Equations for Asynchronous Message Passing

Ludwik Czaja[1,2]
[1] Institute of Informatics, The University of Warsaw
[2] University of Economics and Computer Science Vistula in Warsaw
lczaja@mimuw.edu.pl

1 Introduction

In [Cza 2006] fix-point equations specifying synchronous ("hand-shaking") communication in distributed systems have been proposed. Their solution yielded a communication network of agents, directly presented as a Petri net-like structure, and determined the global state of the specified system. The net-places represented agents, while transitions - transfer of messages. A special algebra being a semi-ring with "addition" (nondeterministic choice) and "multiplication" (simultaneity) was a formal basis for the equations and their solving procedure. Here, the equations are modified to specify asynchronous communication, that is, such that the senders, after sending message, continue their performance without waiting for reception. This required introducing a new type of objects called buffers or mailboxes - apart from the agents (senders/receivers), and changing the semi-ring into a distributive lattice of the agents and mailboxes. In the asynchronous communication, solution to the fix-point equations should determine that: (1) the sender *can* send message as soon as mailboxes of its simultaneous receivers *can* store the message, no matter whether the receivers are ready to get it or not. (2) in the resulting net, the mailboxes are included as net-places too, each one collecting messages from its senders (possibly a number of senders) and transferring them to its (exactly one) receiver for which it is the unique mailbox. The proposed modelling of communication takes some (but only some!) ideas from CSP [Hoa 1978, 1985], CCS [Mil 1980, 1989] (e.g. a concept of agents, ports, synchronization between senders and mailboxes or its absence: no synchronization between senders and receivers, communication media or channels), Petri nets [Rei 1985] (e.g. graphical presentation of solution to the communication equations) or practice of computer networks and distributed systems [C-D-K 2005] (e.g. multicasting and broadcasting,

blocking/non-blocking, connection-oriented/connectionless communication mode). The communication equations may be treated as specifying interconnection system among agents, while net-structures resulted from their solution - as "implementation" of this system. The correctness of such implementation is ensured by identity of each group of receivers from a given sender (and, symmetrically, of each group of senders to a given receiver), both in specification and implementation.

2 Two algebraic systems

In the following two algebras will be needed:

(1) A semiring of terms

Let $\mathbb{X}$ be a set of atomic arguments partitioned into the set $\mathbb{C}$ of *constants* and the set $\mathbb{V}$ of *variables*: $\mathbb{X} = \mathbb{C} \cup \mathbb{V}$, $\mathbb{C} \cap \mathbb{V} = \emptyset$. Each $x \in \mathbb{X}$ is a term; if K and L are terms then strings $(K + L)$ and $(K \bullet L)$ are too, called a sum and product. We say "*terms over* $\mathbb{X}$", their set $\mathbb{T}[\mathbb{X}]$. Addition and multiplication of terms is defined as follows: $K \oplus L = (K + L)$, $K \otimes L = (K \bullet L)$, but $+$ and $\bullet$ are used for $\oplus$ and $\otimes$. It is required that the system $(\mathbb{T}[\mathbb{X}], +, \bullet)$ obeys the following axioms for all $K, L, M \in \mathbb{T}[\mathbb{X}]$:

(i) $K + K = K$
(ii) $K + L = L + K$
(iii) $K + (L + M) = (K + L) + M$
(iv) $K \bullet L = L \bullet K$
(v) $K \bullet (L \bullet M) = (K \bullet L) \bullet M$
(vi) $K \bullet (L + M) = K \bullet L + K \bullet M$
(vii) $K \in \mathbb{X} \Rightarrow K \bullet K = K$

By assuming that $\bullet$ binds stronger than $+$ and by (iii) and (v), some parentheses may be dropped. Two terms are equal $(=)$ iff one may be transformed into the other by means of the axioms. A partial order $(\sqsubseteq)$ between terms is defined as $K \sqsubseteq L$ iff $L = L + K$. If a term K is composed of atoms $x, y, ..., z$ at the most, we write $K(x, y, ..., z)$. That is, some of them may be absent in K. A *power* of K is defined by $K^1 = K$, $K^{n+1} = K \bullet K^n$. This algebra enjoys the properties:
(a) Each term may be transformed by means of equations *(i)*$-$*(vii)* to a sum of products of arguments and then simplified. Such form is called *canonical*
(b) If $n \leq m$ then $K^n \sqsubseteq K^m$.

