995) where an unrestricted bicomplete category of Petri net is introduced (with a set of initial marking, inspired by (Jonsson, 1990)).

In most cases, labeling of graphs is restricted to arcs. However, it might be the case that labeling must be both, on arcs and nodes. In this case, it is expected that labeling on arcs should preserve labeling on source and target nodes. Therefore, labeling can be seen as a graph-morphism where the source graph is the "shape" and the target one are the "labels of arcs and nodes" (as in (Menezes, 1994)).

For all categories of (reflexive) duo-internal graphs with/without labeling or distinguished nodes introduced, conditions for the existence of limits and colimits are set. The results about duo-internalization are, for our knowledge, new.

2 Internal Graphs

The Category of Internal Graphs is defined using the notions of comma category and diagonal functor. In this context, we show that the properties about limits and colimits of the base category are inherited by the category of internal graphs. As expected, using this result, the category of (small) graphs is bicomplete.

Definition 1 Let C be a category. The *Diagonal Functor* Δ_C: $C \rightarrow C^2$ is such that sends each C-object A to the C^2-object $\langle A, A \rangle$ and sends each C-morphism f: $A \rightarrow B$ to the C^2-morphism $\langle f, f \rangle$: $\langle A, A \rangle \rightarrow \langle B, B \rangle$. ❑

Proposition 2 Let C be a category. Consider the diagonal functor Δ_C: $C \rightarrow C^2$. Then:

a) If C has all binary products, then Δ_C preserves colimits;
b) If C has all binary coproducts, then Δ_C preserves limits.

Proof:

a) If C has all binary products, then the functor Π : $C^2 \rightarrow C$ (induced by the product construction) such that sends each C^2-object $\langle A, B \rangle$ the C-object $A \times B$ and sends each C^2-morphism $\langle f, g \rangle$ to the C-morphism $f \times g$ uniquely induced by the product in C, is right adjoint to Δ_C (Mac Lane, 1971). Therefore, Δ_C preserves colimits;
b) If C has all binary coproducts, then the functor $\amalg$: $C^2 \rightarrow C$ (induced by the coproduct) such that sends each C^2-object $\langle A, B \rangle$ to the C-object $A + B$ and sends each C^2-morphism $\langle f, g \rangle$ to the C-morphism $f + g$ uniquely induced by the coproduct in C, is left adjoint to Δ_C (Mac Lane, 1971). Therefore, Δ_C preserves limits. ❑

Definition 3 Let *f*: $A \rightarrow C$ and *g*: $B \rightarrow C$ be functors. The *Comma Category* $f \downarrow g$ is such that (Fig. 1):

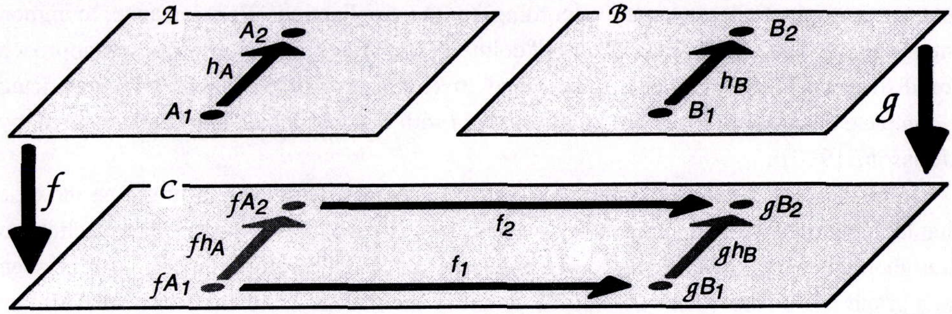

Fig. 1: Comma category

a) An object is a triple $S = \langle A, f, B\rangle$ where A is an A-object, B is a B-object and $f: fA \rightarrow gB$ is a C-morphism;

b) A morphism is a pair $h = \langle h_A, h_B\rangle: \langle A_1, f_1, B_1\rangle \rightarrow \langle A_2, f_2, B_2\rangle$ where $h_A: A_1 \rightarrow A_2$ is a A-morphism and $h_B: B_1 \rightarrow B_2$ is a B-morphism such that $g\, h_B \circ f_1 = f_2 \circ f h_A$

c) The identity morphism of an object $S = \langle A, f, B\rangle$ is $\iota_S = \langle \iota_A: A \rightarrow A, \iota_B: B \rightarrow B\rangle$

d) The composition of two morphisms $f = \langle f_A, f_B\rangle: S_1 \rightarrow S_2$ and $g = \langle g_A, g_B\rangle: S_2 \rightarrow S_3$ is $g \circ f = \langle g_A \circ f_A, g_B \circ f_B\rangle: S_1 \rightarrow S_3$ ❑

Proposition 4 Let $f: A \rightarrow C$ and $g: B \rightarrow C$ be functors. Then:

a) If A, B are complete and g preserves limits then the comma category $f\downarrow g$ is complete;

b) If A, B are cocomplete and f preserves colimits then the comma category $f\downarrow g$ is cocomplete.

Proof: See, for instance, (Casley, 1991). ❑

Definition 5 Let C be a (base) category. The *Category of Internal Graphs over C*, denoted by *Gr(C)*, is the comma category $\Delta_C \downarrow \Delta_C$. ❑

Therefore, an internal graph over C can be seen as quadruple $G = \langle V, T, \partial_0, \partial_1\rangle$ where V, T are C-objects and $\partial_0, \partial_1: T \rightarrow V$ are C-morphisms. An internal graph morphism $h = \langle h_V, h_T\rangle: \langle V_1, T_1, \partial_{0_1}, \partial_{1_1}\rangle \rightarrow \langle V_2, T_2, \partial_{0_2}, \partial_{1_2}\rangle$ where $h_V: V_1 \rightarrow V_2$ and $h_T: T_1 \rightarrow T_2$ are C-morphisms such that the diagrams in Fig. 2 commute.

Remark 6 The category of graphs *Gr* is the category of internal graphs *Gr(Set)*. ❑

Proposition 7 Let C be a category. Then:

a) If C is complete, then *Gr(C)* is complete;

b) If C is cocomplete, then *Gr(C)* is cocomplete.

