On Languages of Channels for Communicating ODP Engineering Objects

El Maati Chabbar, Mohamed Bouhdadi

***Abstract*—The Reference Model for Open Distributed Processing (RM-ODP) defines a framework within which support of distribution, interoperability and portability can be integrated. An ODP system is defined in terms of five viewpoints. The ODP engineering specification consists of a set of engineering objects which communicate via a channel object. The engineering viewpoint defines the ODP transparencies and ODP functions. We focus in this paper on the language of the channel engineering object. We associate to each component state of a global state a set of words that may be contained in channels. We define, for each object, a grammar 'like' context free in which, each rule is of the form $X \rightarrow u^{-1}Yv$, where $u^{-1}Yv$ stand for the residual of the language (L(Y)v) with regard to u. We use context-free grammar properties to make transformations and appear a symbol X in the right member of each X-production to express loop and cycle transitions in the CFSM. The algebraic property of context-free languages is then used to calculate these languages which are minimal solution of a system of equations. These languages can be used to verify some protocol properties such as reachability and deadlock problems.**

Index Terms— **RM-ODP, Engineering Language, Channel Object, Context free languages, Residual languages.**

I. Introduction

The rapid growth of distributed processing has led to a need for coordinating framework for the standardization of Open Distributed Processing (ODP). The Reference Model for Open Distributed Processing (RM-ODP) [1-4] provides a framework within which support of distribution, networking and portability can be integrated. The foundations part [2] contains the definition of the concepts and analytical framework for normalized description of (arbitrary) distributed processing systems. These concepts are grouped in several categories. The architecture part [3] contains the specifications of the required characteristics that qualify distributed processing as open. It defines a framework comprising five viewpoints, viewpoint language, ODP functions and ODP transparencies.

El maati Chabbar, University Mohammed V, chabbar@fsr.ac.ma
Mohamed Bouhdadu, Department of Mathematics & Computer Science, University Mohammed V Rabat, Morocco, email: bouhdadi@ fsr.ac.ma).

The five viewpoints, called enterprise, information, computational, engineering and technology provide a basis for the specification of ODP systems.

Each viewpoint language defines concepts and rules for specifying ODP systems from the corresponding viewpoint. The first three viewpoints do not take into account the distribution and heterogeneity inherent problems. This corresponds closely to the concepts of PIM (Plat-form Independent Model) and PSM (Plat-form Independent Model) models in the OMG MDA architecture. That is, it is the engineering viewpoint which takes into account all the inherent problems to distribution and openness. The ODP functions are required to support ODP systems. The transparency prescriptions show how to use the ODP functions to achieve distribution transparency.

However, RM-ODP only provides a framework for the definition of new ODP standards. These standards include standards for ODP functions; standards for modelling and specifying ODP systems; standards for programming, implementing, and testing ODP systems.

In previous works [5, 6, 7], we have treated the need of formal notation of ODP viewpoint languages. Indeed, the viewpoint languages are abstract in the sense that they define what concepts should be supported, not how these concepts should be represented. It is important to note that, RM-ODP uses the term language in its broadest sense:" a set of terms and rules for the construction of statements from the terms;" it does not propose any notation for supporting the viewpoint languages. In this work we focus on the ODP engineering Channel Object Language. Our objective in this paper is apply a theoretical framework for calculating the set of words that can be contented in the FIFO channel objects [8], in the context of RM-ODP.

Ideed, FIFO systems consist of sequential processes that communicate via unbounded channels; each process can be modeled by a Finite State Machine (FSM). Communicating Finite State Machines (CFSM) can be seen as a system of FSM's equipped with a collection of FIFO channels, and a system of CFSM can be represented by automata [9, 10].
This model is used to verify some properties such as safety which can be reduced to reachability problem. The reachability set is the set of all reachable states and it's the purpose of reachability analysis. Computing this set is not reasonable especially if it's infinite. This problem is, in general, undecidable [10] due to the explosion of state space.