(2) A distributive lattice of terms over constants

For the lattice the same operators + and $\bullet$ are used, thus, the algebraic system $(\mathbb{T}[\mathbb{C}], +, \bullet)$ of terms with constants only is required to obey axioms $(i) - (vii)$ and additionally for $K, L \in \mathbb{T}[\mathbb{C}]$:

(viii) $K \bullet L + K = K$

This algebra enjoys the following:
(c) If $K = K_1 + K_2 + ... + K_n$ where $K_i \in \mathbb{T}[\mathbb{X}]$, is a product of arguments, then $K^n = K^{n+j}$ for $j \geq 0$. In particular, for any term K there exists a number n such that $K^n = K^{n+j}$ for $j \geq 0$. Define a *maximal power* of K as $p(K) = min(n|\ \forall j \geq 0 : K^n = K^{n+j})$. For instance, the maximal power of $a \bullet x + b$ with $a, b \in \mathbb{C}$, $x \in \mathbb{V}$ is 2 since $(a \bullet x + b)^2 = a \bullet x + a \bullet b \bullet x + b$, $(b \bullet a + c)^2 = (b \bullet a + c)^{2+j}$ for $j \geq 0$, and $a \bullet x + b \neq (a \bullet x + b)^2$. Note that if $K \in \mathbb{T}[\mathbb{C}]$ then $p(K) = 1$.
(d) There exist solutions to equation $x = K \bullet x + L$ and their general form is $x = K^{p(K)} \bullet (L + P) + L$, where $p(K)$ is the *maximal power* of K and P is arbitrary term not containing x. In particular, general form of solution to equation $x = K \bullet x$ is $x = K^{p(K)} \bullet P$.

Interpretations: In Section 3 constants will play part of agents represented by places in nets obtained as solutions to the specification equations, while some products of the constants - transitions meant as actions of message transfer. In Section 4, apart for aforesaid role of constants, some of them will also play part of mailboxes. These products will be monomials in terms, being values of unknowns (send and receive variables) in the specification equations presented in the following two sections.

3 Equations of specification - synchronous mode

Synchronous mode of communication (in a fairly abstract setting) has been introduced in [Hoa 1978], in extended form presented in [Hoa 1985] and implemented in some programming languages, e.g. [OCCAM 1984]. To formulate specification equations for such mode, suppose that with each agent-constant $c \in \mathbb{C}$ is bijectively associated:

- its co-agent denoted $\overline{c}$ and belonging to $\mathbb{C}$
- its send-variable denoted $c!$ and belonging to $\mathbb{V}$
- its receive-variable denoted $c?$ and belonging to $\mathbb{V}$

The send and receive variables will get values being solutions to equations set up as follows. For $i \leq n$, $j \leq m$, $b_j, a_i \in \mathbb{C}$ let $K_i(b_1?, ..., b_m?)$ be a term composed only of receive-variables $b_j?$ and $L_j(a_1!, ..., a_n!)$ - a term composed only of send-variables $a_i!$. The specification equations are of the form:

$$a_i! = a_i \bullet K_i(b_1?, ..., b_m?) \qquad \text{(agent } a_i \text{ sends to } b_1, ..., b_m) \qquad (1)$$

$$b_j? = \overline{b_j} \bullet L_j(a_1!, ..., a_n!) \qquad \text{(agent } b_j \text{ gets from } a_1, ..., a_n) \qquad (2)$$

with the requirement:
$b_j?$ occurs in $K_i(b_1?, ..., b_m?)$ iff $a_i!$ occurs in $L_j(a_1!, ..., a_n!)$