Proof: Since *Gr(C)* is the comma category $\Delta_C \downarrow \Delta_C$ and since Δ_C preserves limits and colimits, *Gr(C)* is bicomplete. ❑

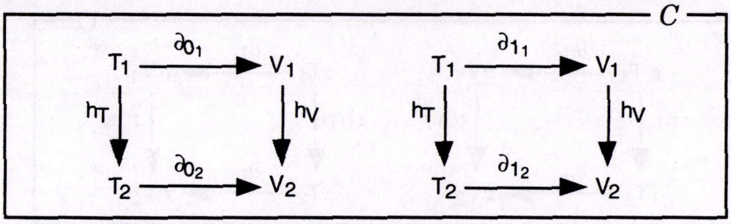

Fig. 2: Commutative diagrams for morphisms between internal graphs

Proposition 8 The category of graphs *Gr = Gr(Set)* is bicomplete.

Proof: Since *Set* is bicomplete, *Gr(Set)* is bicomplete. ❑

3 Duo-Internal Graphs

Duo-internal graphs allow the definition of a special kind of graphs where nodes and arcs may be objects from different categories. They are defined over internal graphs provided that there are functors from the categories of nodes and arcs to the base category. In a graph, the source and target morphisms are taken from the base category. In this context, limits and colimits of categories of duo-internal graphs are inherited from the categories of nodes and arcs.

Definition 9 Let C be a (base) category and $v\colon V \to C$ and $t\colon T \to C$ be functors. The *Category of Duo-Internal Graphs* denoted by *Gr(v, t)*, is the comma category $\Delta_C \circ t \downarrow \Delta_C \circ v$ ❑

Therefore, a *Gr(v, t)*-object G is a quadruple $G = \langle V, T, \partial_0, \partial_1 \rangle$ where V is a V-object, T is a T-object and $\partial_0, \partial_1\colon tT \to vV$ are C-morphisms and a *Gr(v, t)*-morphism $h = \langle h_V, h_T \rangle\colon \langle V_1, T, \partial_{0_1}, \partial_{1_1} \rangle \to \langle V_2, T, \partial_{0_2}, \partial_{1_2} \rangle$ where $h_V\colon V_1 \to V_2$ is a V-morphism and $h_T\colon T_1 \to T_2$ is a T-morphism, such that the diagrams in Fig. 3 commute.

Remark 10 The category of (small) graphs *Gr* is $Gr(id_{Set}, id_{Set})$ and the category of internal graphs *Gr(C)* is $Gr(id_C, id_C)$. ❑

Proposition 11 Let $v\colon V \to C$ and $t\colon T \to C$ be functors. Then:

a) If V, T are complete and v preserves limits then *Gr(v, t)* is complete;

b) If V, T are cocomplete and t preserves colimits then *Gr(v, t)* is cocomplete.

Proof: Since *Gr(v, t)* is the comma category $\Delta_C \circ t \downarrow \Delta_C \circ v$, the proof is a direct corollary. ❑

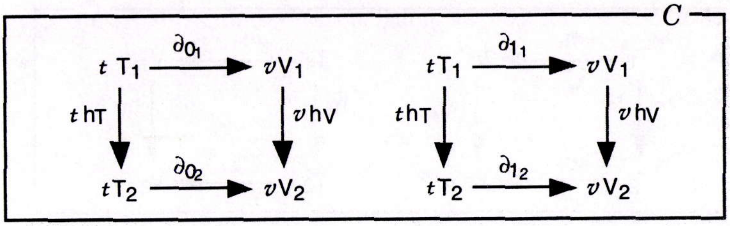

Fig. 3: Commutative diagrams for duo-internal graphs

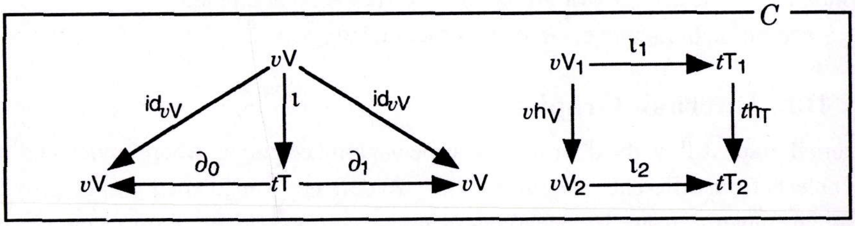

Fig. 4: Commutative diagrams for duo-internal reflexive graphs

4 Duo-Internal Reflexive Graphs

The category of duo-internal reflexive graphs is just an extension of the category of duo-internal graphs where, for each node, the identity morphism is a morphism in the base category. In this context, the properties about limits and colimits are also inherited from the categories of nodes and arcs. Using the results about duo-internal reflexive graphs it is straightforward to verify that the category of (small) reflexive graphs is bicomplete.

Definition 12 Let C be a (base) category and $v\colon V \to C$ and $t\colon T \to C$ be functors. Then:

a) A duo-internal reflexive graph is a quadruple $\mathsf{G} = \langle \mathsf{V}, \mathsf{T}, \partial_0, \partial_1, \iota\rangle$ where $\langle \mathsf{V}, \mathsf{T}, \partial_0, \partial_1\rangle$ is a $Gr(v, t)$-object and $\iota\colon v\,\mathsf{V} \to t\,\mathsf{T}$ is a C-morphism such that the diagram in Fig. 4 (left) commutes;

b) A morphism between duo-internal reflexive graphs $\mathsf{h} = \langle \mathsf{h_V}, \mathsf{h_T}\rangle\colon \langle \mathsf{V_1}, \mathsf{T}, \partial_{0_1}, \partial_{1_1}, \iota_1\rangle \to \langle \mathsf{V_2}, \mathsf{T}, \partial_{0_2}, \partial_{1_2}, \iota_2\rangle$ is a $Gr(v, t)$-morphism such that the diagram in Fig. 4 (right) commutes;

c) Duo-internal reflexive graphs and the corresponding morphisms constitute the *Category of Duo-Internal Reflexive Graphs*, denoted by $RGr(v, t)$. ❑

Remark 13 The category of (small) reflexive graphs RGr is $RGr(id_{Set}, id_{Set})$ and the category of internal reflexive graphs $RGr(C)$ is $RGr(id_C, id_C)$. ❑

Proposition 14 Let $v\colon V \to C$ and $t\colon T \to C$ be functors. Then:

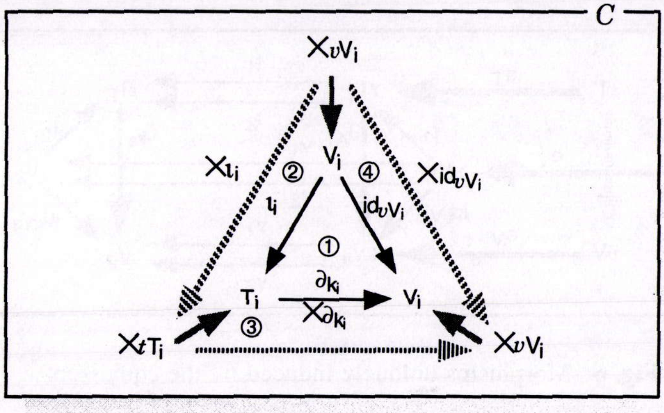

Fig. 5: Morphisms uniquely induced by the product

a) If V and T are complete and υ preserves limits then $RGr(\upsilon, t)$ is complete;

b) If V and T are cocomplete and t preserves colimits then $RGr(\upsilon, t)$ is cocomplete.

