Modular and Decentralized Supervisory Control of Concurrent Discrete Event Systems Using Reduced System Models

Klaus Schmidt
Universität Erlangen-Nürnberg
Lehrstuhl für Regelungstechnik
Cauerstraße 7 - 91058 Erlangen, Germany
klaus.schmidt@rt.eei.uni-erlangen.de

Hervé Marchand
IRISA-INRIA Rennes
Campus univ de Rennes 1
35042 Rennes, France
herve.marchand@irisa.fr

Benoit Gaudin
FOKUS Fraunhofer
Kaiserin Augusta Allee 31
10589 Berlin, Germany
bga@fokus.fraunhofer.de

***Abstract*— This work investigates the supervisor synthesis for concurrent systems based on reduced system models with the intention of complexity reduction. It is assumed that the expected behavior (specification) is given on a subset of the system alphabet, and the system behavior is reduced to this alphabet. Supervisors are computed for each reduced subsystem employing the modular approach in [5] and the decentralized approach in [8]. Depending on the chosen architecture, we provide sufficient conditions for the consistent implementation of the reduced supervisors for the original system.**

Keywords–**Concurrent discrete event systems, hierarchical control, modular and decentralized architecture**.

I. Introduction

The main issue in supervisor synthesis for discrete event systems (DES) is the state-space explosion for large-scale systems. Addressing this problem, recent approaches study *hierarchical*, *decentralized* and *modular* methods to reduce the complexity of supervisor synthesis algorithms.

In hierarchical architectures [17], [3], [7], [14], [9], [13], controller synthesis is based on a plant abstraction (high-level model), which is supposed to be less complex than the original plant model (low-level model).

The structure of concurrent systems (systems modeled by several components) is exploited for decentralized and modular control. In most of the decentralized architectures [15], [10], [1], [11], [6], [8], the methodology is characterized by the fact that the specification (i.e. the expected behavior) can be decomposed according to the structure of the plant. In that case, local modular supervisors operating each concurrent system component individually are implemented, and necessary and sufficient conditions under which the behavior of the controlled plant corresponds to the supremal one are given. In contrast, the authors of [4], [5] consider a modular architecture. The specification does not need to be separable (but *locally consistent* and prefix-closed, which is not the case for most of the previously mentionned works). Modular supervisors can be computed based on the specification and abstractions of the subsystems so that they solve the supervisory control problem without having to build the whole system.

In this paper, we elaborate two approaches for concurrent systems that both avoid the computation of the overall system and are based on a reduced system model. We assume that the specification is given on a subset of the system alphabet and the behavior of the concurrent systems is reduced to this alphabet. Supervisors are synthesized for the reduced system models using the modular approach in [5] and the decentralized approach in [8]. We provide sufficient conditions for the consistent implementation of the reduced supervisors for the original system.

The outline of the paper is as follows. After providing basic definitions in supervisory control in Section II, we present the setting of the paper in Section III. Section IV and V discuss modular and structural decentralized control for reduced system models, respectively. Conclusions are given in Section VI.

II. Preliminaries

We recall basics from supervisory control theory [18], [2].

For a finite alphabet Σ, the set of all finite strings over Σ is denoted Σ^*. We write $s_1 s_2 \in \Sigma^*$ for the concatenation of two strings $s_1, s_2 \in \Sigma^*$. We write $s_1 \leq s$ when s_1 is a *prefix* of s, i.e. if there exists a string $s_2 \in \Sigma^*$ with $s = s_1 s_2$. The empty string is denoted $\varepsilon \in \Sigma^*$, i.e. $s\varepsilon = \varepsilon s = s$ for all $s \in \Sigma^*$. A *language* over Σ is a subset $H \subseteq \Sigma^*$. The *prefix closure* of H is defined by $\overline{H} := \{s_1 \in \Sigma^* | \exists s \in H \text{ s.t. } s_1 \leq s\}$. A language H is *prefix closed* if $H = \overline{H}$. Let $H, F \subseteq \Sigma^*$, then H is nonblocking w.r.t. F if $\overline{H} = \overline{H} \cap F$ [8].