The requirement is a consistency property of the equations. Variables $a_i!$, $b_j?$ are unknown quantities, while constants a_i, $\overline{b_j}$ are known. A *region of receivers* from the agent-sender a_i is a subset of the set $\{b_1, ..., b_m\}$ of receivers to which a_i sends a message simultaneously. Symmetrically, a *region of senders* to receiver b_j is a subset of the set $\{a_1, ..., a_n\}$ of senders from which b_j receives a message simultaneously. Simultaneous message transfer ("multicast message") is represented by product "$\bullet$" of agents. The constants and variables will also be interpreted as propositional functions of state s of the distributed system such that a_i is *ready to send* a message in a state s iff $a_i(s)$ is *true* and it *can send* message iff $a_i!(s)$ is *true*. Similarly, b_j is *ready to receive* a message in a state s iff $\overline{b_j}(s)$ is *true* and it *can receive* message iff $b_j?(s)$ is *true*. Solutions to equations (1), (2) yield in particular net structures, but also a *global* information on *capability* of sending/receiving messages. The information is collected from a *local* information of agents' *readiness* to do this. Existence of the solutions to these equations follows from the general fact concerning equations for formal power series (their finite version) considered in [Sa-So 1978].

Remarks and intuitions

(1) The specification equations formalize the following (recursive) phrases:

- "an agent a *can* send a message if it is ready to do this and each agent in a certain region of its receivers *can* receive this message from a"
- "an agent b *can* receive a message if it is *ready* to do this and each agent in a certain region of its senders *can* send this message to b"

(2) Readiness of an agent to send/receive a message is determined by its internal control pointing to a send/receive instruction within the agent's program.

(3) In semiring defined by axioms *(i)-(vii)* in Section 2, but without *(viii)*, that make it a lattice, the specification equations may have more than one solution. A solution which reflects the communication structure specified by equations (1), (2) is the one where each sender (resp. receiver) has the same regions of receivers (resp. senders) as specified in these equations. Such solution is unique and in [Cza 2006] called "appropriate". Note that appropriate solution is, in fact, the "correct" one if "correctness" is understood as conformity with specification. In this perspective, solving the equations means implementing the specified interrelation of communicating agents in terms of net-structures, along with ensuring correctness of the implementation.

Example 1

Synchronous communication (specification):

$a! = a \bullet (x? \bullet y? + y?)$ (a sends simultaneously to x and y or only to y)

$b! = b \bullet (y? \bullet z? + y?)$ (b sends simultaneously to y and z or only to y)

$x? = \overline{x} \bullet a!$ (x gets from a)

$y? = \overline{y} \bullet (a! + a! \bullet b! + b!)$ (y gets only from a or simultaneously from a and b or only from b)

$z? = \overline{z} \bullet b!$ (z gets from b)

A solution; products represent transitions in the net representation - Fig.1:

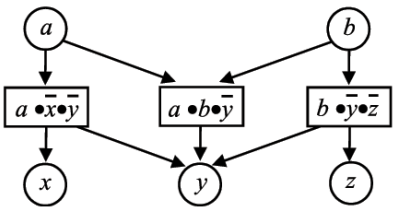

Fig.1

$a! = a \bullet \overline{x} \bullet \overline{y} + a \bullet b \bullet \overline{y}$

$b! = b \bullet \overline{y} \bullet \overline{z} + a \bullet b \bullet \overline{y}$

$x? = a \bullet \overline{x} \bullet \overline{y}$

$y? = a \bullet \overline{x} \bullet \overline{y} + a \bullet b \bullet \overline{y} + b \bullet \overline{y} \bullet \overline{z}$

$z? = b \bullet \overline{y} \bullet \overline{z}$

There are two regions of receivers from agent a: $\{x, y\}, \{y\}$, two regions of receivers from agent b: $\{y, z\}, \{y\}$, as well as one region of senders to agent x: $\{a\}$, one region of senders to agent z: $\{b\}$, and three regions of senders to agent y: $\{a\}, \{b\}, \{a, b\}$. The solution is "appropriate" since the regions of senders and receicers are the same as in the specification equations.