<u>Proof:</u> Let u: $RGr(\upsilon, t) \to Gr(\upsilon, t)$ be a functor such that, for each $RGr(\upsilon, t)$-object, $G = \langle V, T, \partial_0, \partial_1, \iota \rangle$, $u\,G = \langle V, T, \partial_0, \partial_1 \rangle$ and for each $RGr(\upsilon, t)$-morphism $h = \langle h_V, h_T \rangle$: $G_1 \to G_2$, $u\,h = \langle h_V, h_T \rangle$: $u\,G_1 \to u\,G_2$. Since u is a faithful functor, $\langle RGr(\upsilon, t), u \rangle$ is concrete category over $Gr(\upsilon, t)$ (Adámek *et al*, 1990). Suppose that V, T are complete (cocomplete), υ preserves limits (t preserves colimits). Then $Gr(\upsilon, t)$ is complete (cocomplete). Therefore, to prove that $RGr(\upsilon, t)$ is complete (cocomplete) we have just to prove that the for each $RGr(\upsilon, t)$-diagram D the limit (colimit) D in $Gr(\upsilon, t)$ can be lifted as a initial source (final sink) in $RGr(\upsilon, t)$. Suppose I a family of indexes, $i \in I$ and $k \in \{0, 1\}$. For simplicity, in what follows, we omit that $i \in I$ and $k \in \{0, 1\}$.

Lifting products. Let $\{G_i = \langle V_i, T_i, \partial_{0_i}, \partial_{1_i}, \iota_i \rangle\}$ be an indexed family of $RGr(\upsilon, t)$-objects and $\times u\,G_i = \langle \times V_i, \times T_i, \times\partial_{0_i}, \times\partial_{1_i} \rangle$ the corresponding $Gr(\upsilon, t)$-product together with $\{\pi_i: \times u\,G_i \to u\,G_i\}$. Then, $\times G_i = \langle \times V_i, \times T_i, \times\partial_{0_i}, \times\partial_{1_i}, \times\iota_i \rangle$, together with $\{\pi_i: \times G_i \to G_i\}$ is an initial source of $\{G_i\}$ where $\times\iota_i$ is uniquely induced by the product construction as illustrated in Fig. 5. Then:

a) To prove that $\times G_i$ is a duo-internal reflexive graph it is enough to prove that the external diagram in Fig. 5 commutes. In fact, $\times id_{V_i}$ is the unique C-morphism such that ④ commutes. Since ①, ② and ③ commute, $\pi_{V_i} \circ \times id_{V_i} = id_{V_i} \circ \pi_{V_i} = \partial_{k_i} \circ \iota_i \circ \pi_{V_i} = \partial_{k_i} \circ \pi_{T_i} \circ \times\iota_i = \pi_{V_i} \circ \times\partial_{k_i} \circ \times\iota_i$. Therefore, by the uniqueness of $\times id_{V_i}$ in ④, $\times id_{V_i} = \times\partial_{k_i} \circ \times\iota_i$;

b) To prove that $\times G_i$ (together with $\{\pi_i\}$) is an initial source, consider a $RGr(\upsilon, t)$-source $\langle G, \{f_i: G \to G_i\} \rangle$. Since $\times u\,G_i$ is a product in $Gr(\upsilon, t)$, there is an unique $Gr(\upsilon, t)$-morphism h: $u\,G \to \times u\,G_i$ such that, $f_i = \pi_{u G_i} \circ h$. The lifting of h is h.

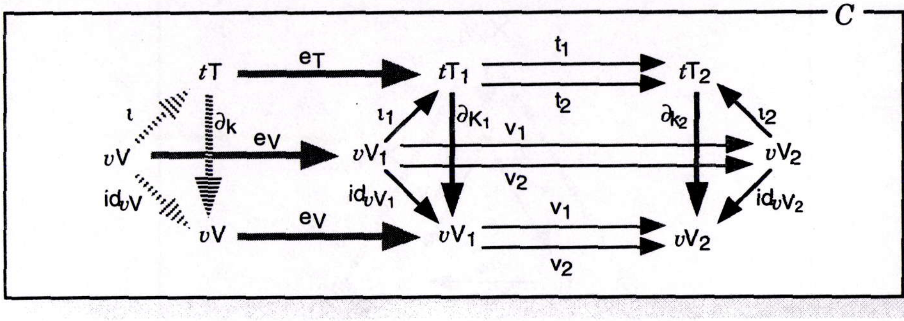

Fig. 6: Morphisms uniquely induced by the equalizers

Lifting equalizers. Consider the *RGr(v, t)*-objects $G_1 = \langle V_1, T_1, \partial_{0_1}, \partial_{1_1}, \iota_1\rangle$, $G_2 = \langle V_2, T_2, \partial_{0_2}, \partial_{1_2}, \iota_2\rangle$, the *RGr(v, t)*-morphisms $f_1 = \langle v_1, t_1\rangle$, $f_2 = \langle v_2, t_2\rangle$: $G_1 \to G_2$ and the corresponding *Gr(v, t)*-equalizer $\langle e_V, e_T\rangle$: $\langle V, T, \partial_0, \partial_1\rangle \to u\, G_1$. Then $G = \langle V, T, \partial_0, \partial_1, \iota\rangle$ together with $e = \langle e_V, e_T\rangle$: $G \to G_1$ is an initial source in *RGr(v, t)*, where ι is uniquely induced by the *C*-equalizers, as illustrated in Fig. 6. The proof showing that G together with e is an initial source is analogous to the proof for products.