ISBN: 978-988-98671-8-8

For the verification of such system and depending on the transition relation of the automata, several approaches have been developed :

The reduction of the global state space using partial order reduction [11, 12, 13].

The well structured transition system in the case where there exists a well quasi- ordering over the set of states [14, 15, 16].

The regular model checking where states are represented by words and the transition relation is regular expression [17, 18, 19]

Some searchers are interested by languages of FIFO channels; the main goal is to have a class of systems for which the limit language of queues is regular and then, it can be analyzable. Pachl [20] introduced assertions in form of recognizable description of content of channels, and proved that if the language of the channels is known then the reachability question is decidable. Boigelot and al. [21] considered individual operations on channel language as well as the iterated operation for loops and proposed heuristic for calculating limit set. Trefler et a. [22, 23] studied the class of piecewise languages, which are those recognized by nondeterministic automata whose only nontrivial strongly connected components are states with self loops, and proved that this class of FIFO channel systems is analyzable. Le Gall et al. [24] extended the model to an abstract lattice for which any increasing chain, using a widening operator, stabilizes after a finite number of steps; the notion of widening operator is to guess the limit of fix-point computation. The approximation, in this framework, can be not exact; in this sense a finite language can be approximated by an infinite one.

Generally speaking, there was no method for calculating limit languages for any FIFO channel systems.

We focus in this paper on FIFO channel languages of Communicating Distributed Engineering Objects. Our objective in this paper is to establish a theoretical framework for calculating the set of words that can be contented in the FIFO channel objects

We consider an ODP engineering object as a Finite State Machines, and an ODP specification as a collection of engineering object which communicate via and ODP channel object. We associate to each component state of a global state a set of words that may be contained in channels. We define, for each CFSM, a grammar ‘like’ context free in which, each rule is of the form $X \rightarrow u^{-1}Yv$, where $u^{-1}Yv$ stands for the residual of the language (L(Y)v) with regard to u such that: X and Y are variables corresponding to control states, and L(Y) is the language generated by Y. We use context-free grammar properties to make transformations and appear a symbol X in the right member of each X-production to express loop and cycle transitions in the CFSM. The algebraic property of context-free languages is then used to compute these languages which are minimal solutions of a system of equations.

The reminder of the paper is organized as follows. We introduce the model of communicating finite state machines in Section 2. FIFO channel languages and grammars are defined in Section 3. In Section 4 we use the algebraic property of context-free languages to calculate the channel languages, which are minimal solutions of some fix-point operator. A conclusion and perspectives end the paper.

II. The System Model

A Communicating Finite State Machine (CFSM) System is a finite state automaton equipped with a collection of finite state machines, and a collection of unbounded channels. There are generally two one-directional FIFO channels between each pair of machines. Each FSM may read from its incoming channel and write on its outgoing channel. If there is n FSM then the CFSM system is with n^2-n channels and is defined as follows:

For a set of FSM $(A_i)_{1 \le i \le n}$, where $A_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$, the CFSM is a structure (Q, Σ, q_0, δ) such that:

$Q = Q_1 \times Q_2 \times \ldots \times Q_n$ is the set of control states.

$\Sigma = (\Sigma_1 \cup \Sigma_2 \cup \ldots \cup \Sigma_N)$ where Σ_i is the alphabet of channel C_i with $\Sigma_i \cap \Sigma_j = \emptyset$, $\forall$ i $\neq$ j.

$q_0 = (q_1^0, \ldots, q_n^0)$ is the initial state.