The *natural projection* $p_i : \Sigma^* \to \Sigma_i^*$, $i = 1,2$, for the (not necessarily disjoint) union $\Sigma = \Sigma_1 \cup \Sigma_2$ is defined iteratively: (i) let $p_i(\varepsilon) := \varepsilon$; (ii) for $s \in \Sigma^*$, $\sigma \in \Sigma$, let $p_i(s\sigma) := p_i(s)\sigma$ if $\sigma \in \Sigma_i$, or $p_i(s\sigma) := p_i(s)$ otherwise. The set-valued inverse of p_i is denoted $p_i^{-1} : \Sigma_i^* \to 2^{\Sigma^*}$, $p_i^{-1}(t) := \{s \in \Sigma^* | p_i(s) = t\}$. The *synchronous product* $H_1 || H_2 \subseteq \Sigma^*$ of two languages $H_i \subseteq \Sigma_i^*$ is $H_1 || H_2 = p_1^{-1}(H_1) \cap p_2^{-1}(H_2) \subseteq \Sigma^*$.

A *finite automaton* is a tuple $G = (X, \Sigma, \delta, x_0, X_m)$, with the finite set of *states* X; the finite alphabet of *events* Σ; the partial *transition function* $\delta : X \times \Sigma \to X$; the *initial state* $x_0 \in X$; and the set of *marked states* $X_m \subseteq X$. We write $\delta(x,\sigma)!$ if δ is defined at (x,σ). In order to extend δ to a partial function on $X \times \Sigma^*$, recursively let $\delta(x,\varepsilon) := x$ and $\delta(x, s\sigma) := \delta(\delta(x,s),\sigma)$, whenever both $x' = \delta(x,s)$ and $\delta(x',\sigma)!$. $L(G) := \{s \in \Sigma^* : \delta(x_0,s)!\}$ and $L_m(G) := \{s \in L(G) : \delta(x_0,s) \in X_m\}$ are the *closed* and *marked language*

generated by the finite automaton G, respectively. G is *nonblocking* if $\overline{L_{\mathrm{m}}(G)} = L(G)$, i.e. if each string in $L(G)$ is the prefix of a marked string in $L_{\mathrm{m}}(G)$. A formal definition of the synchronous composition of two automata G_1 and G_2 is given in e.g. [2]. Note that $L(G_1||G_2) = L(G_1)||L(G_2)$ and $L_m(G_1||G_2) = L_m(G_1)||L_m(G_2)$.

In a supervisory control context, we write $\Sigma = \Sigma_c \cup \Sigma_{\mathrm{uc}}$, $\Sigma_c \cap \Sigma_{\mathrm{uc}} = \emptyset$, to distinguish *controllable* (Σ_c) and *uncontrollable* (Σ_{uc}) events. A *control pattern* is a set γ, $\Sigma_{\mathrm{uc}} \subseteq \gamma \subseteq \Sigma$, and the set of all control patterns is denoted $\Gamma \subseteq 2^{\Sigma}$. A *supervisor* is a map $S\colon L(G) \to \Gamma$, where $S(s)$ represents the set of enabled events after the occurrence of string s; i.e. a supervisor can disable controllable events only. The language $L(S/G)$ generated by G under supervision S is iteratively defined by (i) $\varepsilon \in L(S/G)$ and (ii) $s\sigma \in L(S/G)$ iff $s \in L(S/G), \sigma \in S(s)$ and $s\sigma \in L(G)$. Thus, $L(S/G)$ represents the behavior of the *closed-loop system*. To take into account the marking of G, let $L_{\mathrm{m}}(S/G) := L(S/G) \cap L_{\mathrm{m}}(G)$.

A language H is said to be controllable w.r.t. $L(G)$ and Σ_{uc} if there exists a supervisor S such that $\overline{H} = L(S/G)$. The set of all languages that are controllable w.r.t. $L(G)$ is denoted $\mathcal{C}(L(G))$ and can be characterized by $\mathcal{C}(L(G)) = \{H \subseteq L(G) |\ \exists S \text{ s.t. } \overline{H} = L(S/G)\}$. Furthermore, the set $\mathcal{C}(L(G))$ is closed under arbitrary union. Hence, for every *specification* language E there uniquely exists a *supremal controllable sublanguage* of E w.r.t. $L(G)$ and Σ_{uc}, which is formally defined as $\kappa_{L(G)}(E, \Sigma_{\mathrm{uc}}) := \bigcup\{K \in \mathcal{C}(L(G)) |\ K \subseteq E\}$. A supervisor S that leads to a closed-loop behavior $\kappa_{L(G)}(E, \Sigma_{\mathrm{uc}})$ is said to be *maximally permissive*. A maximally permissive supervisor S can be implemented as an automaton S that generates $\kappa_{L(G)}(E, \Sigma_{\mathrm{uc}})$ such that $L(S/G) = L(S)||L(G)$. The latter can be computed from G and a generator of E. The notion of controllability is extended by the notion of *partial controllability* [4]. Let $M \subseteq L(G)$ be a prefix-closed language and let $\Sigma'_{\mathrm{uc}} \subseteq \Sigma_{\mathrm{uc}}$. The language $M' \subseteq M$ is partially controllable w.r.t. M, $L(G)$, Σ_{uc} and Σ'_{uc}, if (i) M' is controllable w.r.t Σ'_{uc} and $L(G)$ and (ii) M' is controllable w.r.t. Σ_{uc} and M. The unique supremal partially controllable sublanguage w.r.t. M, $L(G)$, Σ_{uc} and Σ'_{uc} is defined as $M^{\uparrow \mathrm{pc}} = \overline{M^{\uparrow \mathrm{pc}}} := \kappa_M\big(\kappa_{L(G)}(M, \Sigma'_{\mathrm{uc}}), \Sigma_{\mathrm{uc}}\big)$.