Agents as propositional functions of state

A state of the communication structure specified by equations (1), (2) is a function $s : \mathbb{C} \to \mathbb{D}$ where $\mathbb{D}$ is a set of data the agents intercommunicate. In particular, if a solution is presented as a place/transition net then $\mathbb{D} = \mathbb{N}$ (natural numbers). With each agent a propositional function (denoted by the same symbol as the agent and called a *readiness condition*) of state is associated:

For $a \in \mathbb{C}$: $a(s) = \begin{cases} \mathit{true} & \text{if } a \text{ is ready to send a message in state } s \\ \mathit{false} & \text{otherwise} \end{cases}$

For $\overline{a} \in \mathbb{C}$: $\overline{a}(s) = \begin{cases} \mathit{true} & \text{if } a \text{ is ready to receive a message in state } s \\ \mathit{false} & \text{otherwise} \end{cases}$

The functions are extended to terms over the set of agents: for $K, L \in \mathbb{T}[\mathbb{C}]$:

$(K + L)(s) = K(s) \vee L(s)$

$(K \bullet L)(s) = K(s) \wedge L(s)$

Thus, solutions $a_i! = K$ and $b_j? = L$ to the equations (1), (2) become propositional functions of state, defined by $a_i!(s) = K(s)$ and $b_j?(s) = L(s)$. This justifies phrases "sender" and "receiver", since agent a *can* send (receive) a message, in a state s iff $a!(s) = \mathit{true}$ $(a?(s) = \mathit{true})$.

Products of agents (monomials in terms) as state transformers

The products in solutions to equations (1), (2) play part of *state transformers*. This interpretation is more abstract than interpretation of transitions in Petri nets because of more general meaning of state ("marking") and its transformation. Let $t_1 + ... + t_p$ be a solution (in canonical form) to $a!$ or $a?$, that is $t_i = (a_1 \bullet ... \bullet a_{q_i}) \bullet (\overline{b_1} \bullet ... \bullet \overline{b_{r_i}})$ with $a_1, ..., a_{q_i}, \ \overline{b_1}, ..., \overline{b_{r_i}} \in \mathbb{C}$. The product t_i will represent a transition with preset $a_1, ..., a_{q_i}$ and postset $b_1, ..., b_{r_i}$. Its firing rule (semantics) is given by a relation $[[t_i]] \subseteq \mathbb{S} \times \mathbb{S}$ ($\mathbb{S} = \mathbb{D}^{\mathbb{C}}$ is the set of all states) satisfying: if $(s, s') \in [[t_i]]$ then

(loc) $\quad s|(\mathbb{C} \backslash {}^\bullet t_i^\bullet) = s'|(\mathbb{C} \backslash {}^\bullet t_i^\bullet)$ (locality condition)

(fir) $\quad a_1(s) \wedge ... \wedge a_{q_i}(s) \wedge \overline{b_1}(s) \wedge ... \wedge \overline{b_{r_i}}(s)$ is true (firability condition)

where ${}^\bullet t_i^\bullet = \{a_1, ..., a_{q_i}, b_1, ..., b_{r_i}\}$ ("neighbourhood" of t_i) and | - "restriction to...". Consequently, by $[[a!]]$ ($[[a?]]$ respectively) is denoted the relation $[[t_1]] \cup ... \cup [[t_p]]$. We say that t_i transforms s into s'. Note that the condition (fir) may be written "$t_i(s)$ is $true$".