$\delta \subseteq Q \times (\cup_i\{i!a, i?b \ / \ a, b \in \Sigma_i\}) \times Q$ is the transition relation, where i!a denotes the sending operation of the message a to the channel C_i (or write operation) and i?b the receiving operation of the message b from the channel C_i (or read operation). The transition relation is, in fact, built up from the transition relations of the FSM A_i’s where $\delta_i \subseteq Q_i \times (\cup_i\{i\} \times \{!,?\} \times \Sigma_i) \times Q_i$ such that, if (q, op, q’) $\in \delta$ then there is a FSM A_i such that $(q_i, op, q_i') \in \delta_i$ with op $\in (\cup_i\{i\} \times \{!, ?\} \times \Sigma_i)$, q_i and q_i' are components of q and q’.

Fig.1 shows an example where two different automata P_1 and P_2 are communicating via two FIFO channels; P_1 is a sender process and P_2 is the receiver. The corresponding CFSM is showed in Fig.2.

A global state of CFSM is a configuration which consists of two parts: component states representing the local states of processes and the contents of channels. Let $S = Q \times \Sigma_1^* \times \ldots \times \Sigma_n^*$ be the set of global states and $s_0 = \langle q_0, \varepsilon, \ldots, \varepsilon \rangle$ be the initial configuration. A labeled transition relation $\longrightarrow$ is defined, on the configuration set of CFSM, by the following two rules: $(\langle s_1, w_1, \ldots, w_i, \ldots, w_n \rangle \in S)$

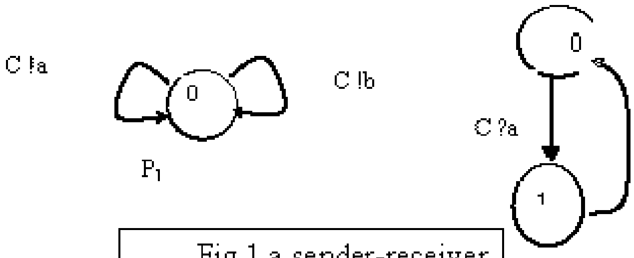

Fig.1 a sender-receiver

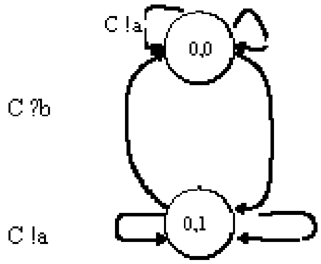

Fig.2 CFSM corresponding to

(i) If $(s_1, i!m, s_2) \in \delta$, where $m \in \Sigma_i$ then
$<s_1, w_1, \ldots, w_i, \ldots, w_n> \xrightarrow{i!m} <s_2, w_1, \ldots, w_i.m, \ldots, w_n>$
(ii) If $(s_1, i?m, s_2) \in \delta$, where $m \in \Sigma_i$ and $w_i = m.w_i'$ then
$<s_1, w_1, \ldots, w_i, \ldots, w_n> \xrightarrow{i?m} <s_1, w_1, \ldots, w_i', \ldots, w_n>$

The reflexive transitive closure $\xrightarrow{*}$ is defined as usual. A global state s is reachable if it's reachable from the initial state s_0 i.e. $s_0 \xrightarrow{*} s$.

III. FIFO CHANNEL LANGUAGES

We associate to each component state of a global state a set of words that may be contained in channels; this set is a language which can be calculated as shown in this section.

When the system has only one queue, there is no problem to represent a content of queue, we use finite words. But if there are many queues, we use a vector to represent N words (N>1). We set $\Sigma = (\Sigma_1, \Sigma_2, \ldots, \Sigma_N)$ where Σ_i is the alphabet of queue i, and we have $\Sigma^k = (\Sigma_1, \Sigma_2, \ldots, \Sigma_N)^k = (\Sigma_1^k, \Sigma_2^k, \ldots, \Sigma_N^k)$ and then $\Sigma^* = (\Sigma_1^*, \Sigma_2^*, \ldots, \Sigma_N^*)$.

Let (Q, Σ, q_0, δ) be a CFSM, and $Q \times \Sigma^*$ be the global state space. The reachability set is an element of $\wp(Q \times \Sigma^*)$. We associate a set of queue contents to each control state by considering a map $L : Q \longrightarrow \wp(\Sigma^*)$ associating a control state q with L(q) representing all possible contents of queues when the system is in the control state q.