A language E is $L_{\mathrm{m}}(G)$-*closed* if $\overline{E} \cap L_{\mathrm{m}}(G) = E$ and the set of $L_{\mathrm{m}}(G)$-closed languages is denoted $\mathcal{F}_{L_{\mathrm{m}}(G)}$. The closed-loop system S/G is nonblocking under maximally permissive supervision for specifications $E \in \mathcal{F}_{L_{\mathrm{m}}(G)}$.

III. Setting

As a system model, we consider *concurrent* DES represented by finite automata $(\tilde{G}_i)_{1 \le i \le n}$ over the corresponding alphabets $\tilde{\Sigma}_i = \tilde{\Sigma}_{i,\mathrm{uc}} \dot{\cup} \tilde{\Sigma}_{i,\mathrm{c}}$. Here, $\tilde{\Sigma}_{i,\mathrm{uc}}$ and $\tilde{\Sigma}_{i,\mathrm{c}}$ denote the uncontrollable and the controllable events, respectively. We assume that all subsystems are directly or indirectly connected to all other subsystems via events from the set $\Sigma_{i,\mathrm{s}} := \cup_{k \neq i}(\tilde{\Sigma}_i \cap \tilde{\Sigma}_k)$ of *shared events*. The global set of shared events is thus given by $\Sigma_{\mathrm{s}} = \cup_i \Sigma_{i,\mathrm{s}}$.

The overall system model is $\tilde{G} := \|_i \tilde{G}_i$ over the alphabet $\tilde{\Sigma} := \cup_i \tilde{\Sigma}_i$. Moreover, we assume that the components that share an event agree on the control status of this event, i.e. $\forall i,k,\ \tilde{\Sigma}_{i,\mathrm{uc}} \cap \tilde{\Sigma}_{k,\mathrm{c}} = \emptyset$. Under this hypothesis, we have that $\tilde{\Sigma}_{\mathrm{uc}} = \cup_i \tilde{\Sigma}_{i,\mathrm{uc}}$ and $\tilde{\Sigma}_{\mathrm{c}} = \cup_i \tilde{\Sigma}_{i,\mathrm{c}}$.

The main objective of this paper is to study control architectures which reduce the computational complexity of supervisor synthesis for a given specification $\tilde{K} \subseteq \tilde{\Sigma}^*$ by avoiding the computation of $\tilde{G}$. To this end, we are interested in the case where the complexity of the specification $\tilde{K}$ is lower than that of the plant $\tilde{G}$. In the literature, there are different approaches tackling this problem.

An approach for the modular control of concurrent systems is proposed in [4], [5]. Modular supervisors are computed using abstractions of the decentralized subsystems and corresponding local specifications. The supremal partially controllable sublanguages of the local specifications solve the supervisory control problem if the specification $\tilde{K}$ is *locally consistent* and prefix-closed, and the languages of the subsystems are *mutually controllable*.

The method in [8] suggests structural decentralized control. It requires the specification $\tilde{K}$ to be *separable*, i.e. $\tilde{K} = \|_i \tilde{p}_i(\tilde{K})$, where $\tilde{p}_i : \tilde{\Sigma}^* \to \tilde{\Sigma}_i^*$ is the natural projection. If the languages of the subsystems are *mutually controllable* and *shared event marking*,[1] then using nonblocking local controllers for the specifications $\tilde{p}_i(\tilde{K})$ is equivalent to the nonblocking overall supervisor.

This approach is supplemented with hierarchical control in [14]. Monolithic control is applied to a reduced (hierarchical) system model which is derived by projecting the behavior of the original model to the set of shared events Σ_{s}. However, this approach requires the computation of an overall reduced system model which is not always feasible.

