# On the Architectures in Decentralized Supervisory Control

Sophie Pinchinat and Stéphane Riedweg
Sophie Pinchinat is with IRISA-INRIA, F-35042, Rennes, France
Stéphane Riedweg is with LVS, ENS Cachan, F-94235, Cachan, France
IRISA, Campus Universitaire de Beaulieu, 35042 Rennes, France
{Sophie.Pinchinat@irisa.fr,Stephane.Riedweg@lsv.ens-cachan.fr}

***Abstract*— In this paper, we clarify the notion of *architecture* in decentralized control, in order to investigate the *realizability* problem: given a discrete-event system, a desired behavior and an architecture for a decentralized control, can the desired behavior be achieved by decentralized controllers in accordance with the given architecture? We consider the problem for any mu-calculus definable behavior and for classic architectures from the literature. The method consists in compiling in a single formula both the desired behavior and the architecture. Applications of this approach are a single synthesis algorithm of decentralized controllers (with full observation) for the whole considered family of architectures, and the development of a convenient mathematical framework for a theory of decentralized control architectures.**

## I. INTRODUCTION

The Decentralized Control Problem is a challenging topic in the discrete-event systems theory. In this setting, decentralized (or local) controllers supervise the global behavior of a component-based system on the basis of partial observation of its moves. The controllers might collaborate to disallow moves of the components they are in charge of, in order to ensure a behavioral property of the (global) system. The desired behavior can be a set of finite sequences of reactions, defined by a regular language, or as we do here, by a temporal logic formula.

Classically, a decentralized control problem relies on a predefined *architecture of control* which specifies how the criteria for the local decision rules and the criterion for the global decision rule (the collaboration between the controllers). Obeying its local decision rule, each controller makes a local decision to select the transitions it allows in the current state of the global system, and by the global decision rule, the local selections are fused to deliver the set of transitions that finally can be fired. For example, in the (conjunctive permissive) architecture of [13], the local decision rule amounts to allow any uncontrollable transition, while the global decision rule keeps only those transitions that are allowed by all the controllers. As explained by [13], most of the works in the literature on decentralized control are based on a conjunctive architecture (see the pioneer works of [3] and [12]): However, a few works investigate other decisions criteria: [8] have studied rules where a transition is allowed provided a majority of controllers have selected it, and [13] have proposed a so-called *general architecture* where the global decision rule relies on a disjunctive fusion of the local decision rules; also a combination of a conjunctive fusion and a disjunctive fusion is studied in details.

In this paper, we make clear the central notion of *architecture* by considering a logical framework: the local decision rules are formulas which characterize the class each controller belongs to, and the global decision rule is a set of formulas indexed by the set of events of the global system; the latter formulas will be embedded in the logical formula describing the desired behavior. As a consequence, the essential issue of *Realizability*, stated as “given a discrete-event system, a desired behavior and an architecture for a decentralized control, can the desired behavior be achieved by decentralized controllers obeying the given architecture?” can be turned into the model-checking of some second order formula on the global system.

Realizability have already been investigated for a behavior which is some regular language $K$, as proposed by the Ramadge and Wonham control theory [9]. Since [3] and [12], followed by [13], the *co-observability* property of $K$ is commonly accepted as the algebraic characterization of its realizability. Moreover, the co-observability property is known to be decidable, with established computational complexity results, and synthesis algorithms exist in this successful case. Unfortunately, sticking to the conjunctive architecture while relaxing the class of behavior leads to undecidability: [2] showed that the undecidable Post Correspondence Problem [7] reduces to a Realizability Problem for the conjunctive architecture and for a very simple temporal logic definable behavior.

In this paper, we investigate the Realizability Problem where the desired behavior can be any propositional Mu-calculus formula [5] and where the family of architectures is comes from the literature. The principle of our approach relies on a second order extension of the Mu-calculus, in the spirit of [11]. For lack of space, we demonstrate the correctness of our logic-based approach for convincing cases of the conjunctive-permissive and disjunctive-antipermissive architectures of [13].

As expected, by [2], our verification problem is necessarily undecidable in general, but a uniform synthesis procedure can be exhibited in the restricted case of total observation; this is a strong result, since in general, the local controllers are not entitled to take the same kind of local decisions, and the synthesis procedure is not trivial.

The paper is organized as follows: Section II presents the Mu-calculus $L_\mu$ and its interpretation over discrete-event systems, called *processes*. In Section III, we recall the state the art on decentralized control architectures and we define the Realizability Problem for Mu-calculus definable desired behaviors. A second order extension of the Mu-calculus is proposed in Section IV, by allowing for quantifications on propositions. In Section V, we revise the Realizability Problem by introducing the notion of *logically defined architectures*; we prove the correctness of the approach by a comparing it with the original approaches for the conjunctive-permissive and the disjunctive-antipermissive architectures. In Section VI, for the favorable case of total observation, a uniform synthesis procedure solving the Realizability Problem is explained.

## II. Mu-calculus and Processes

We assume given finite sets $\mathtt{Ev} = \{\sigma, \sigma_1, \sigma_2, \dots\}$ and $Var = \{X, Y, \dots\}$, respectively of *events* and *variables*.

*Definition 1:* **Syntax of the Mu-calculus.** The set of formulas of the Mu-calculus, written $L_\mu$, is defined by the following grammar:

$$L_\mu(\ni \varphi, \varphi') ::= \top \mid \neg\varphi \mid \varphi \vee \varphi' \mid\ {<}\sigma{>}\varphi \mid X \mid \mu X.\varphi(X)$$

where $\sigma \in \mathtt{Ev}$ and $X \in Var$.

Technically, fix-points formulas $\mu X.\varphi(X)$ can properly be interpreted (Def.3) whenever each occurrence of $X$ in $\varphi(X)$ is under an even number of negation symbols $\neg$; see [1]. We respectively use $[\sigma]\varphi$, $\varphi \wedge \varphi'$, and $\nu X.\varphi(X)$ for $\neg\ {<}\sigma{>}\neg\varphi$, $\neg(\neg\varphi \vee \neg\varphi')$, and $\neg\mu X.\neg\varphi(\neg X)$, moreover, $[\ ]\varphi$ is a notation for $\bigwedge_{\sigma\in\Sigma}[\sigma]\varphi$. Formulas where each occurrence of a variable $X$ is binded by a fix-point symbol $\mu$ or $\nu$ are called *sentences*.