Let $O = \cup_i \{i\} \times \{!,?\} \times \Sigma_i$ be the set of operations. We consider the function φ_i (res. ψ_i) from O to Σ_i^* which extracts the input (res. the output) message defined as: $\varphi_i(j!m) = m$ if $i = j$ and ε otherwise (res. $\psi_i(j?m) = m$ if $i = j$ and ε otherwise). These functions can be extended to homomorphisms from O^* to Σ^* by taking $\varphi(op) = (\varphi_1(op), \ldots, \varphi_N(op))$ and $\psi(op) = (\psi_1(op), \ldots, \psi_N(op))$.

Definition

Let L be a language on Σ ($L \subseteq \wp(\Sigma^*)$) and i?m be an operation in O. The residual of L related to a word m, noted $(i?m))^{-1}$ L, is defined as follows:

$(\psi(i?m))^{-1} L = \{(u_1, \ldots, u_i, \ldots u_N) \in \Sigma^* \, / \, (u_1, \ldots, mu_i, \ldots u_N) \in L\}$.

We note that if for any $(w_1, \ldots, w_i, \ldots w_N) \in L$ such that w_i is not prefixed by m, then $(\psi(i?m))^{-1} L = \varnothing$ ($\varnothing = (\varnothing, \ldots, \varnothing)$).

B. *Proposition*

Let (Q, Σ, q_0, δ) be a CFSM. According to rules (i) and (ii) the two following rules hold:

a) $(s_1, i!m, s_2) \in \delta$ with $m \in \Sigma_i \Rightarrow L(s_1)\, \varphi(i!m) \subseteq L(s_2)$.
b) $(s_1, i?m, s_2) \in \delta$ with $m \in \Sigma_i \Rightarrow (\psi(i?m))^{-1} L(s_1) \subseteq L(s_2)$.

Proof

a) Let be $(w_1, \ldots, w_i, \ldots, w_n) \in L(s_1)$. The global state $<s_1, w_1, \ldots, w_i, \ldots, w_n>$ is, then, reachable.
If $(s_1, i!m, s_2) \in \delta$ with $m \in \Sigma_i$ then $<s_2, w_1, \ldots, w_im, \ldots, w_n>$ is a global state, which is an immediate successor of $<s_1, w_1, \ldots, w_i, \ldots, w_n>$ according to rule (i). We have, then, $(w_1, \ldots, w_im, \ldots, w_n) \in L(s_2)$ i.e. $(w_1, \ldots, w_i, \ldots, w_n)\, \varphi(i!m) = (w_1, \ldots, w_im, \ldots, w_n) \in L(s_2)$; it follows that $L(s_1)\, \varphi(i!m) \subseteq L(s_2)$.
b) Let $(w_1, \ldots, w_i, \ldots, w_n) \in (\psi(i?m))^{-1} L(s_1)$, i.e. $(w_1, \ldots, mw_i, \ldots, w_n) \in L(s_1)$. If $(s_1, i?m, s_2) \in \delta$ with $m \in \Sigma_i$ then $<s_2, w_1, \ldots, w_i, \ldots, w_n>$ is an immediate successor of $<s_1, w_1, \ldots, mw_i, \ldots, w_n>$ according to rule (ii), and then $(w_1, \ldots, w_i, \ldots, w_n) \in L(s_2)$.
According to this proposition, we have, for any $s' \in Q$:
$L(s') = \{L(s)\, \varphi(i!m) \, / \, i \in \{1, \ldots N\}, m \in \Sigma_i, s \in Q : s \xrightarrow{i!m} s'$ is a labelled transition $\}$
$\cup \{(\psi(i?m))^{-1} L(s) \, / \, i \in \{1, \ldots N\}, m \in \Sigma_i, s \in Q : s \xrightarrow{i?m} s'$ is a labelled transition $\}$
$\cup$ E(s') where
$E(s') = \{\boldsymbol{\varepsilon}\}$ if $s' = q_0$ and $\varnothing$ otherwise, with convention $\boldsymbol{\varepsilon} = (\varepsilon, \ldots, \varepsilon)$.
This defines a system of card (Q) equations and card (Q) unknown to de determinated.