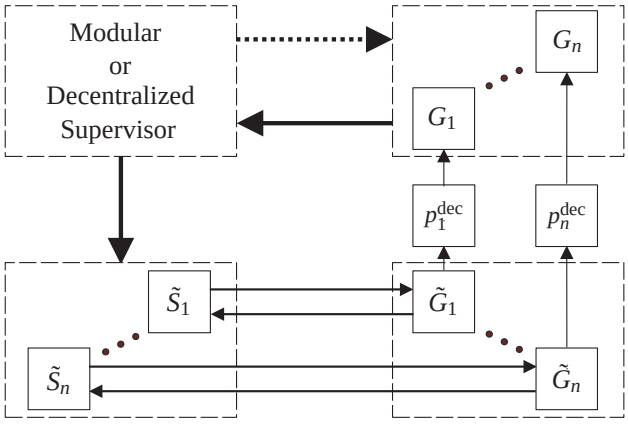

Fig. 1. Modular/Decentralized Architecture

Motivated by these considerations, we elaborate two methods that employ reduced concurrent system models, but avoid computing an overall reduced system model. To this end, we investigate the case where the specification $K \subseteq (\Sigma)^*$ for the supervisory control problem is given on a reduced

[1] Definitions of these notions are given in Section IV and V.

alphabet $\Sigma \subset \tilde{\Sigma}$ with $\Sigma_s \subseteq \Sigma$.[2] Hence, the reduction is based on projecting out events that occur in only one subsystem. With the reduced decentralized alphabets $\Sigma_i := \Sigma \cap \tilde{\Sigma}_i$ and the decentralized natural projections $p_i^{\text{dec}} : \tilde{\Sigma}_i^* \to \Sigma_i^*$, the decentralized reduced system models are $(G_i)_{1\le i\le n}$, where $L(G_i) = p_i^{\text{dec}}(L(\tilde{G}_i))$ and $L_m(G_i) = p_i^{\text{dec}}(L_m(\tilde{G}_i))$.

In the following sections, we utilize the approaches in [5] and [8] to design supervisors for the reduced system models. Based on these supervisors, we provide conditions for the decentralized supervisor implementation $\tilde{S}_i$ for the original systems. The first approach results in an estimation of the supremal controllable sublanguage of a prefix-closed non-separable specification. The second method provides an estimation of the supremal controllable and nonblocking sublanguage of a not necessarily prefix-closed but separable specification. Figure 1 illustrates the control scheme.

IV. Modular Control

According to [5], modular supervisors $S_i^{-1} : \Sigma^* \to \Gamma_i$ with the set of control patterns $\Gamma_i := \{\gamma \subseteq \Sigma | \Sigma_{i,\text{uc}} \subseteq \gamma\}$ are computed for the abstractions G_i^{-1} of the reduced system models and the local specifications $K_i^{-1} := K \cap L(G_i^{-1})$, where $L(G_i^{-1}) = p_i^{-1}(L(G_i)) \subseteq \Sigma^*$, with the natural projection $p_i : \Sigma^* \to \Sigma_i^*$. The main result of [5] is based on the following definitions.

Definition 4.1: G_i and G_k are mutually controllable if

1) $L(G_i)(\Sigma_{k,\text{uc}} \cap \Sigma_i) \cap p_i^{ik}((p_k^{ik})^{-1}(L(G_k))) \subseteq L(G_i)$
2) $L(G_k)(\Sigma_{i,\text{uc}} \cap \Sigma_k) \cap p_k^{ik}((p_i^{ik})^{-1}(L(G_i))) \subseteq L(G_k)$

where $p_i^{ik} : (\Sigma_i \cup \Sigma_k)^* \to \Sigma_i^*$ and $p_k^{ik} : (\Sigma_i \cup \Sigma_k)^* \to \Sigma_k^*$.

Mutual controllability ensures that after any execution of the system, the occurrence of a shared uncontrollable event is either allowed by every subsystem which shares it, or it is not allowed by any subsystem.

Definition 4.2 (Local consistency): A specification $K = \overline{K}$ is said to be locally consistent w.r.t. Σ_{uc} and $(L(G_i))_{1\le i\le n}$, if for any i we have: $\forall s \in K_i^{-1}$ and $\forall u \in \Sigma_{\text{uc}}^*$ such that $su \in K_i^{-1}$ and $\forall v \in \Sigma_{i,\text{uc}}^*$ it holds that $sp_i(u)v \in K_i^{-1} \Rightarrow suv \in K_i^{-1}$.

Based on the above definitions, it holds that the computation of modular supervisors implementing the supremal partially controllable sublanguages of K_i^{-1} is equivalent to the monolithic supervisor for the specification K.