The Mu-calculus is interpreted over *traces* of*processes*. Formally,

*Definition 2:* **Processes and their Traces.** A *process with type* $\Sigma \subseteq \mathtt{Ev}$ is a tuple $\mathcal{S} = \langle S, s^0, \longrightarrow\rangle$, where $S$ is a set of states, $s^0 \in S$ is the initial state, and $\longrightarrow: S \times \Sigma \to S$ is a partial function called the *transition function*. A process $\mathcal{S}$ is *finite* if $S$ is finite; and it is *complete* if $\longrightarrow(s, \sigma)$ is defined for all $s \in S$ and $\sigma \in \Sigma$. We use typical elements $\mathcal{S}, \mathcal{S}_1, \mathcal{S}_2, \mathcal{R}$, for processes.

We write $\Sigma(\mathcal{S})$ for the type of $\mathcal{S}$, or simply $\Sigma$ when it is clear from the context. Also, we introduce intuitive notations: $s \xrightarrow{\sigma} s'$ is an intuitive notation whenever $\longrightarrow(s, \sigma) = s'$, $s \longrightarrow s'$ means that $s \xrightarrow{\sigma} s'$ for some $\sigma$, and $s \xrightarrow{\sigma}$ means that $s \xrightarrow{\sigma} s'$ for some $s'$.

An *execution* $\pi$ of $\mathcal{S}$ is a finite sequence of the form $s_0 s_1 \dots s_k$ where $s_0 = s^0$ and $s_{j-1} \longrightarrow s_j$ for all $0 < j \leq k$; $|\pi| = k$ is the length of $\pi$, and $\pi(j) = s_j$, hence $\pi(|\pi|)$ is the last state of $\pi$. Strongly related to executions, *traces* put the emphasis on events rather than states: a *trace* of $\mathcal{S}$ is a finite sequence $\theta = \sigma_1 \dots \sigma_k$ such that there exists an execution $\pi$ of $\mathcal{S}$ with $|\pi| = k$ where $\pi(j-1) \xrightarrow{\sigma_j} \pi(j)$ for all $1 \leq j \leq k$; clearly, via the transition function of $\mathcal{S}$, there is a one-to-one correspondence between traces and executions. Each trace $\theta$ defines a unique execution, written $\pi_\theta$. Hence, by abuse of vocabulary, we speak about the last state of a trace $\theta$, and write it $state(\theta)$; it is the last state of $\pi_\theta$. We write $Tr(\mathcal{S})$ the set of traces of $\mathcal{S}$, and denote by $\epsilon$ the empty trace; clearly $\pi_\epsilon = s^0$. In the following, $\theta\sigma$ denotes the sequence obtained by concatenating $\sigma$ at the end of the sequence $\theta$; $\theta\sigma \in Tr(\mathcal{S})$ whenever $\theta \in Tr(\mathcal{S})$ and $state(\theta) \xrightarrow{\sigma}$.

Now, given a process $\mathcal{S}$, a formula $\varphi \in L_\mu$ is interpreted as a subset of $Tr(\mathcal{S})$, those traces which *satisfy* $\varphi$. As the semantics will be given by induction over $\varphi$, we need to fix an interpretation for the atomic variable formulas $X \in Var$; we will consider a *valuation* $val$, namely a function $val : Var \to 2^{Tr(\mathcal{S})}$, for the interpretation of each $X \in Var$.

The Mu-calculus formulas' semantics is defined by the following:

*Definition 3:* **Semantics of the Mu-calculus.** Given a process $\mathcal{S} = \langle S, s^0, \longrightarrow\rangle$ and a valuation $val : Var \to 2^{Tr(\mathcal{S})}$, the interpretation of the formula $\varphi$ is $\llbracket\, \varphi \,\rrbracket_{\mathcal{S}}^{val} \subseteq Tr(\mathcal{S})$ inductively defined by:

$$\llbracket\, \top \,\rrbracket_{\mathcal{S}}^{val} = Tr(\mathcal{S}) \qquad\qquad \llbracket\, \neg\varphi \,\rrbracket_{\mathcal{S}}^{val} = Tr(\mathcal{S}) \setminus \llbracket\, \varphi \,\rrbracket_{\mathcal{S}}^{val}$$

$$\llbracket\, \varphi \vee \varphi' \,\rrbracket_{\mathcal{S}}^{val} = \llbracket\, \varphi \,\rrbracket_{\mathcal{S}}^{val} \cup \llbracket\, \varphi' \,\rrbracket_{\mathcal{S}}^{val}$$

$$\llbracket\, {<}\sigma{>}\varphi \,\rrbracket_{\mathcal{S}}^{val} = \{\theta \in Tr(\mathcal{S}) | state(\theta) \xrightarrow{\sigma} \text{ and } \theta\sigma \in \llbracket\, \varphi \,\rrbracket_{\mathcal{S}}^{val}\}$$

$$\llbracket\, X \,\rrbracket_{\mathcal{S}}^{val} = val(X)$$

$$\llbracket\, \mu X.\varphi(X) \,\rrbracket_{\mathcal{S}}^{val} = \bigcap\{T \subseteq Tr(\mathcal{S}) \,|\, \llbracket\, \varphi \,\rrbracket_{\mathcal{S}}^{val(T/X)} \subseteq T\}$$

Since for a sentence $\varphi$ any variable is bounded by a fix-point operator, the interpretation of $\varphi$ is independent of the valuation $val$; we then simply write $\llbracket\, \varphi \,\rrbracket_{\mathcal{S}}$, and we use $\mathcal{S} \models \varphi$ to express that $\epsilon \in \llbracket\, \varphi \,\rrbracket_{\mathcal{S}}$, and we read it "the process $\mathcal{S}$ satisfies the sentence $\varphi$".

The reader might wonder why our interpretation of the Mu-calculus formulas is not standard, as in general, see for example [1], the interpretation is a set of states rather than a set of traces. The reason is that we aim at comparing formulas and local/global decision rules of controllers, and the latter rely on executions (traces). However, the standard definition is retrieved by considering the set $state(\llbracket\, \varphi \,\rrbracket_{\mathcal{S}}^{val}) \subseteq S$.

For any sentence $\varphi$, $\mathbf{Inv}(\varphi)$ is a notation for $\nu X.([\ ]X \wedge \varphi)$. The statement $\mathbf{Inv}(\varphi)$ intuitively means that "from now on, the property $\varphi$ always holds". Namely, $\mathcal{S} \models \mathbf{Inv}(\varphi)$ if and only if for all $\theta \in Tr(\mathcal{S})$, $\theta \in \llbracket\, \varphi \,\rrbracket_{\mathcal{S}}$. Clearly, $\mathbf{Inv}(\varphi \wedge \varphi')$ is equivalent to $\mathbf{Inv}(\varphi) \wedge \mathbf{Inv}(\varphi')$, for any sentences $\varphi$ and $\varphi'$.