Example 1. (Case of one channel)

Given the CFSM depicted in fig:1 , which models sender and receiver processes acting only on one channel, we have the following system:

$$L(0,0) = L(0,0)\, a + L(0,0)\, b + a^{-1} L(0,1) + \varepsilon$$
$$L(0,1) = L(0,1)\, a + L(0,1)\, b + b^{-1} L(0,0)$$

This is equivalent to the system:

$$L(0,0) = L(0,0)\, (a + b) + a^{-1} L(0,1) + \varepsilon$$
$$L(0,1) = L(0,1)\, (a + b) + b^{-1} L(0,0).$$

C. *CFSM grammars*

The languages L(s), $s \in Q$, of the previous system can be calculated as solution of some fix-point operator. We define an algebraic grammar G, and associate to each state x a non terminal symbol X such that $L_G(X) = L(x)$.

Let $A = (Q,\Sigma, q_0,\delta)$ be a CFSM. We define a grammar $G = (\Sigma, V, P)$, where V and P are defined as follow:

1) $s \in Q$ if and only if $S \in V$. (card (V) = card (Q))

2) - $s_1 \xrightarrow{i!m} s_2$ is a labeled transition in A if and only if $s_2 \rightarrow s_1\, \varphi(i!m)$ is a production in P

- $s_1 \xrightarrow{i?m} s_2$ is a labeled transition in A if and only if $s_2 \rightarrow (\psi(i?m))^{-1}\, s_1$ is a production in P

- $q_0 \rightarrow \varepsilon$ is in P.

The grammar in tab.1 is a grammar corresponding to CFSM in fig.1, where S_{00} and S_{01} denote respectively states (0,0) and (0,1).

$$S_{00} \rightarrow S_{00}\, a + S_{00}\, b + b^{-1} S_{01} + \varepsilon$$
$$S_{01} \rightarrow S_{01}\, a + S_{01}\, b + a^{-1} S_{00}$$

Tab.1

IV. ALGEBRAIC PROPERTY

The inclusion relation gives to $\wp(\Sigma^*)$ a complete lattice structure that can be extended to Cartesian product of sets of Σ^* such that $L = (L_1,\dots, L_m) \subseteq M = (M_1,\dots, M_m)$ if and only if $\forall\, i\ (1 \le i \le m)\quad L_i \subseteq M_i$. Let $T = \wp(\Sigma^*)\ x\dots x\ \wp(\Sigma^*)$ be the Cartesian product m times. We denote by $\perp = (\emptyset,\dots, \emptyset)$ the smallest element of T.

A. *Theorem (Knaster-Tarski [26])*

Let $(E, \subseteq, \perp)$ be a complete lattice set and let $\tau : E \rightarrow E$ be a monotonous function, then τ has a least fix-point which is $\cup i \ge 0 \tau i(\perp)$.

B. *Definition*

Let $G = (\Sigma, V, P)$ be a grammar. To each $L = (L_1, \dots, L_m) \in T$, we associate the homomorphism $\lambda_L : \wp(\Sigma \cup V)^* \rightarrow \wp(\Sigma^*)$ defined as follows:

$\lambda_L(x) = x$, $\forall\, x \in \Sigma$, and $\lambda_L(S_i) = L_i\ \ \forall\, S_i \in V$.