The lifting of coproducts and coequalizers are analogous to the lifting of products and equalizers, respectively. ❑

Proposition 15 The category of reflexive graphs *RGr* is bicomplete.
Proof: Since *RGr* is *RGr*(id_{Set}, id_{Set}), *Set* is bicomplete and id_{Set} trivially preserves limits and colimits, *RGr* is bicomplete. ❑

Proposition 16 Let *C* be a category. Then:

a) If *C* is complete, then *RGr(C)* is complete;
b) If *C* is cocomplete, then *RGr(C)* is cocomplete.

Proof: Since *RGr(C)* is *RGr*(id_C, id_C) and id_C trivially preserves limits and colimits, if *C* is complete (cocomplete) *RGr(C)* is complete (cocomplete). ❑

5 Duo-Internal Graphs with Distinguished Nodes

The internalization approach is generalized for duo-internal (reflexive) graphs with distinguished nodes. The generalization can be both for the domain category of the (distinguished) nodes and for the category where the nodes are matched. In this context, the requirements for the bicompleteness of categories of duo-internal (reflexive) graphs with distinguished nodes are set. As expected, the results about limits and colimits for (small) graphs and internal graphs with distinguished nodes are just special cases. The proposed approach is analogous for duo-internal graphs and duo-internal reflexive graphs. Therefore, in what follows, *G(v, t)* denotes *Gr(v, t)* or *RGr(v, t)*.

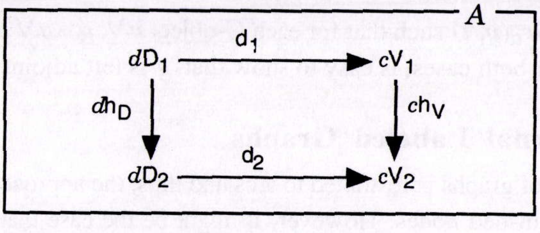

Fig. 7: Commutative diagram for graphs with distinguished nodes

Definition 17 The *Category of Duo-Internal (Reflexive) Graphs with Distinguished Nodes* is the comma category $G_d(v, t) = d \downarrow nodes$ where $d: D \rightarrow A$ is a functor where D is the domain category of distinguished nodes and A is the (target) category where the distinguished nodes are matched and $nodes = c \circ u: G(v, t) \rightarrow A$ is a functor such that:

a) $u: G(v, t) \rightarrow C$ is a forgetful functor that sends each $G(v, t)$-object to its corresponding nodes in C and each $G(v, t)$-morphism $h = \langle h_V, h_T \rangle$ to the C-morphism $v\, h_V$;

b) $c: C \rightarrow A$ is a functor that relates the base category of graphs with the category that matches the distinguished nodes. □

Therefore, a $G_d(v, t)$-object M is a triple $M = \langle D, d, G \rangle$, where G is a $G(v, t)$-graph, D is a D-object denoting the distinguished nodes and $d: d\,D \rightarrow nodes\,G$ is a A-morphism that matches the distinguished nodes in G. A $G_d(v, t)$-morphism is a pair $\langle h_G, h_D \rangle: \langle D_1, d_1, G_1 \rangle \rightarrow \langle D_2, d_2, G_2 \rangle$ where $\langle h_V, h_T \rangle: G_1 \rightarrow G_1$ is a $G(v, t)$-morphism and $h_D: D_1 \rightarrow D_1$ is a D-morphism, such that the diagram in Fig. 7 commutes.

Proposition 18 Let $d: D \rightarrow A$ and $c: C \rightarrow A$ be functors. Consider the category $G_d(v, t)$. Then:

a) *Limits for graphs*. Suppose that $G_d(v, t) = Gr_d(v, t)$. If C and D are complete, c preserves limits and C has initial object, then $G_d(v, t)$ is complete;

b) *Limits for reflexive graphs*. Suppose that $G_d(v, t) = RGr_d(v, t)$. If C and D are complete and c preserves limits, then $G_d(v, t)$ is complete;

c) *Colimits*. If C and D are cocomplete and d preserves colimits, then $G_d(v, t)$ is cocomplete.

Proof: Since $G_d(v, t)$ is the category $d \downarrow nodes$ where $nodes = c \circ v$, we have just to prove that the $v: G(v, t) \rightarrow C$ preserves limits. *Case 1*: for $G_d(v, t) = Gr_d(v, t)$, consider the C-initial object 0 and the functor $g: C \rightarrow Gr_d(v, t)$ such that for each C-object $v\,V$, $g \circ v\,V$ is the graph $\langle V, 0, !, ! \rangle$. *Case 2*: for $G_d(v, t) = RGr_d(v, t)$, consider the functor $g: C \to Gr_d(v, t)$ such that for each C-object $v\,\mathsf{V}$, $g \circ v\,\mathsf{V}$ is the graph $\langle \mathsf{V}, \mathsf{V}, \mathsf{id}_{v\mathsf{V}}, \mathsf{id}_{v\mathsf{V}}, \mathsf{id}_{v\mathsf{V}} \rangle$. For both cases, is easy to show that, g is left adjoint to v. ❑

6 Duo-Internal Labeled Graphs

Usually, the labeling of graphs is restricted to arcs and thus, the approach could be similar to the one for distinguished nodes. However, it might be the case that labeling must be both, on arcs and nodes. In this case, it may be expected that labeling for each arc should preserve the labeling of corresponding source and target nodes. In this context, labeling can be seen as a graph-morphism where the source graph represents the "shape" and the target one represents the "labels" (of arcs and nodes). Again, the requirements for the bicompleteness of categories of duo-internal graphs with labeling are set. The approach is analogous for duo-internal graphs and duo-internal reflexive graphs with and without distinguished nodes. Therefore, in what follows, $G(v, t)$ denotes $Gr(v, t)$, $RGr(v, t)$, $Gr_d(v, t)$ and $RGr_d(v, t)$.

Definition 19 The *Category of Duo-Internal Labeled Graphs* is the comma category $LG(shape, lab) = shape \downarrow lab$ where $shape\colon G(v_s, t_s) \to G(v, t)$, $lab\colon Lab \to G(v, t)$ are functors, $G(v_s, t_s)$ is the category of "shapes" and Lab is the category of "labels".❑

Therefore, a $LG(shape, lab)$-object is a triple $\mathsf{N} = \langle \mathsf{G}, \mathsf{lab}, \mathsf{L} \rangle$, where G is a graph in $G(v_s, t_s)$ representing the "shape", L is a Lab-object representing the "labels" and lab: $shape\,\mathsf{G} \to lab\,\mathsf{L}$ is a graph morphism in $G(v, t)$ corresponding to the labeling. A $LG(shape, lab)$-morphism is $\langle \mathsf{h_G}, \mathsf{h_L} \rangle\colon \langle \mathsf{G_1}, \mathsf{lab_1}, \mathsf{L_1} \rangle \to \langle \mathsf{G_2}, \mathsf{lab_2}, \mathsf{L_2} \rangle$ such that $lab\,\mathsf{h_L} \circ \mathsf{lab_1} = \mathsf{lab_2} \circ shape\,\mathsf{h_G}$.