Theorem 4.1 (Supervisor Computation [5]): Let $(G_i)_{1\le i\le n}$ be mutually controllable[3] and assume that $\forall i,k, \tilde{\Sigma}_{i,\text{uc}} \cap \tilde{\Sigma}_{k,\text{c}} = \emptyset$. If the specification $K = \overline{K} \subseteq \Sigma^*$ is locally consistent w.r.t. Σ_{uc} and $(L(G_i))_{1\le i\le n}$, then

$$\bigcap\nolimits_i (K_i^{-1})^{\uparrow \text{pc}} = \kappa_{L(G)}(K \cap L(G), \Sigma_{\text{uc}}).$$

Using the concept of modularity [4], the overall supervisor $S^{-1} : \Sigma^* \to \Gamma$ with $\Gamma := \{\gamma \subseteq \Sigma | \Sigma_{\text{uc}} \subseteq \gamma\}$ and $L(S^{-1}/G) = \bigcap_i (K_i^{-1})^{\uparrow \text{pc}}$ can now be implemented as the intersection of the control actions of the modular supervisors S_i^{-1} with $L(S_i^{-1}/G_i^{-1}) = (K_i^{-1})^{\uparrow \text{pc}}$ (see Figure 2).

[2]This assumption is no restriction. If $\Sigma_s - \Sigma \neq \emptyset$, $K' = K||(\Sigma_s - \Sigma)^* \subseteq (\Sigma_s \cup \Sigma)^*$ fulfills the requirement.

[3]It can be shown that the condition of global mutual controllability required in [5] is equivalent to mutual controllability.

However, the supervisors S_i^{-1} are computed based on the reduced system model. In what follows, we provide an implementation of these supervisors for the original systems. For this purpose, $p_i(L(S_i^{-1}/G_i^{-1}))$ is used as an approximation of the modular closed-loop behavior $L(S^{-1}/G) = ||_i L(S_i^{-1}/G_i^{-1})$ projected on the reduced subalphabets Σ_i. As shown in the next lemma, $p_i(L(S_i^{-1}/G_i^{-1}))$ is controllable w.r.t. $L(G_i)$ and $\Sigma_{i,\text{uc}}$. Thus it can be enforced by a supervisor for G_i.

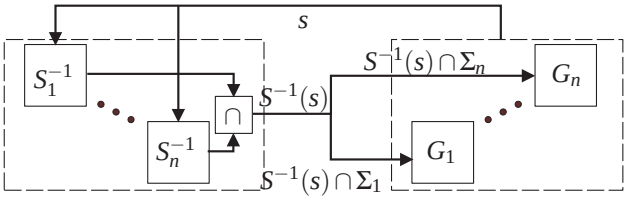

Fig. 2. Modular architecture

Lemma 4.1: With the preceding notations, $p_i(L(S_i^{-1}/G_i^{-1}))$ is controllable w.r.t $L(G_i)$ and $\Sigma_{i,\text{uc}}$.

Proof: Let us consider

$$s\sigma \in p_i(L(S_i^{-1}/G_i^{-1}))\Sigma_{i,\text{uc}} \cap L(G_i)$$

and $s' \in L(S_i^{-1}/G_i^{-1})$ such that $p_i(s') = s$. Since $s\sigma \in L(G_i)$ and $\sigma \in \Sigma_i$, it is also true that $p_i(s'\sigma) \in L(G_i)$. This entails that $s'\sigma \in L(G_i^{-1})$ and

$$s'\sigma \in L(S_i^{-1}/G_i^{-1})\Sigma_{i,\text{uc}} \cap L(G_i^{-1}).$$

But $L(S_i^{-1}/G_i^{-1})$ is controllable w.r.t $\Sigma_{i,\text{uc}}$ and $L(G_i^{-1})$ since it is partially controllable. Therefore, $s'\sigma \in L(S_i^{-1}/G_i^{-1})$ and $p_i(s'\sigma) = s\sigma \in p_i(L(S_i^{-1}/G_i^{-1}))$, which concludes the proof. ■

Now as $\forall i$, $p_i(L(S_i^{-1}/G_i^{-1})) \subseteq L(G_i)$ is controllable w.r.t. $L(G_i)$ and $\Sigma_{i,\text{uc}}$, there exist n supervisors $(S_i)_{1\le i\le n}$ such that

$$L(S_i/G_i) = p_i(L(S_i^{-1}/G_i^{-1})).$$

Based on the results in [12], an admissible supervisor for the original system is given with the *consistent implementations* $\tilde{S}_i : \tilde{\Sigma}_i^* \to \tilde{\Gamma}_i$ (see [12]) of the decentralized reduced supervisors S_i. It is defined for $s \in L(\tilde{G}_i)$ as $\tilde{S}_i(s) := S_i(p_i(s)) \cup (\tilde{\Sigma}_i - \Sigma_i)$. Note the equality $p_i^{\text{dec}}(L(\tilde{S}_i/\tilde{G}_i)) = L(S_i/G_i)$.