This homomorphism can be extended to m Cartesian products of $\wp(\Sigma \cup V)^*$ such that:

If $B = (B_1,\dots, B_m)$ then $\lambda_L(B) = (\lambda_L(B_1),\dots, \lambda_L(B_m))$ and we set $\hat{b}(L) = \lambda_L(B)$ for any $B \in \wp(\Sigma \cup V)^*\ x\dots\ x\ \wp(\Sigma \cup V)^*$ (m times).

Now, if $P = (P_1, P_2,\dots, P_m)$ where $P_i = \{u \in (\Sigma \cup V)^* / S_i \rightarrow u$ is a production in P$\}$, we have, for any $L \in T$, $\lambda_L(P) = \hat{p}(L)$. ($\hat{p}$ is a function from T into T defined by the grammar).

C. *Proposition*

Let $G = (\Sigma, V, P)$ be a grammar proper and $\hat{p}$ defined as before. There exists a unique $L = (L_1, \dots, L_m) \in T$ such that $L = \hat{p}(L)$.

The proof use theorem of Knaster -Tarski.and we have $L = \cup_{n\ge0}\, \hat{p}^n(\perp)$. We can then calculate the solution L by successive approximations: $\hat{p}^0(\perp) = \perp$, $\hat{p}^1(\perp)$, ….

Example1. (System with only one queue)

For the grammar, in tab.1, corresponding to fig.2, we have:

$\Sigma = \{a, b\}$, $V = \{S_{00}, S_{01}\}$, $P_0 = \{ S_{00}\, a + S_{00}\, b + a^{-1} S_{01} + \varepsilon\}$, $P1 = \{ S_{01}\, a + S_{01}\, b + b^{-1} S_{00}\}$ and $P = (P_0, P_1)$.

Let be $L = (L_{00}, L_{01})$ such that $L = \hat{p}(L)$. The solution can be calculated by successive approximations:

$\hat{p}^0 (\perp) = (\emptyset, \emptyset)$,

$\hat{p}^1 (\perp) = \hat{p}(\emptyset,\emptyset) = (\varepsilon,\emptyset)$,

$\hat{p}^2 (\perp) = \hat{p}(\varepsilon,\emptyset) = (a + b + \varepsilon, \emptyset)$,

$\hat{p}^3 (\perp) = \hat{p}(a + b + \varepsilon, \emptyset) = ((a + b)^2 + (a +b) + \varepsilon, \varepsilon)$,

$\hat{p}^4 (\perp) = \hat{p}((a + b)^2 + (a +b) + \varepsilon, \varepsilon) = ((a + b)^3 + (a + b)^2 + (a +b) + \varepsilon, (a +b) + \varepsilon)$,

…

We calculate LG(S), for each $S \in V$, by successive approximations. We note that if there is a cycle (or a loop) of sending transitions or receiving transitions, in the control structure of CFSM, we may, then, use the star operation of messages labeling this cycle, and it may be calculated in a single step. The approximation converges quickly in this case. For this reason we enounce some results to transform a grammar into an equivalent grammar.

We transform each production $S \rightarrow u^{-1}\ B\ v$ into a production of the form $S \rightarrow x^{-1}Sy + C$. For example, we replace the production $S_{00} \rightarrow b^{-1}\ S_{01}$, for grammar in tab.1, by the production $S_{00} \rightarrow b^{-1}\ a^{-1}\ S_{00} + C$, or $S_{00} \rightarrow (a\ b)^{-1}\ S_{00} + C$ where $C = b^{-1}\ S_{01}\ (a + b)$.(see tab.2 for equivalent grammar).