Proposition 20 Let $shape\colon G(v_s, t_s) \to G(v, t)$, $lab\colon Lab \to G(v, t)$ be functors. Then:

a) If $G(v_s, t_s)$, Lab are complete and lab preserves limits then $LG(shape, lab)$ is complete;
b) If $G(v_s, t_s)$, Lab are cocomplete and $shape$ preserves colimits then $LG(shape, lab)$ is cocomplete.

<u>Proof:</u> Since $LG(shape, lab) = shape \downarrow lab$, the proof is a direct corollary. ❑

7 Some Categories of Models for Concurrency Based on Graphs

Using duo-internal graphs, categories of Labeled Transition Systems, Petri Nets and Nonsequential Automata are easily defined and the verification of the existence of limits and colimits are straightforward (the proofs are omitted - they are direct corollary of previous results).

7.1 Labeled Transition Systems

A labeled transition system is basically a graph with an initial state and labeling on arcs. The corresponding category of duo-internal graph is bicomplete. However, while the coproduct construction can be interpreted as a choice between component systems, the product construction defines a kind of "total synchronization" with little practical applications. A more useful category can be obtained using reflexive graphs. Since labeling is restricted to arcs, the target object of a labeling morphism is a reflexive one node graph. In this case, a notion of "encapsulation" of transitions can be defined. In what follows, $Set^\bullet$ denotes the category of pointed sets (sets with a distinguished element) and pointed functions (the distinguished element is preserved by morphisms) which is bicomplete. In fact, $Set^\bullet$ is isomorphic to the category of one node reflexive graphs.

Definition 21 Consider:

a) The category of reflexive graphs $RGr(u^\bullet, id_{Set})$ such that $u^\bullet: Set^\bullet \to Set$ is the obvious forgetful functor;
b) The identity functor $shape: RGr(u^\bullet, id_{Set}) \to RGr(u^\bullet, id_{Set})$;
c) The "inclusion" functor $lab: Set^\bullet \to RGr(Set)$ where each pointed set L_τ is taken into the corresponding one node reflexive graph $\langle\{\bullet\}, L_\tau, !, !, \iota\rangle$ (isomorphic to L_τ) such that $\{\bullet\}$ is a pointed set, $!: u^\bullet L_\tau \to u^\bullet\{\bullet\}$ is the unique function and $\iota: u^\bullet\{\bullet\} \to u^\bullet L_\tau$ takes the unique element of $\{\bullet\}$ into the distinguished element $\tau \in L_\tau$.

Then, the *Category of Reflexive Labeled Transitions Systems* is the category of duo-internal graphs $LTS = LRGr(shape, lab)$. ❑

Therefore, the shape of a transitions system is a reflexive graph and labeling is over an one node reflexive graph (labeling on nodes is not required). A transition of the shape labeled by τ (the identity transition of the one node reflexive graph) can be considered as an encapsulated transition. Note that all identity transitions of the shape are labeled by τ (usually, an identity transition in a transition system means "no operation" and they are encapsulated). The initial state of a transitions system is the distinguished element of the point set of nodes and therefore, we did not have to use the notion of distinguished nodes defined for duo-internal graphs.

Proposition 22 The Category of Reflexive Labeled Transitions Systems LTS is bicomplete. ❑

Example 23 In LTS, the product and coproduct constructions can be interpreted as choice and parallel composition (all possible combination of component transitions - transition systems are sequential systems) as illustrated in Fig. 8 where an initial state is identified by an arrow without a source node. For simplicity, the label τ of the identity transitions is omitted. ❑

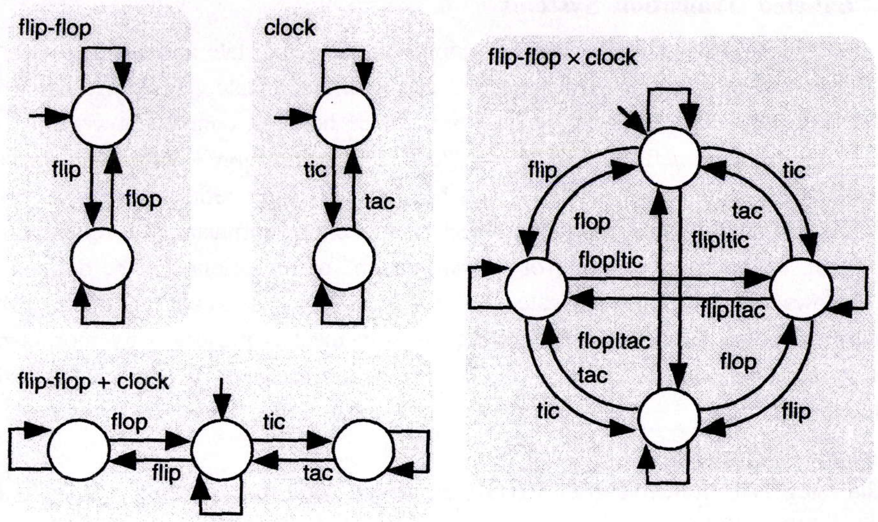

Fig. 8: LTS - coproduct and product

7.2 Petri Nets

To define a Petri net as a graph we follow the approach in (Meseguer & Montanari, 1990), where nodes are elements of a commutative monoid. In this case, nodes and arcs stand for states and transitions of a net, respectively, where for each transition, n tokens consumed or produced in a place A is represented by nA and n_i tokens consumed or produced simultaneously in a place A_i with i ranging over $1, \ldots, p$ is represented by $n_1A_1 \oplus n_2A_2 \oplus \ldots \oplus n_pA_p$ where $\oplus$ is the monoidal operation. Therefore, a Petri net is basically a graph with a monoidal structure on nodes and the corresponding category of duo-internal graph is bicomplete. However, while the coproduct construction can be interpreted as an asynchronous compositions of component systems, the product construction defines a kind of "total synchronization" with little practical applications (analogous to the non-reflexive labeled transitions systems). A more useful category can be obtained assuming that arcs are elements of a pointed set.