Combining the steps described above, the main result of this section can be stated.

Theorem 4.2: Recalling that $\Sigma_s \subseteq \Sigma$ and with the notation from above, the supervisor implementation

$$L(\tilde{S}/\tilde{G}) = \big(||_i L(S_i^{-1}/G_i^{-1})\big)||\big(||_i L(\tilde{S}_i/\tilde{G}_i)\big)$$

leads to consistent control of the original system, i.e.

$$\begin{aligned} p(L(\tilde{S}/\tilde{G})) &= L(S^{-1}/G), \\ L(\tilde{S}/\tilde{G}) &\subseteq p^{-1}(K), \end{aligned}$$

where $p : \tilde{\Sigma}^* \to \Sigma^*$.

The following Lemma aids the proof of Theorem 4.2.

Lemma 4.2 ([14]): Let $(L_i)_{1\le i\le n}$ be languages over the respective alphabets Σ_i. Assume that $\Sigma_0 \subseteq \cup_i\Sigma_i$ and $\cup_{i\neq k}(\Sigma_i\cap\Sigma_k)\subseteq\Sigma_0$ with the natural projections $p_0:(\cup_i\Sigma_i)^*\to\Sigma_0^*$ and $p_i':\Sigma_i^*\to(\Sigma_i\cap\Sigma_0)^*$ for $i=1,\ldots,n$. Then

$$p_0(\|_i L_i) = \|_i p_i'(L_i).$$

Proof of Theorem 4.2:

Now Theorem 4.2 can be proven. Because of Lemma 4.2,

$$\begin{aligned} p(L(\tilde S/\tilde G)) &= p\big(\big(L(S^{-1}/G)\big)||\big(\|_i L(\tilde S_i/\tilde G_i)\big)\big)\\ &= \big(L(S^{-1}/G)\big)||p\big(\|_i L(\tilde S_i/\tilde G_i)\big)\\ &= \big(\|_i L(S_i^{-1}/G_i^{-1})\big)||\big(\|_i p_i^{\mathrm{dec}}(L(\tilde S_i/\tilde G_i))\big). \end{aligned}$$

This can be written as $\big(\|_i L(S_i^{-1}/G_i^{-1})\big)||\big(\|_i L(S_i/G_i)\big)$ with $p_i^{\mathrm{dec}}(L(\tilde S_i/\tilde G_i)) = L(S_i/G_i)$.

Now according to Lemma 4.1, the previous equation can be rearranged as

$$\begin{aligned} p(L(\tilde S/\tilde G)) &= \big(\|_i(L(S_i^{-1}/G_i^{-1}))\,||\,\big(\|_i(L(S_i^{-1}/G_i^{-1})\big)\\ &= \|_i(L(S_i^{-1}/G_i^{-1}). \end{aligned}$$

Moreover, the supervisor computation implies that $\|_i L(S_i^{-1}/G_i^{-1}) \subseteq K$. This means that $L(\tilde S/\tilde G)\subseteq p^{-1}(K)$. ■

The reduced modular architecture is shown in Figure 3. The control actions of the decentralized supervisors for the original systems evaluate to $\tilde S_i(s_i) = p_i(S_i^{-1}(s))\cup(\tilde\Sigma_i-\Sigma_i)$.

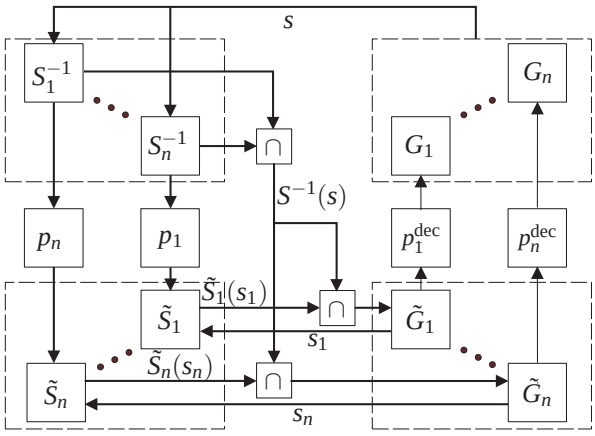

Fig. 3. Reduced modular architecture

So far, our method allows to perform local computations instead of building a single finite automaton for solving the supervisory control problem. However, only the case of prefix-closed specifications was considered. Based on a decentralized architecture, we now give sufficient conditions under which a nonblocking solution can be computed whenever the reduced specification is separable.