## III. THE DECENTRALIZED CONTROL PROBLEM WITH ARCHITECTURE

The natural framework for decentralized control problems is based on a *conjunctive architecture* ([3],[12]), followed by [8], [13] who proposed other constructions. We recall these recent approaches and formalize them in a unified mathematical setting, for the sake of clarity.

Let us write $\mathcal{I}$ for the finite set $\{1, ..., n\}$, to name $n$ controllers we shall explain now. Each controller $i \in I$ is represented by a partial function $f_i : \texttt{Ev}^* \to 2^{\texttt{Ev}}$, called a *local decision rule*, with particular properties according to two subsets of events: $\Sigma_{c,i}$ and $\Sigma_{o,i}$ respectively denoting the events the controller $i$ has control on, and the events the controller can observe. Events in $\Sigma_{uc} = \texttt{Ev} \setminus \bigcup_{\texttt{i} \in \mathcal{I}} \Sigma_{\texttt{c,i}}$ are said to be *uncontrollable*. Although not explicit, e.g. in [13], the $\Sigma_{c,i}$'s should be assumed pairwise disjoint.

We preliminarily introduce some convenient technical definitions and notations. Given, $i \in \mathcal{I}$, the projection $P_i : \texttt{Ev}^* \to \Sigma^*_{\texttt{o,i}}$ forgets all events outside $\Sigma_{o,i}$. Henceforth, $P_i(\theta)$ is what the controller $i$ observes from a trace $\theta$. We say that two traces $\theta$ and $\theta'$ are *indistinguishable for* $i$, written $\theta \sim_i \theta'$, whenever $P_i(\theta) = P_i(\theta')$.

We can now formalize the notion of *decision rules* as the basis of the notions of *decentralized control* and *architecture*. We assume fixed a process $\mathcal{S}$ with type $\Sigma$, and $2 * n$ sets $\Sigma_{c,i}$ and $\Sigma_{o,i}$ (with $i \in \mathcal{I}$).

*Definition 4:* **Decision Rules, Decentralized Control and Type of Architecture.** A *decision rule* is a function $f : \texttt{Ev}^* \to 2^{\texttt{Ev}}$, with $\Sigma_{uc} \subseteq f(\theta)$ for all $\theta \in \texttt{Ev}^*$. The *$f$-control of $\mathcal{S}$* is the process $\mathcal{S}_{|f} = \langle Tr(\mathcal{S}), \epsilon, \longrightarrow \rangle$ where $\theta \xrightarrow{\sigma} \theta\sigma$ whenever $\sigma \in f(\theta)$ (hence $state(\theta) \xrightarrow{\sigma}$). A decision rule $f$ is *$i$-local* if moreover $\theta \sim_i \theta'$ implies $f(\theta) = f(\theta')$

A *decentralized control* is a pair $F = (\{f_i\}_{i \in \mathcal{I}}, f)$, where each $f_i$ is an $i$-local decision rule and $f$ is a decision rule called the *global decision rule*. The $F$-control of $\mathcal{S}$ is the process $\mathcal{S}_{|f}$, also written $\mathcal{S}_{|F}$. In general, $f$ is strongly related to the $f_i$'s, as we will see bellow.

An *architecture* is a (possibly infinite) family of decentralized controls. We use $\texttt{Arch}$ for a typical architecture.

Architectures can be specified by stating constraints on the local and the global decision rules of its elements, as in [8] and [13]. For example, we specify the *conjunctive-permissive architecture* by Eq.(CP), and the dual *disjunctive-antipermissive architecture* by Eq.(DA).

$$\begin{array}{l} \forall \theta \in Tr(\mathcal{S}), f(\theta) = \bigcap_{i \in \mathcal{I}} f_i(\theta) \\ \forall k \neq i, \Sigma_{c,k} \subseteq f_i(\theta) \end{array} \qquad \text{(CP)}$$

$$\begin{array}{l} \forall \theta \in Tr(\mathcal{S}), f(\theta) = \bigcup_{i \in \mathcal{I}} f_i(\theta) \\ \forall k \neq j, \Sigma_{c,k} \cap f_j(\theta) = \emptyset \end{array} \qquad \text{(DA)}$$

The *mixed architecture* of [13] (called *general architecture* in their paper) combines Eq.(CP) and Eq.(DA), according to two groups of controllers arranged respectively in a conjunctive and disjunctive manner, the result obeys a disjunctive fusion. Formally, given a subset $I \subseteq \mathcal{I}$, and its complementary set $\bar{I} = \mathcal{I} \setminus I$, the $CPDA(I)$-architecture is defined by Equation (CPDA(I)), for any $\theta \in Tr(\mathcal{S})$:

$$\left\{ \begin{array}{ll} \Sigma_{c,k} \subseteq f_i(\theta) & \forall i \in I, \forall k \neq i \\ \Sigma_{c,k} \cap f_j(\theta) = \emptyset & \forall j \in \bar{I}, \forall k \neq j \\ f(\theta) = (\bigcap_{i \in I} f_i(\theta)) \cup (\bigcup_{j \in \bar{I}} f_j(\theta)) & \end{array} \right. \qquad \text{(CPDA(I))}$$

In a slightly different spirit, [8] propose a global decision which rules out an event $\sigma$ whenever there is at least a fixed number $k_\sigma \leq |\mathcal{I}|$ of controllers which disallow $\sigma$. Dually, $\sigma$ is globally allowed whenever for any possible subset $J \subseteq \mathcal{I}$, with $|J| = k_\sigma$, there is at least one controller $j \in J$ which allows $\sigma$. This is formalized by the Equation (K) (where $K = \{k_\sigma\}_{\sigma \in \Sigma}$ is a fixed parameter) giving the *Below-K* global decision rule: for all $\theta \in \texttt{Ev}^*$,

$$f(\theta) = \bigcup_{\sigma \in \Sigma} \bigcap_{\substack{J \subseteq \mathcal{I} \\ |J| = k_\sigma}} (\bigcup_{j \in J} f_j(\theta) \cap \{\sigma\}) \qquad \text{(K)}$$

Consider now **the Realizability Problem (RP)**: Given a process $\mathcal{S} = \langle S, s^0, \longrightarrow \rangle$, a sentence $\varphi \in L_\mu$, and an architecture $\texttt{Arch}$, Compute, when it exists, a decentralized control $F \in \texttt{Arch}$ s.t. $\mathcal{S}_{|F} \models \varphi$. We then say that $\varphi \in L_\mu$ is $\texttt{Arch}$-*realizable* on $\mathcal{S}$

From [13], we know that the RP is decidable when we stick to formulas which characterize regular languages, and to the architectures of Eq.(CPDA(I)). On the contrary, by [2], the RP becomes undecidable even for the single case of the conjunctive architecture, if additionally $|\Sigma_{c,i} \setminus \Sigma_{c,j}|, |\Sigma_{c,j} \setminus \Sigma_{c,i}| \geq 1$, for some $i, j \in \mathcal{I}$.