If an initial marking is added to the net structure, the resulting category do not have coproduct (Winskel, 1987). If an initial marking has at most one token in each place, the resulting category of nets have coproducts (Meseguer & Montanari, 1990). However the coproduct construction reflects a kind of "total choice" composition with restricted applications (Menezes & Costa, 1996). If a marked Petri net has a *set* of initial markings (the choice of which initial marking is considered at run time is a nondeterminism -

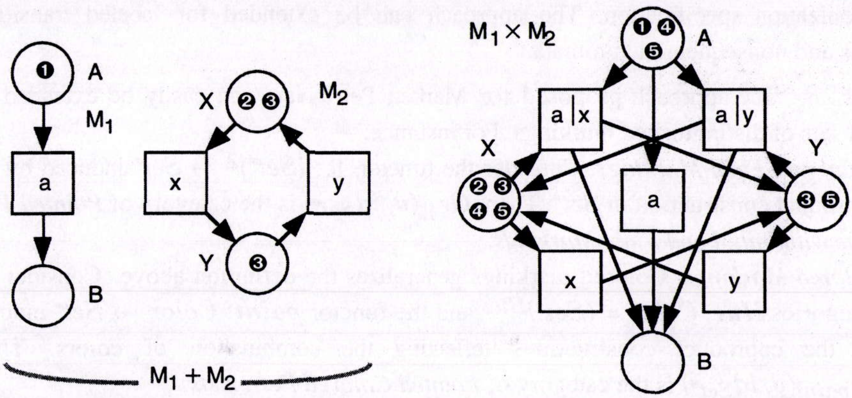

Fig. 9: Market Pointed Petri Nets - coproduct and product

(Menezes, 1995) then the resulting category is bicomplete and the coproduct construction reflects an asynchronous composition of nets.

Note that a Petri net has no labeling (in its standard definition). In what follows *CMon* denotes the category of commutative monoids which is bicomplete.

Definition 24

a) Consider the obvious forgetful functor v: $CMon \rightarrow Set^{\bullet}$ and the identity functor $id_{Set^{\bullet}}$: $Set^{\bullet} \rightarrow Set^{\bullet}$. The *Category of Pointed Petri Nets* is the category of duo-internal graphs $Petri = Gr(v, id_{Set^{\bullet}})$;

b) Consider the functor *nodes*: $Petri \rightarrow Set^{\bullet}$. The *Category of Marked Pointed Petri Nets* is the category of duo-internal graphs $MPetri = Gr_{id_{Set^{\bullet}}}(v, id_{Set^{\bullet}})$. ❑

Proposition 25 The categories *Petri* and *MPetri* are bicomplete. ❑

Example 26 Consider the Fig. 9 and the following symbols for tokens (and the corresponding initial markings):

❶ A ❷ X ❸ X⊕Y ❹ A⊕X ❺ A⊕X⊕Y

Then for the nets M_1 and M_2:

a) *Coproduct*. The resulting nets puts "side by side" the component nets with {A, X, X⊕Y} as the set of initial markings;

b) *Product*. The resulting nets reflects the parallel composition and has {A, X, X⊕Y, A⊕X, A⊕X⊕Y} as the set of initial markings. ❑

Remark 27 In (Menezes & Costa, 1996), a functorial operation for synchronization of nets is given, defined for transitions calling and sharing. It is defined using the fibration technique. The synchronization operation erases from the parallel composition (categorial product) of given pointed nets all those transition which do not reflect the given synchronization specification. The approach can be extended for labeled transitions systems and nonsequential automata. ❑

Remark 28 The approach proposed for Marked Petri Nets can easily be extended for several sets of distinguished markings. For instance:

a) *Initial and Final Markings*. Consider the functor $\amalg$: $(Set^\bullet)^2 \rightarrow Set^\bullet$ induced by the coproduct construction in $Set^\bullet$. Then, $Gr_{\amalg}(v, id_{Set^\bullet})$ is the category of *Pointed Petri Nets with Initial and Final Markings*;

b) *Colored Markings*. Colored markings generalizes the definition above. Consider the categories Hue, $Color = (Set^\bullet)^{Hue}$ and the functor $paint$: $Color \rightarrow Set^\bullet$ induced by the coproduct constructions reflecting the combination of colors. Then, $Gr_{paint}(v, id_{Set^\bullet})$ is the category of *Pointed Colored Petri Nets*. ❑

7.3 Nonsequential Automata

Nonsequential automata (Menezes & Costa, 1995) constitute a categorial semantic domain of processes around the concepts of sate and transition. It is inspired by (Meseguer & Montanari, 1990) and consists of a reflexive graph with monoidal structure on both, states and transitions, initial and final states and labeling on transitions. The interpretation of a structured state is the same as in Petri nets: it is viewed as a "bag" of local states representing a notion of tokens to be consumed or produced. A structured transition is a way to specify that the component transitions are independent, i.e., structured transitions specify which component transitions are concurrent of which as in (Bednarczyk, 1988) or (Mazurkiewicz, 1994). Nonsequential automata where introduced in order to achieve the diagonal compositionality requirement, i.e., reifications (implementations) compose and distribute over parallel composition. Reification is a special kind of net morphism were the target object is enriched with all conceivable sequential and concurrent computations.

Note that the following definition is analogous to the category LTS, replacing Set by $CMon$. Also, the approach for initial and final state are similar to the one introduced for Petri nets with initial and final markings.

Definition 29 Consider:

a) The category of reflexive graphs internal to $CMon$ where the distinguished nodes have a monoidal (commutative) structure $RGr_{\amalg}(CMon)$ such that $\amalg$: $CMon^2 \rightarrow CMon$ is the functor induced by the coproduct construction in $CMon$;

b) The obvious forgetful functor $shape$: $RGr_{\amalg}(CMon) \rightarrow RGr(CMon)$;

c) The "inclusion" functor lab: $CMon \rightarrow RGr(CMon)$ where each commutative monoid L_τ (τ is the unity) is taken into the corresponding one node reflexive graph internal to $CMon$ $\langle 1, L_\tau, !, !, \iota \rangle$ such that 1 is a fixed zero object (the only element of the support is the unity) and $!: L_\tau \rightarrow 1$, $\iota: 1 \rightarrow L_\tau$ are the unique morphisms.