V. Structural Decentralized Control

Consider a concurrent system given by a set of nonblocking decentralized systems $(\tilde G_i)_{1\le i\le n}$ (i.e., $\forall i,\ \overline{L_{\mathrm m}(\tilde G_i)} = L(\tilde G_i)$). It follows that the reduced system models G_i are also nonblocking. We assume that the specification $K\subseteq\Sigma^*$ over the subalphabet Σ is separable, i.e. $K = \|_i p_i(K)$, where $p_i:\Sigma^*\to\Sigma_i^*$ and each local specification $K_i := p_i(K)$ is $L_{\mathrm m}(G_i)$-closed, i.e. $K_i\in\mathcal{F}_{L_{\mathrm m}(G_i)}$. Our aim is to use the methodology of [8] to compute nonblocking decentralized supervisors acting upon the subsystems G_i and to implement these supervisors for the original system $\tilde G = \|_i\tilde G_i$.

We first formally describe the approach in [8] and then provide new results that are useful in our setting.

Definition 5.1: Let $\Sigma'\subseteq\Sigma$ and $H\subseteq\Sigma^*$, then H marks Σ' whenever $\Sigma^*\Sigma'\cap\overline{H}\subseteq H\Sigma'$

Using the above definition combined with mutual controllability, Theorem 5.1 follows. The structural decentralized architecture is illustrated in Figure 4.

Theorem 5.1 ([8]): Let $(G_i)_{i\le n}$ be nonblocking subsystems and $K = \|_i K_i$ be the separable specification where $K_i\in\mathcal{F}_{L_{\mathrm m}(G_i)}$. Suppose that for $i,k\le n$ and $i\neq k$, $L_{\mathrm m}(G_i)$ marks $\Sigma_i\cap\Sigma_k$ and $L_{\mathrm m}(G_k)$ marks the same set, [4] and G_i and G_k are mutually controllable, then

1) $\|_i\,\overline{\kappa_{L(G_i)}(K_i)}\cap L(G) = \overline{\kappa_{L(G)}(K)}$
2) *$p_i(\overline{\kappa_{L(G)}(K)})$ is nonblocking with respect to $L_{\mathrm m}(G_i)$*

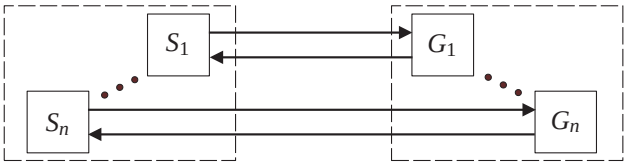

Fig. 4. Structural decentralized architecture

In addition to this result due to [8], one can prove that the overall closed-loop behavior is actually nonblocking. To do so, we first show that whenever the local specifications K_i are $L_{\mathrm m}(G_i)$-closed then so is the global specification K with respect to the reduced plant G.

Lemma 5.1: Let $(G_i)_{1\le i\le n}$ be the set of decentralized subsystems and $K_i\in\mathcal{F}_{L_{\mathrm m}(G_i)}$. Then $K=\|_i K_i\in\mathcal{F}_{L_{\mathrm m}(G)}$.

Proof: First we clearly have that $K\subseteq\overline{K}$. Now since $\forall i$, $K_i\subseteq L_{\mathrm m}(G_i)$, it holds that $\|_i K_i\subseteq\|_i L_{\mathrm m}(G_i)$ which entails that $K\subseteq L_{\mathrm m}(G)$. Therefore $K\subseteq\overline{K}\cap L_{\mathrm m}(G)$.

Reciprocally, consider $s\in\overline{K}\cap L_{\mathrm m}(G)$. We thus have that $p_i(s)\in p_i(\overline{K})$ and $p_i(s)\in p_i(L_{\mathrm m}(G))$.

- As $L_{\mathrm m}(G) = \cap_j(p_k^{-1}(L_{\mathrm m}(G_k))$, this entails that $p_i(s)\in p_i(p_i^{-1}(L_{\mathrm m}(G_i))) = L_{\mathrm m}(G_i)$.
- Let us now show that $p_i(s)\in\overline{K_i}$. First, we have that $\overline{K} = \overline{\|_i K_i} = \cap_i p_i^{-1}(K_i)\subseteq\cap_i p_i^{-1}(K_i)\subseteq p_i^{-1}(\overline{K_i})$.
  Also $p_i(s)\in p_i(\overline{K})\subseteq p_i(\cap_k(p_k^{-1}(\overline{K_k})))\subseteq\cap_k(p_i(p_k^{-1}(\overline{K_k})))$, and hence $p_i(s)\in p_i(p_i^{-1}(\overline{K_i})) = \overline{K_i}$.