## IV. ADDING QUANTIFIED PROPOSITIONS TO THE MU-CALCULUS

We add propositions to the logic and its models to encode the decision rules. In the following, (the last state of) a trace carries the proposition $c^\sigma$ whenever the controller allows $\sigma$ to occur after the trace. For this, we assume fixed an infinite set $AP = \{p, c^\sigma, c^\sigma_1, c^\sigma_2, \dots\}$ of atomic propositions, and we allow any $p \in AP$ to be a sentence of $L_\mu$.

We say that a sentence is *propositional* if it is composed of symbols $\top$, $p$, $\neg$, or $\vee$.

The interpretation of the propositions is obtained by considering *labeled* processes: a process $\mathcal{S} = \langle S, s^0, \longrightarrow \rangle$ is *labeled over* $P \subseteq AP$ if it is provided with a function $\lambda : S \to 2^P$; we write $\mathcal{S} = \langle S, s^0, \longrightarrow, \lambda \rangle$. We take the convention that $\lambda(\theta) = \lambda(state(\theta))$, for any trace $\theta$. We interpret the proposition $p$ on $\mathcal{S}$ according to $[\![ \, p \, ]\!]^{val}_{\mathcal{S}} = \{\theta \in Tr(\mathcal{S}) \, | p \in \lambda(\theta)\}$.

Labeled processes are combined by means of a synchronous product as follows:

*Definition 5:* **Synchronous Product.** The *synchronous product* of $\mathcal{S}_1 = \langle S_1, s^0_1, \longrightarrow_1, \lambda_1 \rangle$ and $\mathcal{S}_2 = \langle S_2, s^0_2, \longrightarrow_2 , \lambda_2 \rangle$, with $\Sigma_1 = \Sigma(\mathcal{S}_1)$ and $\Sigma_2 = \Sigma(\mathcal{S}_2)$, is the process

$\mathcal{S}_1 \otimes \mathcal{S}_2 = \langle S_1 \times S_2, (s_1^0, s_2^0), \rightarrow \rangle$, with $\Sigma(\mathcal{S}) = \Sigma_1 \cup \Sigma_2$, where $(s_1, s_2) \xrightarrow{\sigma} (s_1', s_2')$ whenever either:

- $\sigma \in \Sigma_1 \cap \Sigma_2$ and $s_1 \xrightarrow{\sigma} s_1'$ and $s_2 \xrightarrow{\sigma} s_2'$, or
- $\sigma \in \Sigma_1 \setminus \Sigma_2$ and $s_1 \xrightarrow{\sigma} s_1'$ and $s_2' = s_2$, or
- $\sigma \in \Sigma_2 \setminus \Sigma_1$ and $s_2 \xrightarrow{\sigma} s_2'$ and $s_1' = s_1$.

and where $\lambda((s_1, s_2)) = \lambda_1(s_1) \cup \lambda_2(s_2)$, that local labels of the components are merged in the product.

The traces of the process $\mathcal{S}$ are meant to be labeled, which technically amounts to compose $\mathcal{S}$ with a *labeling process* $\mathcal{E}$:

*Definition 6:* **Labeling Process.** Given a set $C = \{c^\sigma\}_{\sigma \in \Sigma}$ of atomic propositions, a *C-labeling process* is a process $\mathcal{E} = \langle E, e^0, \rightarrow, \lambda \rangle$ labeled over $C$, and which is complete (over its type). For $\Sigma \subseteq \mathtt{Ev}$, we let $LProc_\Sigma(C)$ be the set of $C$-labeling processes with type $\Sigma$, and we use $\mathcal{E}$, $\mathcal{E}_i$ for typical elements.

A *$C$-labeling of $\mathcal{S}$*, is a product $\mathcal{S} \otimes \mathcal{E}$ where $\mathcal{E} \in LProc_\Sigma(C)$ and $\Sigma \subseteq \Sigma(\mathcal{S})$.

Remark that $Tr(\mathcal{S}) = Tr(\mathcal{S} \otimes \mathcal{E})$, up to the labels, since $\Sigma(\mathcal{E}) \subseteq \Sigma(\mathcal{S})$ and $\mathcal{E}$ is complete. Remark also that given $C_i$-labeling of $\mathcal{S}$, $\mathcal{E}_i \in LProc_{\Sigma_{o,i}}(C_i)$ (for each $i \in \mathcal{I}$), where $C_i = \{c_i^\sigma\}_{\sigma \in \Sigma}$, a unique $i$-local decision rule $f_i$ is derived: $\sigma \in f_i(\theta)$ if and only if $c_i^\sigma \in \lambda(\theta)$, in $\mathcal{S} \otimes \mathcal{E}_i$.

Actually, the RP amounts to state the existence of labeling processes $\mathcal{E}_i \in LProc_{\Sigma_{o,i}}(C_i)$. We add logical constructs to the Mu-calculus in order to state this existence as $\exists C_i(\Sigma_{o,i})$. We obtain a second order logic, that will be called the *second order Mu-calculus* and we write $\mathtt{SO}L_\mu$, defined as follows:

$$\mathtt{SO}L_\mu(\ni \alpha, \alpha') ::= \varphi \mid \exists C(\Sigma).\alpha \mid \neg\alpha \mid \alpha \vee \alpha'$$

where $\varphi$ is $L_\mu$ sentence, $C$ is a short abstract notation designating a set $\{c^\sigma\}_{\sigma \in \Sigma}$ of fresh atomic propositions and $\Sigma \subseteq \mathtt{Ev}$. Remark that a quantification $\exists$ cannot occur inside fixed-point terms of Mu-calculus formulas.