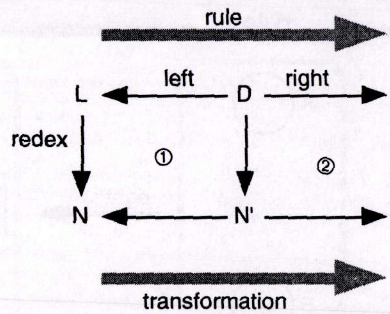

Fig. 10: Double pushout approach for transformation of graph-based systems

Then, the *Category of Nonsequential Automata* is the category of duo-internal graphs *NAut = LRGr(shape, lab)*. ❑

Proposition 30 The Category of Nonsequential Automata *NAut* is bicomplete. ❑

In (Menezes & Costa, 1995) it is shown that the category of Pointed Petri Nets is isomorphic to a (neither wide nor full) subcategory of Nonsequential Automata, i.e., Petri nets constitute a special case of nonsequential automata.

8 Anticipatory Systems

In (Menezes, 1999) we proposed an anticipation mechanism based on graph transformations, using the so called *single pushout* approach (Löwe, 1993) on a category of nets with *partial* morphisms. In this context, a graph transformation stands for a possible *system anticipation* (Dubois, 1998), (Rosen, 1985) In this paper we show a similar approach using the so called *double pushout* approach (Ehrig, 1979) extended for Petri nets viewed as graphs with *total* morphisms. The generalization for labeled transition systems and nonsequential automata can be easily obtained using the above results.

For the double pushout approach (see Fig. 10) consider the net N to be transformed, the rule ⟨left, right⟩ (a pair of morphisms where left usually is a monomorphism) which specifies how the transformation should be done and the morphism redex (usually a monomorphism) which instantiates the part to be replaced in the original net N. The pushout complement ① determines the net N' and then the pushout ② determines the transformed net M. The transformed net M is a *possible system anticipation* of N according to the given rule ⟨left, right⟩ and instantiation redex. Therefore, based on (Menezes 1999), the specification of an anticipatory system is a grammar Ant = ⟨R, N⟩ where R is a collections rules and N is a *MPetri*-object called initial net (assume that instantiations are all possible monomorphisms between corresponding nets). Note that a partial morphism r: L → R can be seen as pair of (total) morphisms ⟨left: D → L, right: D

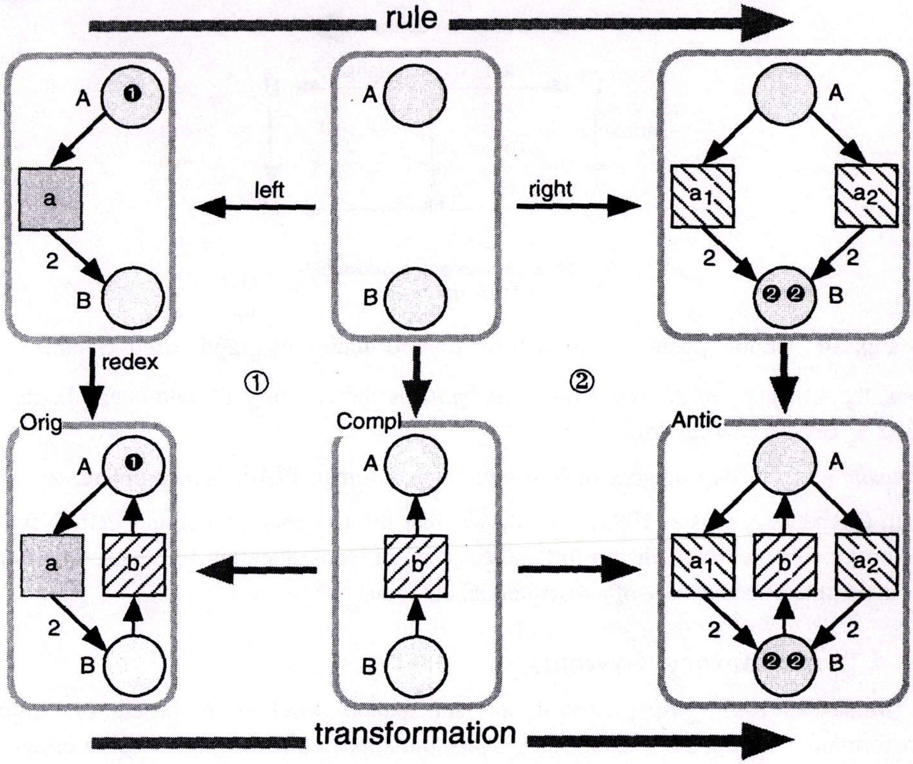

Fig. 11: Anticipation of a net

→ R⟩ where left is a monomorphism and D is the "domain" of r (in fact r is a class of equivalence of pair of morphisms). This view shows that the approach using the double pushout in this paper is very close to the single pushout with partial morphisms in (Menezes, 1999). However, they are not isomorphic (the discussion of this topic is not a goal of this paper).

Example 31 In the Fig. 11, consider the net Orig to be transformed, the rule ⟨left, right⟩ which specifies the replacement (the transition a is replaced by transitions a_1 and a_2, the corresponding source and target states are preserved, the marking A is forgotten and the marking 2B is introduced) and the monomorphism redex which instantiates the part to be replaced in the original net. The pushout complement ① determines the net Compl and then the pushout ② determines the transformed net Antic. This example illustrates a replacement of a transition (a possible system anticipation or system refinement) and also a replacement of a marking (also a possible system anticipation or modeling of a token game such as the firing of transitions). ❑

9 Conclusion

We construct a categorial framework for graph based systems. The concepts of duo-internal (reflexive/non-reflexive) graphs, with/without distinguished nodes and labeling on both arcs and nodes is proposed. The requirements for the existence of limits and colimits inherited from the component categories is set. In this context we show how bicomplete categories of Marked Petri Nets, Labeled Transitions Systems and Nonsequential Automata are defined. Also, we discuss the existence of colimits in categories of Petri nets equipped with initial markings (which is not usual) where the interpretation of coproducts and pushouts are adequate for giving semantics for concurrent, anticipatory systems. A double pushout approach for graph transformation is proposed as a mechanism for systems anticipations. In this case, the specification of an anticipatory system is a graph grammar extended for nets. Using the proposed duo-internal graph approach, the generalization of the graph transformation for other graph based systems such as labeled transitions systems or nonsequential automata are straightforward.