Solutions to equations (1), (2) as net structures

As shown in Example 1, a solution to (1), (2) is directly mapped onto a net structure with places representing agents and transitions - some of their products. This net-structure is, in fact, identical with a set of its transitions defined above as $t_i = (a_1 \bullet ... \bullet a_{q_i}) \bullet (\overline{b_1} \bullet ... \bullet \overline{b_{r_i}})$. Such a net-structure exhibits admissible communication network specified by the equations. Fixing the set $\mathbb{D}$ of values, the relations $[[t_i]]$ and the readiness conditions of agents, various classes of nets are obtained. For instance, in [Cza 2006], P/T and coloured nets (with and without capacity of places and arrow weights) are generated this way. The synchronous (handshaking [Hoa 1985], [Mil 1989], [OCCAM 1984] or blocking [C-D-K 2005]) mode of communication is determined by the form of terms being solutions to specification equations. Transitions t_i as state transformers, are interpreted as actions producing messages for agents $b_1, ..., b_{r_i}$ out of messages delivered by agents $a_1, ..., a_{q_i}$. Such abstract setting allows to think of transitions as transformers of various types of data [Cza 2007]. Note that t_i may also be thought of as an *exit* (not entry!) junction-box (or port) shared by agents $a_1, ..., a_{q_i}$ and *entry* junction-box shared by agents $b_1, ..., b_{r_i}$. From the perspective of programming practice, a transition acts somewhat like a shared procedure statement taking data from agents $a_1, ..., a_{q_i}$ and forwarding results to $b_1, ..., b_{r_i}$ with no side-effects (because of the condition (loc)). Note that semantics of a net-structure $\mathcal{N}$, which is a set of transitions, may be defined as $[[\mathcal{N}]] = \bigcup_{t \in \mathcal{N}} [[t]]$

4 From synchronous to asynchronous mode

In contrast to synchronized communication, in the asynchronous mode (implemented e.g. in Linda [C-G 1989]), the senders dispatch messages to receivers through mailboxes of the latter. A sender may send message to a number of mailboxes and each mailbox may get message from a number of senders. However each receiver has exactly one mailbox and each mailbox has exactly one receiver. To formulate the respective equations suppose that with each agent $a \in \mathbb{C}$ is bijectively associated its mailbox denoted $\langle a \rangle$ and with $\langle a \rangle$ is bijectively associated:

- its co-mailbox $\overline{\langle a \rangle}$

- its send-variable denoted $\langle a\rangle!$
- its receive-variable denoted $\langle a\rangle?$

Now, the specification equations for the asynchronous mode of communication are:

$a_i! = a_i \bullet K_i(\langle b_1\rangle?, ..., \langle b_m\rangle?)$ (senders send to mailboxes of receivers) (3)

$\langle b_j\rangle? = \overline{\langle b_j\rangle} \bullet L_j(a_1!, ..., a_n!)$ (mailboxes of receivers get from senders) (4)

$\langle b_j\rangle! = \langle b_j\rangle \bullet b_j?$ (mailboxes of receivers send to the latter) (5)

$b_j? = \overline{b_j} \bullet \langle b_j\rangle!$ (receivers get from their mailboxes) (6)

with the requirement:

$\langle b_j\rangle?$ occurs in $K_i(\langle b_1\rangle?, ..., \langle b_m\rangle?)$ iff $a_i!$ occurs in $L_j(a_1!, ..., a_n!)$

Note that equations (5), (6) representing delivery of messages to receivers from their mailboxes have straightforward solutions:

$\langle b_j\rangle! = \langle b_j\rangle \bullet b_j$

$b_j? = \overline{b_j} \bullet \langle b_j\rangle$

Equations (3), (4) are solved like those in (1), (2) in Section 3. The senders' capability of sending message depends here on capability of reception by mailboxes only. The readiness of a mailbox to get messages from senders means that it is not filled up, while its readiness to deliver a message to its receiver means that it is not empty. Therefore we consider:

Mailboxes as propositional functions of state

For $\langle b\rangle \in \mathbb{C}$: $\langle b\rangle(s) = \begin{cases} \mathit{true} & \text{if } \langle b\rangle \text{ is ready to deliver a message in state } s \\ \mathit{false} & \text{otherwise} \end{cases}$

For $\overline{\langle b\rangle} \in \mathbb{C}$: $\overline{\langle b\rangle}(s) = \begin{cases} \mathit{true} & \text{if } \langle b\rangle \text{ is ready to get a message in state } s \\ \mathit{false} & \text{otherwise} \end{cases}$

The functions are extended to terms as formerly in the case of agents.