The semantics of $\alpha \in \mathtt{SO}L_\mu$ is a subset $[\![\, \alpha \,]\!]_\mathcal{S} \subseteq Tr(\mathcal{S})$ inductively defined by:

- $[\![\, \varphi \,]\!]_\mathcal{S}$ is given by Def.1;
- $[\![\, \neg\alpha \,]\!]_\mathcal{S}$ and $[\![\, \alpha \vee \alpha' \,]\!]_\mathcal{S}$ are standard;;
- $\theta \in [\![\, \exists C(\Sigma).\alpha \,]\!]_\mathcal{S}$ whenever $\theta \in [\![\, \alpha \,]\!]_{\mathcal{S} \otimes \mathcal{E}}$ for some $\mathcal{E} \in LProc_\Sigma(C)$.

The interpretation of $\exists C(\Sigma).\alpha$ needs some more explanation: $\Sigma$ represents the set of observations. A trace $\theta$ is labeled by $c^\sigma$ when the underlying controller locally decides to allow $\sigma$ after $\theta$. As local decision rules must be consistent with the observation of the controller, any two indistinguishable traces $\theta, \theta'$ should then be either both labeled by $c^\sigma$ or both not labeled by $c^\sigma$. If $c^\sigma$ holds in $\theta$, it should hold in $\theta\sigma'$, if $\sigma'$ is not observed ($\sigma' \notin \Sigma$). By the interpretation of $\exists C(\Sigma).\alpha$, the propositions $c^\sigma \in C$ come from labeling process which type is $\Sigma$; henceforth, by the synchronous product, on the occurrence of the event $\sigma'$, the labeling process $\mathcal{E}$ does not change its current state carrying the label $c^\sigma$, which entails that $c^\sigma$ also holds at $\theta\sigma'$.

In the next section, we explain how the logic $\mathtt{SO}L_\mu$ can express the RP.

## V. Revising the Realizability Problem

Consider a set $C_i = \{c_i^\sigma\}_{\sigma \in \mathtt{Ev}}$ of atomic propositions for each controller $i \in \mathcal{I}$.

We now revise the notion of *architecture* in a correct manner as stated by Theo.10. We prefix the logic-based definitions by "L-" to distinguish them from the original ones (of Sec.III).

*Definition 7:* **L-Architecture.** A *logically defined architecture* is a structure $\mathtt{LArch} = (\{\varphi_i\}_{i \in \mathcal{I}}, B)$ where each $\varphi_i$ is sentence, called the *$i$-local decision formula*, and $B = \{B_\sigma\}_{\sigma \in \mathtt{Ev}}$ is a set of propositional formulas, where each $B_\sigma$ is build upon a set of fresh propositions $\{c_i^\sigma\}_{i \in \mathcal{I}} \subseteq AP$, and is called the $\sigma$-global decision formula.

We first combine the $\sigma$-global decision formulas $B_\sigma$ with the desired behavior $\varphi \in L_\mu$, using the notion of *adjustment*:

*Definition 8:* **Adjustment.** Given a set $B = \{B_\sigma\}_{\sigma \in \mathtt{Ev}}$ of global decision formulas, and $\varphi \in L_\mu$, the *$B$-adjustment* of $\varphi$, written $\varphi * B$ is defined by induction on $\varphi$ by:

$$\top * B = \top, \qquad p * B = p, \qquad X * B = X$$
$$(\varphi \vee \varphi') * B = (\varphi * B) \vee (\varphi' * B), (\neg\varphi) * B = \neg(\varphi * B)$$
$$(\langle\sigma\rangle\varphi) * B = B_\sigma \wedge (\varphi * B), (\mu X.\varphi) * B = \mu X.(\varphi * B)$$

*Realizability* can now be revised;

*Definition 9:* **L-Realizability.** Given a sentence $\varphi \in L_\mu$ and an logically defined architecture $\mathtt{LArch} = (\{\varphi_i\}_{i \in \mathcal{I}}, B)$, we say that $\varphi$ is $\mathtt{LArch}$*-realizable* on $\mathcal{S}$ if

$$\mathcal{S} \models \exists C_1(\Sigma_{o,1}) ... \exists C_n(\Sigma_{o,n}).(\bigwedge_{i \in \mathcal{I}} \varphi_i) \wedge (\varphi * B) \quad \text{(REAL)}$$

*Example 1:* Examples of architectures from [13], [8] are:

1) Conjunctive-Permissive $\mathtt{LArch}_{CP}$

$$\begin{array}{l} \varphi_i^P = \mathbf{Inv}(\bigwedge_{\sigma \notin \Sigma_{c,i}} c_i^\sigma) \\ B_\sigma = \bigwedge_{i \in \mathcal{I}} c_i^\sigma \end{array} \quad \text{(L-CP)}$$

$\varphi_i^P$ expresses that the propositions $c_i^\sigma$'s always hold provided $\sigma$ is not controlled by $i$.

2) Disjunctive-Antipermissive $\mathtt{LArch}_{DA}$

$$\begin{array}{l} \varphi_i^A = \mathbf{Inv}(\bigwedge_{\sigma \notin \Sigma_{c,i}} \neg c_i^\sigma) \\ B_\sigma = \bigvee_{i \in \mathcal{I}} c_i^\sigma \end{array} \quad \text{(L-DA)}$$

3) Mixed $\mathtt{LArch}_{CPDA(I)}$

$$\begin{array}{l} \varphi_i = \varphi_i^P \text{ if } i \in I, \varphi_i^A \text{ otherwise} \\ B_\sigma = (\bigwedge_{i \in I} c_i^\sigma) \vee (\bigvee_{j \notin I} c_j^\sigma) \end{array} \quad \text{(L-CPDA(I))}$$

4) Below-K $\mathtt{LArch}_K$

$$B_\sigma = \bigwedge_J (\bigvee_{j \in J} c_j^\sigma) \text{ where } J \subseteq \mathcal{I} \text{ and } |J| = k_\sigma \quad \text{(L-K)}$$

The correctness of the logical approach is stated by the following theorem:

*Theorem 10:* Given $\varphi \in L_\mu$, for all process $\mathcal{S}$, we have: $\varphi$ is $\mathtt{Arch}$-realizable on $\mathcal{S}$ if and only if $\varphi$ is $\mathtt{LArch}$-realizable on $\mathcal{S}$, where $\mathtt{Arch}$ ranges over the architectures presented in Sec.III and $\mathtt{LArch}$ is its logically defined counterpart given in the Ex.1.