Currently we are developing a similar approach for categories (duo-internal categories), generalizing the notion of internal categories. Using adjunction between duo-internal graphs and duo-internal categories we will be able to express computations of graph based systems. Note that, enriching (reflexive) graphs with a special notion of transitive closure (able to express sequential and nonsequential computations) we get a category. Also, in this context, we will be able to express reifications (implementations) where a transition is mapped into a (possible complex) transaction. Therefore, a system reification is just a duo-internal graph morphism where the target graph is enriched with its computations.

In the near future, we plan to extend the notions of internalization and duo-internalization for partial graphs, generalizing previous work in (Menezes, 1999).

References

Adámek J., Herrlich H. & Strecker G. (1990). *Abstract and Concrete Categories*, Wiley.

Asperti A. & Longo G. (1991). *Categories, Types and Structures - An Introduction to the Working Computer Science*, Foundations of Computing (M. Garey, A. Meyer, Eds.), MIT Press.

Bednarczyk M. A. (1988). *Categories of Asynchronous Systems*, Ph.D. thesis, technical report 1/88, University of Sussex.

Casley R. T. (1991). *On the Specification of Concurrent Systems*, Ph.D. thesis, University of Stanford.

Corradini A. (1990). *An Algebraic Semantics for Transition Systems and Logic Programming*, Ph.D. thesis, technical report TD-8/90, Università di Pisa.

Degano P., De Nicola R. & Montanari U. (1988). *A Distributed Operational Semantics for CCS Based on Condition/Event Systems*, Acta Informatica 26, pp. 59-91.

Degano P. & Montanari U. (1987). *Concurrent Histories: A Basis for Observing Distributed Systems* J. Comput. System Sci. 34, pp. 422-462, Nos. 2/3.

Dubois D. M. (1996). *Introduction to the Aristotle's Final Causation in CAST: Concept and Method of Incursion and Hyperincursion*, EUROCAST'95, LNCS 1030, Springer-Verlag, pp. 477-493.

Dubois D. M. (1998). *Introduction to Computing Anticipatory Systems*, International Journal of Computing Anticipatory Systems Vol. 2, CHAOS, pp. 3-23.

Ehrig H. (1979). *Introduction to the Algebraic Theory of Graph Grammars*, 1st Graph Grammar Workshop (V. Claus, H. Ehrig, G. Rozenberg, Eds.), pp. 1-69, LNCS 93, Spring-Verlag.

van Glabbeek R. J. & Vaandrager F. W. (1987). *Petri Net Model for Algebraic Theories of Concurrency*, pp. 224-242, LNCS 259, Spring-Verlag.

Jonsson B. (1990). *A Hierarchy of Compositional Models of I/O-Automata*, Symposium on Mathematical Foundations of Computer Science, pp. 347-354, LNCS 452, Springer-Verlag.

Löwe M. (1993). *Algebraic Approach to Single-Pushout Graph Transformation*, TCS 109, pp. 181-224.

Mac Lane S. (1971). *Categories for the Working Mathematician*, Springer-Verlag, 1971.

Mazurkiewicz A. (1994). *Basic Notion of Trace Theory*, REX 88: Linear Time, Branching Time and Partial Orders in Logic and Models for Concurrency (J. W. de Bakker, W. -P. de Roever, G. Rozenberg, Eds.), pp. 285-363, LNCS 354, Springer-Verlag.

Menezes P. B. (1994). *Object Reification*, IS-CORE'94 - Workshop on Information System - Correctness and Reusability, Working Papers pp. 183-197, Rapportnr. IR-357, Vrije Universiteit, Amsterdan.

Menezes P. B. (1995). *Marked Petri Nets within a Categorial Framework*, RITA - Journal of Theoretical and Applied Informatics (in Portuguese, Revista de Informática Teórica e Aplicada), Vol. 2, No. 2, pp. 71-92, UFRGS/Brazil.

Menezes P. B. & Costa J. F. (1995). *Compositional Reification of Concurrent Systems*, Journal of the Brazilian Computer Society - Special Issue on Parallel Computation, No. 1, Vol. 2, pp. 50-67, SBC/Brazil.

Menezes P. B. & Costa J. F. (1996). *Synchronization in Petri Nets*, Fundamenta Informaticae Vol. 26.1, pp. 11-22, Annales Societatis Mathematicae Polonae, IOS Press.

Menezes P. B. (1997). *Reification of Concurrent Objects* (in Portuguese, *Reificação de Objetos Concorrentes*), 148pp, IST/UTL, Lisbon.

Menezes P. B., Sernadas A. & Costa J. F. (1998). *Nonsequential Automata Semantics for a Concurrent Object-Based Language*, First US-Brazil Workshop on Formal Foundations of Software Systems, Rio de Janeiro, Electronic Notes in Theoretical Computer Science 14, 29 pp., URL http://www.elsevier.nl/locate/entcs/volume14.html.

Menezes P. B. (1998). *Compositional Reification of Petri Nets Trough Graph Transformation*, Second US-Brazil Workshop on Formal Foundations of Software Systems, New Orleans, Electronic Notes in Theoretical Computer Science 14, URL http://www.elsevier.nl/locate/entcs/volume14.html, 27 pp.

Menezes P. B. (1999). *A Categorial Framework for Concurrent, Anticipatory Systems*, CASYS'98 - The Second International Conference on Computing Anticipatory Systems, American Institute of Physics, AIP Conference Proceedings Vol. 465, pp. 185-199, New York.

Meseguer J. & Montanari U. (1990). *Petri Nets are Monoids*, Information and Computation 88, pp. 105-155, Academic Press.

Milner R. (1989). *Communication and Concurrency*, Prentice Hall.

Olderog E. R. (1987). *Operational Petri Net Semantics for CSP*, Advances in Petri Nets 1987 (G. Rozenberg, Ed), pp. 196-223, Springer-Verlag.

Reisig W. (1985). *Petri Nets: An Introduction*, EATCS Monographs on Theoretical Computer Science 4, Springer-Verlag.

Rosen R. (1985). *Anticipatory Systems*, Pergamon Press.

Sassone V., Nielsen M. & Winskel G. (1993). *A Classification of Models for Concurrency*, CONCUR 93: Fourth International Conference of Concurrency (E. Best, Ed.), pp. 82-96, LNCS 715, Springer-Verlag.

Winskel G. (1984). *Categories of Models for Concurrency*, LNCS 197, Seminar on Concurrency (S. Brookes, Roscoe, G. Winskel, Eds.), pp. 246-267, Springer-Verlag.

Winskel G. (1987). *Petri Nets, Algebras, Morphisms and Compositionality*, Information and Computation 72, pp. 197-238, Academic Press.