Example 2

Asynchronous communication (specification):

$a! = a \bullet (\langle x\rangle? \bullet \langle y\rangle? + \langle y\rangle?)$ (a sends simultaneously to mailboxes $\langle x\rangle$ and $\langle y\rangle$ or only to $\langle y\rangle$)

$b! = b \bullet (\langle y\rangle? \bullet \langle z\rangle? + \langle y\rangle?)$ (b sends simultaneously to mailboxes $\langle y\rangle$ and $\langle z\rangle$ or only to $\langle y\rangle$)

$\langle x\rangle? = \overline{\langle x\rangle} \bullet a!$	($\langle x\rangle$ gets from a)
$\langle y\rangle? = \overline{\langle y\rangle} \bullet (a! + a! \bullet b! + b!)$	($\langle y\rangle$ gets only from a or simultaneously from a and b or only from b)
$\langle z\rangle? = \overline{\langle z\rangle} \bullet b!$	($\langle z\rangle$ gets from b)
$\langle x\rangle! = \langle x\rangle \bullet x?$	($\langle x\rangle$ delivers to x)
$\langle y\rangle! = \langle y\rangle \bullet y?$	($\langle y\rangle$ delivers to y)
$\langle z\rangle! = \langle z\rangle \bullet z?$	($\langle z\rangle$ delivers to z)
$x? = \overline{x} \bullet \langle x\rangle!$	(x gets from $\langle x\rangle$)
$y? = \overline{y} \bullet \langle y\rangle!$	(y gets from $\langle y\rangle$)
$z? = \overline{z} \bullet \langle z\rangle!$	(z gets from $\langle z\rangle$)

A solution:

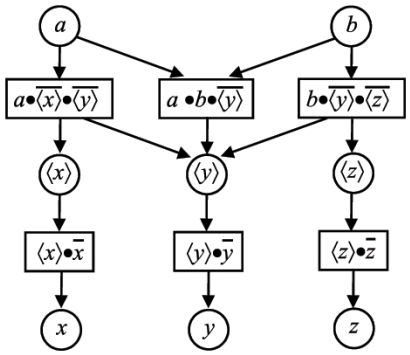

Fig.2

$a! = a \bullet \overline{\langle x\rangle} \bullet \overline{\langle y\rangle} + a \bullet b \bullet \overline{\langle y\rangle}$

$b! = b \bullet \overline{\langle y\rangle} \bullet \overline{\langle z\rangle} + a \bullet b \bullet \overline{\langle y\rangle}$

$\langle x\rangle? = a \bullet \overline{\langle x\rangle} \bullet \overline{\langle y\rangle}$

$\langle y\rangle? = a \bullet \overline{\langle x\rangle} \bullet \overline{\langle y\rangle} + a \bullet b \bullet \overline{\langle y\rangle} + b \bullet \overline{\langle y\rangle} \bullet \overline{\langle z\rangle}$

$\langle z\rangle? = b \bullet \overline{\langle y\rangle} \bullet \overline{\langle z\rangle}$

$\langle x\rangle! = \langle x\rangle \bullet \overline{x}$

$\langle y\rangle! = \langle y\rangle \bullet \overline{y}$

$\langle z\rangle! = \langle z\rangle \bullet \overline{z}$

$x? = \overline{x} \bullet \langle x\rangle$

$y? = \overline{y} \bullet \langle y\rangle$

$z? = \overline{z} \bullet \langle z\rangle$

The solution exhibits directly data channels (links) of the agents' communication and their capability to transfer the data in a given state. For instance, agent y *can* receive a message from its mailbox $\langle y\rangle$ in a state s iff $y?(s) = \overline{y}(s) \wedge \langle y\rangle(s)$ is $true$, meaning that the control in y reached its receive statement (instruction) and its mailbox is not empty. The mailbox $\langle y\rangle$ in a state s *can* receive a message either from a only or from b only, or one message from a and the other from b simultaneously iff

$$\langle y\rangle?(s) = a(s) \wedge \overline{\langle x\rangle}(s) \wedge \overline{\langle y\rangle}(s) \vee a(s) \wedge b(s) \wedge \overline{\langle y\rangle}(s) \vee b(s) \wedge \overline{\langle y\rangle}(s) \wedge \overline{\langle z\rangle}(s)$$

is $true$, meaning that the control in a reached its send statement and neither of the mailboxes $\langle x\rangle, \langle y\rangle$ is filled up or the controls in a and b reached their send statements and the maibox $\langle y\rangle$ is not filled up or the control in b reached its send statement and neither of the mailboxes $\langle y\rangle, \langle z\rangle$ is filled up.