Note that the expression $x^{-1}Sy$ denotes $x^{-1}(Sy)$: the left residual of language (Sy) related to x.

$$S_{00} \rightarrow S_{00}\, (a + b) + (a\ b)^{-1} S_{00} + b^{-1} S_{01}\, (a +b) + \varepsilon$$
$$S_{01} \rightarrow S_{01}\, (a + b) + (b\ a)^{-1} S_{01} + a^{-1} S_{00}\, (a +b)$$

Tab.2

The grammar in tab.2 is equivalent to the following grammar:

$$S_{00} \rightarrow S_{00}\, (a + b)^* + ((a\ b)^*)^{-1} S_{00} + b^{-1} S_{01}(a + b) + \varepsilon$$
$$S_{01} \rightarrow S_{01}\, (a + b)^* + ((b\ a)^*)^{-1} S_{01} + a^{-1} S_{00}(a + b)$$

Tab.3

The solution, corresponding to the grammar in tab.3, can be calculated as follows:

$\hat{p}^1 (\emptyset,\emptyset) = (\varepsilon, \emptyset)$

$\hat{p}^2 (\emptyset, \emptyset) = \hat{p}\ (\varepsilon, \emptyset) = ((a + b)^*, \emptyset)$

$\hat{p}^3 (\emptyset, \emptyset) = \hat{p}\ ((a + b)^*, \emptyset) = ((a + b)^*, (a + b)^*)$

$\hat{p}^4 (\emptyset, \emptyset) = \hat{p}^3 (\emptyset, \emptyset) =((a + b)^*, (a + b)^*)$.

We have then, the language of the channel, when the system is in control states (0,0) and (0,1), is $(a + b)^*$, i.e. $L_{C1}\ (0,0) = L_{C1}\ (0,1) = (a + b)^*$.

V. CONCLUSION AND FUTURE WORK

In this paper, we apply a theoretical framework for calculating the languages of channels for communicating finite state engineering objects in the context of RM-ODP. These languages are fix-point of a system of equations which can be generated by context-free grammars; where the right member of each production is of the form $A^{-1}\ S\ B$, A is a set of terminal symbols and A^{-1} is the left residual of the language generated by S B, i. e. $A^{-1}L(S\ B)$. Hence, we are studying how the proposed framework can be effective by developing tools for this method. As the objective is to verify properties of communication protocols, we are studying how to determine the test sequences. We are also trying to extend the framework to timed automata.