The rest of the section is dedicated to the proof of Theo. 10; because of a lack of space, we will only discuss the cases of conjunctive-permissive and disjunctive-antipermissive architectures.

*Definition 11:* $B$-**Pruning.** Given a set $B = \{B_\sigma\}_{\sigma\in\mathtt{Ev}}$ of global decision formulas (each $B_\sigma$ based on the propositions $c_i^\sigma$'s), and given a process $\mathcal{R} = \langle R, r^0, \longrightarrow, \lambda\rangle$ labeled over $\{c_i^\sigma\}_{\mathtt{Ev},\mathcal{I}}$, the *$B$-pruning of $\mathcal{R}$* is the process $\mathcal{R}_{B\rightarrow} = \langle R, \epsilon, \underset{B}{\longrightarrow}, \lambda\rangle$ where $r \underset{B}{\overset{\sigma}{\longrightarrow}} r'$ whenever $r \in \llbracket B_\sigma \rrbracket_{\mathcal{R}}$ and $r \xrightarrow{\sigma} r'$

Now, combining Def.8 and Def.11 leads to the following fundamental proposition:

*Proposition 12:* Given a sentence $\varphi \in L_\mu$, a process $\mathcal{R}$, and a set $B = \{B_\sigma\}_{\sigma\in\mathtt{Ev}}$ of global decision formulas. Then, $\theta \in \llbracket \varphi \rrbracket_{\mathcal{R}_{B\rightarrow}}$ if and only if $\theta \in \llbracket \varphi * B \rrbracket_{\mathcal{R}}$.

*Proof:* By induction over $\varphi$, where the only interesting case is the one of $<\!\sigma\!>\varphi$: assume $\theta \in \llbracket <\!\sigma\!>\varphi \rrbracket_{\mathcal{R}_{B\rightarrow}}$, we prove that $\theta \in \llbracket (<\!\sigma\!>\varphi) * B \rrbracket_{\mathcal{R}} = \llbracket B_\sigma \wedge <\!\sigma\!>(\varphi * B) \rrbracket_{\mathcal{R}}$, that is $\theta \in \llbracket B_\sigma \rrbracket_{\mathcal{R}}$ and $\theta \in \llbracket <\!\sigma\!>(\varphi * B) \rrbracket_{\mathcal{R}}$. By definition, if $\theta \in \llbracket <\!\sigma\!>\varphi \rrbracket_{\mathcal{R}_{B\rightarrow}}$, then $\theta' = \theta\sigma$ is a trace of $\mathcal{R}_{B\rightarrow}$ s.t. $\theta' \in \llbracket \varphi \rrbracket_{\mathcal{R}_{B\rightarrow}}$. Because $\theta'$ is a trace of $\mathcal{R}_{B\rightarrow}$, by Def.11, we necessarily have $\theta \in \llbracket B_\sigma \rrbracket_{\mathcal{R}}$. Moreover, by induction on $\varphi$, $\theta' \in \llbracket \varphi \rrbracket_{\mathcal{R}_{B\rightarrow}}$ implies $\theta' \in \llbracket \varphi * B \rrbracket_{\mathcal{R}}$, which concludes. The reciprocal is similar. ■

We now prove Theo.10 for the conjunctive-permissive architecture:

Assume $\varphi$ is $\mathtt{LArch}_{CP}$-realizable on $\mathcal{S}$. Then there exist $\mathcal{E}_i \in LProc_{\Sigma_{o,i}}(C_i)$, $(i \in \mathcal{I})$, s.t. $\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n$ satisfies $\bigwedge_{i\in\mathcal{I}} \mathbf{Inv}(\bigwedge_{\sigma\notin\Sigma_{c,i}} c_i^\sigma) \wedge (\varphi * \{\bigwedge_{i\in\mathcal{I}} c_i^\sigma\}_{\sigma\in\mathtt{Ev}})$.

From $\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n \models \mathbf{Inv}(\bigwedge_{\sigma\notin\Sigma_{c,i}} c_i^\sigma)$ for each fixed $i \in \mathcal{I}$, and by the semantics of $\mathbf{Inv}(.)$, we have $\theta \in \llbracket c_i^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}$, for any $\theta \in Tr(\mathcal{S})$ and all $\sigma \notin \Sigma_{c,i}$,.

Now, define the decision rule $f_i : \mathtt{Ev}^* \rightarrow 2^{\mathtt{Ev}}$ by: for all $\sigma \in \mathtt{Ev}$, $\sigma \in f_i(\theta)$ iff $\theta \in \llbracket c_i^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}$. Henceforth, $f_i$ is conform to the $i$-local decision rule of Eq.(CP). Moreover, since $\mathcal{E}_i \in LProc_{\Sigma_{o,i}}(C_i)$, $f_i$ is consistent with $\sim_i$ (the observation equivalence of the controller $i$). Finally, by Prop.12, $(\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n)_{B\rightarrow} \models \varphi$. Now, using the fact that the processes $(\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n)_{B\rightarrow}$ and $\mathcal{S} \otimes (\mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n)_{B\rightarrow}$ are isomorphic, we define the global decision rule $f : \mathtt{Ev}^* \rightarrow 2^{\mathtt{Ev}}$ by: for all $\sigma \in \mathtt{Ev}$, $\sigma \in f(\theta)$ iff $\theta\sigma \in Tr(\mathcal{S} \otimes (\mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n)_{B\rightarrow})$. It is easy to prove that $f$ is a global decision rule satisfying Eq.(CP).

As $\mathcal{S}_{|f}$ and $\mathcal{S} \otimes (\mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n)_{B\rightarrow}$ have the same set of traces, $\mathcal{S}_{|f} \models \varphi$, leading to the conclusion that $\varphi$ is $\mathtt{Arch}_{CP}$-realizable on $\mathcal{S}$.

Conversely, assume that $\varphi$ is $\mathtt{Arch}_{CP}$-realizable on $\mathcal{S}$. Then, there exists some decentralized control $F = (\{f_i\}_{i\in\mathcal{I}}, f) \in \mathtt{Arch}_{CP}$ where $f(\theta) = \bigcap_{i\in\mathcal{I}} f_i(\theta)$, for all trace $\theta \in Tr(\mathcal{S})$, s.t. $\mathcal{S}_{|F} \models \varphi$. For each $i \in \mathcal{I}$, define $\mathcal{E}_i = \langle Tr(\mathcal{S}), \epsilon, \rightarrow_i, \lambda_i\rangle$ the complete process of type $\Sigma_{o,i}$ with $\lambda_i(\theta) = \{c_i^\sigma \,|\, \sigma \in f_i(\theta)\}$. By construction, $\mathcal{E}_i \in LProc_{\Sigma_{o,i}}(C_i)$ with possibly unreachable states; notice that $\mathcal{E}_i$ can be made finite provided $f_i^{-1}(\Sigma')$ is a regular sublanguage of $\mathtt{Ev}^*$, for all $\Sigma' \in 2^{\mathtt{Ev}}$, or equivalently if the local decision rule $f_i$ has a bounded memory.