5 Final notes

(1) The role of variables $a_i!, b_j?$ is similar to that of "meta variables" (non-terminals) in BNF: they assume terms, that is strings of constants (terminals), as values. Also the role of constants is similar: they assume elements of $\mathbb{D}$, the set of data, as values.

(2) Solutions to specification equations for synchronous (blocking) and asynchronous (non-blocking) mode, describe a kind of intercommunication where communication lines between the agents are fixed for a given computational session. The lines are pictorialy visible as arcs of nets representing the solutions. The interesting issue is to adjust the idea to systems where the lines are being established dynamicaly - during the session. To this end, the idea of self-modifying nets, introduced by Ruediger Valk [Val 1978], [Val

1981], or perhaps, stratified nets [B-D 1997], seems promissing. Also, modelling of the so-called connection oriented (sender and receiver set up a line and make an arrangement on some details of communication prior to the communication itself; the line remains as long as it is needed during communication) and connectionless communication (e.g. [C-D-K 2005]) in the framework pursued here is a research challenge, close to practical application. Another approach to self-modification is based on the abovementioned view of a net-structure as a collection of transitions, each being a pair of two sets of agents (places). Firing transition would, then, encompass not only change of marking but also modification of the set of transitions that constitutes the net-structure (motivation: users of distributed systems undertake send/receive actions independently of any external control). This is a work in progress.

6 References

[B-D 1997] Badouel E., Darondeau P.: *Stratified Petri Nets*, FCT'97, Lecture Notes in Computer Science vol. 1279 (1979), pp. 117-128

[C-G 1989] Carriero N, Gelernter D.: *Linda in Context*, Comm. of the ACM, 1989

[C-D-K 2005] Coulouris G, Dollimore J, Kindberg T: *Distributed Systems. Concepts and Design.* Addison Wesley, Editions: 1988, 1994, 1995, 1996, 2005.

[Cza 2006] Czaja L.: *Equations for Message Passing*, Fundamenta Informaticae, Vol. 72, Numbers, 1-3, July/August 2006, pp. 81-93

[Cza 2007] Czaja L.: *Interpreted Nets*, Fundamenta Informaticae, Vol.79, Numbers 3-4, September 2007, pp.283-293

[Hoa 1978] Hoare C.A.R.: *Communicating Sequential Processes*, Comm. of the ACM, Vol 21, pp. 666-677, 1978

[Hoa 1985] Hoare C.A.R.: *Communicating Sequential Processes*, Prentice-Hall, 1985

[Mil 1980] Milner A.J.R.G.: *A Calculus of Communicating Systems*, Lecture Notes in Computer Science vol. 92 (1980)

[Mil 1989] Milner A.J.R.G.: *A Calculus of Communicating Systems*, Prentice-Hall, 1989

[OCCAM 1984] INMOS Limited: *OCCAM Programming Manual*, Prentice-Hall, 1984

[Rei 1985] Reisig W.: *Petri Nets, An Introduction*, EATC Monographs on Theoretical Computer Science, Springer-Verlag, 1985

[Sa-So 1978] Salomaa A., Soittola M., *Automata-Theoretic Aspects of Formal Power Series*, Springer, 1978

[Val 1978] Valk R.: *Self-Modifying Nets, a Natural Extension of Petri Nets*, Icalp'78, Lecture Notes in Computer Science vol. 62 (1978), pp. 464-476

[Val 1981] Valk R.: *Generalization of Petri Nets*, MFCS'81 Lecture Notes in Computer Science vol. 118 (1981), pp. 140-155