REFERENCES

[1] ISO/IEC, ''Basic Reference Model of Open Distributed Processing-Part1: Overview and Guide to Use, ''ISO/IEC CD 10746-1, 1994
[2] ISO/IEC, ''RM-ODP-Part2: Descriptive Model, '' ISO/IEC DIS 10746-2, 1994.
[3] ISO/IEC, ''RM-ODP-Part3: Prescriptive Model, '' ISO/IEC DIS 10746-3, 1994.
[4] ISO/IEC, ''RM-ODP-Part4: Architectural Semantics, '' ISO/IEC DIS 10746-4, July 1994.
[5] M. Bouhdadi, el al. '' An UML-based Meta-language for the QoS-aware Enterprise Specification of Open Distributed Systems, '' IFIP International Federation for Information Processing Series, Vol. 85, Springer, pp 255-264. 2002.
[6] M. Bouhdadi, Y. Balouki, E.M. Chabbar, ''Meta-modelling Syntax and Semantics of Structural Concepts for Open Networked Enterprises,'' International Conference on Computational Science and its Applications, Kuala Lumpur, Malaysia, 2007, ICCSA(3) LNCS Vol. 4707, Springer, pp. 45-54, 2007.
[7] M. Bouhdadi, et al. '' A UML/OCL Denotational Semantics for ODP Structural Computational Concepts,'' First IEEE Research Challenges in Information Science, RCIS 2007, pp
[8] E.M. Chabbar, M. Bouhdadi,'' Context free grammars for Languages of FIFO channel Systems, ''Information and Communication and Technologies International Symposium, ICTIS 2007 Fez, Morocco, to appear in IJCSA journal, ICTIS'07 Special Issue.
[9] G. V. Bochmann, ''Finite state description of communication protocols, '' Computer Networks, vol. 2, pp. 361-372, 1978.
[10] D. Brand et al." On communicating finite-states machines. " J. ACM, vol. 30, no.2, , pp. 323-342, 1983.
[11] P. Godefroid. "Using partial orders to improve automatic verification methods, " In the 2nd International Conference on Computer Aided Verification (CAV '90), New Jersey, June 18-21, 1990, LNCS vol. 531, Springer Berlin-Heidelber, 1991, pp : 176-185.
[12] A. Lluch-Lafuente, et al. , ''Partial Order Reduction in Directed Model Checking, ''. In Proceedings of the 9th International SPIN Workshop on Software Model Checking, France, April 11-13, 2002, LNCS vol. 2318 ,Springer, 2002, pp : 112-127
[13] D. Peled, ''Ten years of partial order reduction, ''. In 10th International Conference on Computer Aided Verification, (CAV 98), Vancouver, Canada, June 28- July 2, LNCS vol. 1427, Springer, 1998, pp : 17- 28
[14] P. Abdulla, A. Annichini and A. Bouajjani, ''Symbolic verification of lossy channel systems: Application to bounded retransmission protocol, '', In Proc. Fifth International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99), ETAPS'99, Amsterdam, the Netherlands, March 22-28, 1999, LNCS vol. 1579, Springer, pp : 208-222
[15]] P. Abdulla, et al, '' On-the-fly analysis of systems with unbounded, lossy FIFO channels, '' In Proc. 10th Inter. Conf. on Computer Aided Verification, (CAV 98), Vancouver, Canada, June 28-July 2, LNCS vol. 1427, Springer, 1998, pp. 305-318.
[16] A. Finkel, S. P. Iyer and G. Sutre, ''Well-Abstracted Transition systems: Application to FIFO Automata, ''. Information and Computation, vol. 181, no. 1, pp. 1-31, 2003.
[17] A. Bouajjani, P. Habermehl, T. Vojnar, '' Abstract Regular Model Checking''. In 16th International Conference on Computer Aided versification (CAV 2004), Boston, USA, Jully 13-17, 2004, LNCS vol. 3114, Springer, 2004, pp .372-386
[18] A. Bouajjani, B. Jonsson, M. Nilson and T. Touili. ''Regular Model Checking''. In 12th Int. Conf. on Computer Aided Verification (CAV'00), Chicago, USA, July 15-19, 2000, LNCS vol. 1855, Springer, 2000, pp. 403-418
[19] J. K. Pachl, ''Protocol description and analysis based on a state transition model with channel expressions, ''. Proc. of Protocol Specification, Testing and Verification, (PSTN'87), North Holland, 1987, pp. 207-219.
[20] B. Boigelot and P. Godefroid, " Symbolic verification of communicating protocols with infinite state spaces using QDDs, '' Formal Methods in System design, vol. 14 no. 3, 1999
[21] N. Ghfari and R. Trefler. Piecewise FIFO Channels are Analyzable. 7th International Conferonce on Verification, Model Checking, and Abstract Interpretation (VMCAI '06), Charleston, USA, LNCS 3855, Springer, 2006, pp:252-266
[22] N. Klarlund and R. J. Treler., ''Regularity Results for FIFO Channels, '' Electronic Notes of theor. Comput. Sci. vol.128, no. 6, Elsevier, Mai 2005, pp: 252-266.
[23] T. Le Gall, B. Jeannet, T. Jeron, ''Verification of Communication Protocols using Abstract Interpretation of FIFO queues, '' In the 11th International Conference on Algebraic Methodology and Software Technology (AMAST '06), Kuressaare, Estonia, 2006, LNCS 4019, Springger, 2006, pp. 204-219.
[24] J. Brozozowski., ''Derivatives of regular expressions, '' J. ACM, vol. 11, no. 4, 1964.
[25] J. M. Autebert, ''Théorie des langages et des automates, '' Masson (eds.), 1994.
[26] J. E. Hopcroft and J. D. Ullman, ''Introduction to automata theory, Languages, and Computation., '' Addison-Wesley, 1979.