Because the label $c_i^\sigma$ comes from $\mathcal{E}_i$, $\llbracket c_i^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n} = \llbracket c_i^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_i}$, for any $i$ and $\sigma$. By definition of $\mathcal{E}_i$, each $\theta \in Tr(\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n)$ is labeled by $c_i^\sigma$ whenever $\sigma \notin \Sigma_{c,i}$, since $\sigma \in f_i(\theta)$. As a conclusion, $\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n \models \bigwedge_{i\in\mathcal{I}} \mathbf{Inv}(\bigwedge_{\sigma\notin\Sigma_{c,i}} c_i^\sigma)$.

Now, for each event $\sigma$, define $B_\sigma^f = \bigwedge_{i\in\mathcal{I}} c_i^\sigma$ if $\sigma \in \Sigma(\mathcal{S})$, and by $\neg\top$ otherwise. We prove that for all $\varphi \in L_\mu$ (not only sentences),

$$\theta \in \llbracket \varphi \rrbracket_{\mathcal{S}_{|f}}^{val} \text{ implies } \theta \in \llbracket \varphi * B^f \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}^{val} \qquad (1)$$

which, when $\theta = \epsilon$ and when $\varphi$ is a sentence, entails $\mathcal{S} \otimes \mathcal{E}_1 \otimes \ldots \otimes \mathcal{E}_n \models \varphi * B^f$.
The proof of (1) is conducted by induction over $\varphi$; the only difficult cases are formulas $<\!\sigma\!>\varphi$ and $\neg\varphi$.

Remark that $\theta \in \llbracket <\!\sigma\!>\varphi \rrbracket_{\mathcal{S}_{|f}}^{val}$ means on the one hand,

$$\sigma \in \Sigma(\mathcal{S}) \text{ and } \theta\sigma \in Tr(\mathcal{S}_{|f}) \qquad (2)$$

and on the other hand,

$$\theta\sigma \in \llbracket \varphi \rrbracket_{\mathcal{S}_{|f}}^{val} \qquad (3)$$

From (2), $\sigma \in f(\theta) = \bigcap_i f_i(\theta)$, and by definition of the labelings $\lambda_i$'s,

$$\theta \in \llbracket \bigwedge_{i\in\mathcal{I}} c_i^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}^{val} = \llbracket B_\sigma^f \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}^{val} \qquad (4)$$

Finally, by applying the induction hypothesis on (3),

$$\theta\sigma \in \llbracket \varphi * B^f \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}^{val} \qquad (5)$$

Now, from (5) and (4), $\theta \in \llbracket (<\!\sigma\!>\varphi) * B^f \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}$.

The case of $\neg\varphi$ needs being check carefully as (1) is not an equivalence, but we omit the proof here, but $\neg$ symbols should be pushed innermost in order to consider propositions and their negations.

We now consider the disjunctive-antipermissive architecture and we sketch the proof that $\varphi$ is $\mathtt{Arch}_{DA}$-realizable on $\mathcal{S}$ if and only if $\varphi$ is $\mathtt{LArch}_{DA}$-realizable on $\mathcal{S}$. The proof follows the same line as for the conjunctive-permissive case.

First, we derive a decentralized control $F = (\{f_i\}_{i\in\mathcal{I}}, f) \in \mathtt{Arch}_{DA}$ from a logically defined architecture $\mathtt{LArch}_{DA} = (\{\varphi_i\}_{i\in\mathcal{I}}, B)$, where the $\varphi_i$'s and $B$ respect Eq.(L-DA), s.t.$\mathcal{S} \models \exists C_1(\Sigma_{o,1}) \ldots \exists C_n(\Sigma_{o,n}).(\bigwedge_{i\in\mathcal{I}} \varphi_i) \wedge \varphi * B$D. Define the $j$-local decision rules $f_j$'s by $\sigma \in f_j(\theta)$ iff $\theta \in \llbracket c_j^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}$. Now by Eq.(L-DA), for all $\theta \in Tr(\mathcal{S})$, $\theta \in \llbracket c_j^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}$ whenever $\sigma \in \Sigma_{uc}$, which makes $f_j$ a decision rule. Moreover, since for all $\sigma \in \Sigma_{c,k}$ $(k \neq j)$, we have $\theta \in \llbracket \neg c_j^\sigma \rrbracket_{\mathcal{S}\otimes\mathcal{E}_1\otimes\ldots\otimes\mathcal{E}_n}$, we obtain $\sigma \notin f_j(\theta)$, and prove Eq.(DA). Defining $f^B$ by $f^B(\theta) = \bigcup_{j\in\mathcal{I}} f_j(\theta)$ renders a decision rule s.t. $\mathcal{S}_{|f^B} \models \varphi$.

Secondly, from the assumption that some solution $F = (\{f_i\}_{i\in\mathcal{I}}, f) \in \mathtt{Arch}_{DA}$ exists for $\varphi$ and $\mathcal{S}$, we can build labeling-processes $\mathcal{E}_i$'s, and formulas $\varphi_i$ $(i \in \mathcal{I})$ and $B_\sigma$ $(\sigma \in \mathtt{Ev})$ so that Eq.(REAL) holds.

The proofs for the Mixed and the Below-K architectures are very similar.

## VI. CONCLUSION AND DISCUSSION

We have presented a second order logic to specify the Realizability Problem of a decentralized control with architecture. A strong point in favor of the approach are clear definitions for both the notion of architecture and the statement of the problem.

The logically defined architectures rely, on the one hand, on branching-time formulas which characterize the local decision rules of the controllers, and on the other hand, on propositional formulas which denote the global decision rule. The logical view enables us to reduce the Realizability Problem to a verification problem of a second order formula over the process under control; this was proved for standard architectures of the literature [3], [12], [8], [13], which logical counterparts emerge in a very natural and uniform manner. It is worthwhile noting that new kinds of architectures can be directly specified in the logic, and we believe it is very natural.

Since by [2], the Realizability Problem for the conjunctive-permissive architecture is undecidable, as it reduces to the Post Correspondence Problem [7]. An immediate corollary of Theo.10 is the undecidability of the model-checking of $\mathtt{Obs}L_\mu$ statements, as already noticed by [6]. Nevertheless, we can take advantage of the logical framework to derive a decision procedure for the Realizability Problem, provided the controllers can observe all event from the system, but whatever the architecture can be.

*Theorem 13:* **[10] Realizability and Synthesis.** Given any architecture $\mathtt{LArch} = (\{\varphi_i\}_{i\in\mathcal{I}}, B)$, any process $\mathcal{S}$ of type $\Sigma$, and any sentence $\varphi \in L_\mu$, it can be decided whether

$$\mathcal{S} \models \exists C_1(\Sigma)\dots\exists C_n(\Sigma).(\wedge_{i\in\mathcal{I}}\varphi_i) \wedge (\varphi * B) \quad \text{(FullObs)}$$

In the affirmative, we can synthesize a finite solution, namely finite processes $\mathcal{E}_1, \dots, \mathcal{E}_n$ s.t. $\mathcal{S}\otimes\mathcal{E}_i \models \varphi_i$, $(i \in \mathcal{I})$, and $\mathcal{S} \otimes (\mathcal{E}_1 \otimes \dots \otimes \mathcal{E}_n)_{B\to} \models \varphi$.

The arguments are the following: the formula in (FullObs) belongs to the logic of [10] which model-checking is decidable; we briefly explain the principles of the algorithm. According to [4], from any Mu-calculus sentence, we can construct a parity tree automaton which accepts all processes satisfying this formula; moreover, if a process is accepted by an automaton, then a finite process is also accepted by this automaton. Consider the automaton of the formula $(\bigwedge_{i\in\mathcal{I}}\varphi_i) \wedge (\varphi * B)$ - remark the propositions $c_i^\sigma$'s. By [10], this automaton can be projected to abstract from the $c_i^\sigma$'s, so that the resulting accepts the models of (FullObs). Still from [10], a $\{c_i^\sigma\}_{\mathtt{Ev},\mathcal{I}}$-labeling finite process $\mathcal{E}$ of type $\Sigma$ can be synthesized s.t. $\mathcal{S}\otimes\mathcal{E} \models (\bigwedge_{i\in\mathcal{I}}\varphi_i) \wedge (\varphi * B)$.

As far as we are concern in Theo. 13, this computation amounts to synthesize in a single step the process $\mathcal{E} = \mathcal{E}_1 \otimes \dots \otimes \mathcal{E}_n$ with $\mathcal{S}\otimes\mathcal{E} \models \varphi_i, \forall i$ and $\mathcal{S}\otimes\mathcal{E}_{B\to} \models \varphi$. By defining $\mathcal{E}_i$ as the process $\mathcal{E}$ where the $c_j^\sigma$'s $(j \neq i)$ are removed, and because the propositions in $\varphi_i$ range over $\{c_i^\sigma\}_{\sigma\in\Sigma}$, we get the right objects $\mathcal{E}_1, \dots, \mathcal{E}_n$ to conclude.

Complexity of (FullObs) is EXPTIME-complete: the synthesis procedure of [10] based on parity tree automata is EXPTIME. The algorithm is optimal: indeed, (FullObs) is EXPTIME-hard because the Realizability Problem for the universal process under control with the trivial architecture of a single controller contains the satisfiability problem of Mu-calculus sentences [4].

Future work mainly will focus on investigating the logically defined architectures and their properties: for example their expressiveness, their comparison, preorders between them s.a. independence. Notice that the framework is very easy to use; for example, general kinds of architectures can be considered, with global decision formulas being temporal rather than only propositional, e.g. expressing that a given event cannot be disallowed twice in a row by the same controller. We expect a generalization of Theo.10 for which we already know that a single general proof can be given, provided the decision rules have some “regular” feature.

## REFERENCES

[1] A. Arnold and D. Niwinski, *Rudiments of mu-calculus.* North-Holland, 2001.

[2] A. Arnold, A. Vincent, and I. Walukiewicz, “Games for synthesis of controllers with partial observation,” *Theoretical Computer Science*, vol. 1, pp. 7–34, 2003.

[3] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya, “Supervisory control of discrete-event processes with partial observations,” *IEEE transactions on automatic control*, vol. 33, no. 3, pp. 249–260, 1988.

[4] E. A. Emerson and C. S. Jutla, “Tree automata, mu-calculus and determinacy,” in *Proceedings 32nd Annual IEEE Symp. on Foundations of Computer Science, FOCS'91, San Jose, Puerto Rico, 1–4 Oct 1991.* Los Alamitos, California: IEEE Computer Society Press, 1991, pp. 368–377.

[5] Kozen, “Results on the propositional $\mu$-calculus,” *Theoretical Computer Science*, vol. 27(3), pp. 333–354, 1983.

[6] S. Pinchinat and S. Riedweg, “A decidable class of problems for control under partial observation,” *to appear in ”Information Processing Letters”*, vol. 95, pp. 454–460, 2005.

[7] E. Post, “A variant of a recursiveley unsolvable problem,” *Bulletin of the American Mathematical Society*, vol. 53, pp. 264–268, 1946.

[8] J. H. Prosser, M. Kam, and H. G. Kwatny, “Decision fusion and supervisor synthesis in decentralized discrete-event systems,” *Proceedings of the American control conference*, pp. 2251–2255, 1997.

[9] Ramadge and Wonham, “The control of discrete event systems,” in *Proc. of the IEEE*, vol. 77(1), 1989, pp. 81–98.

[10] Riedweg and Pinchinat, “Quantified mu-calculus for control synthesis,” in *Mathematical Foundations of Computer Science*, ser. LNCS, vol. 2747, 2003, pp. 642–651.

[11] S. Riedweg and S. Pinchinat, “A decidable class of problems for control under partial observation,” in *Proc. 2005 American Control Conference.*, Portland, Oregon, jun 2005.

[12] K. Rudie and W. M. Wonham, “Think globally, act locally : decentralized supervisory control,” *IEEE transactions on automatic control*, vol. 37, no. 11, pp. 1692–1708, 1992.

[13] T.-S. Yoo and S. Lafortune, “A general architecture for decentralized supervisory control of discrete-event systems,” *Discrete event dynamic systems : theory and applications*, 2002.