Overall, $\forall i,\ p_i(s)\in\overline{K_i}\cap L_{\mathrm m}(G_i) = K_i$ as $K_i\in\mathcal{F}_{L_{\mathrm m}(G)}$. Thus $\forall i,\ s\in p_i^{-1}(K_i)$ and $s\in\cap_i(p_i^{-1}(K_i)) = K$. ■

[4] Note that it is equivalent to say that $\forall i,\ L_{\mathrm m}(G_i)$ marks $\Sigma_{i,\mathrm s}$.

We now need to show that the behavior of the closed-loop reduced system can be actually obtained by a collection of supervisors each of them acting upon a local decentralized subsystem G_i. This is the aim of the next lemma:

Lemma 5.2: With the preceding notations, we have that

1) $\|_i \overline{\kappa_{L(G_i)}(K_i)} = \overline{\kappa_{L(G)}(K)}$
2) $\|_i \kappa_{L(G_i)}(K_i) = \kappa_{L(G)}(K)$

Proof:

1) Due to Theorem 5.1, we have that

$$\begin{aligned}\overline{\kappa_{L(G)}(K)} &= \|_i \overline{\kappa_{L(G_i)}(K_i)} \cap L(G) \\ &= \|_i \overline{\kappa_{L(G_i)}(K_i)} \cap \|_i L(G_i) \\ &= \|_i \overline{\kappa_{L(G_i)}(K_i)} \cap L(G_i) \\ &= \|_i \overline{\kappa_{L(G_i)}(K_i)} \quad \text{as } \overline{\kappa_{L(G_i)}(K_i)} \subseteq L(G_i)\end{aligned}$$

2) Because of Lemma 5.1, we have

$$\begin{aligned}\kappa_{L(G)}(K) &= \overline{\kappa_{L(G)}(K)} \cap L_{\mathrm{m}}(G) \\ &= \|_i \overline{\kappa_{L(G_i)}(K_i)} \cap \|_i L_{\mathrm{m}}(G_i) \\ &= \|_i (\overline{\kappa_{L(G_i)}(K_i)} \cap L_{\mathrm{m}}(G_i)) \\ &= \|_i \kappa_{L(G_i)}(K_i) \text{ due to } L_{\mathrm{m}}(G_i)\text{-closure}\end{aligned}$$

■

With the above lemmas, the existence of a nonblocking supervisor for G can be shown.

Theorem 5.2: Under the assumptions of Theorem 5.1, the supervisor S such that $L(S/G) = \kappa_{L(G)}(K)$ is nonblocking

Proof: Based on Lemma 5.2, we consider nonblocking supervisors S_i such that $L(S_i/G_i) = \kappa_{L(G_i)}(K_i)$. It holds that

$$\begin{aligned}\kappa_{L(G)}(K) &= \|_i L(S_i/G_i) \\ \overline{\kappa_{L(G)}(K)} &= \|_i \overline{L(S_i/G_i)}\end{aligned}$$

Let us now consider S such that $S = \|_i S_i$, where each supervisor S_i is seen as a finite automaton. We have that

$$S/G = S\|G = (\|_i S_i)\|(\|_i G_i) = \|_i (S_i\|G_i) = \|_i S_i/G_i$$

We thus have that $L_{\mathrm{m}}(S/G) = \kappa_{L(G)}(K)$ and $L(S/G) = \overline{\kappa_{L(G)}(K)}$

Now, based on Lemma 5.1, we know that K is $L_{\mathrm{m}}(G)$-closed. Moreover, $L_{\mathrm{m}}(G)$-closure is preserved under control, which ensures us that the overall closed-loop decentralized system $\|_i(S_i/G_i) = (\|_i S_i)/G$ is nonblocking. ■

Remark 1: It is interesting to note that this result gives sufficient conditions under which a concurrent system is nonblocking. Indeed, based on Theorem 5.1 and 5.2, given a concurrent system $G = \|_i G_i$, if $L_{\mathrm{m}}(G_i)$ marks Σ_s and $\forall i \neq j$, G_i and G_j are mutually controllable, then G is nonblocking. This gives access to an efficient way to test if a concurrent system is nonblocking.

Next, the implementation of the supervisors computed with respect to the reduced system models for the original system is discussed. We again suggest the consistent implementation, and investigate two different sets of conditions which guarantee nonblocking and consistent control.

In the first case, the original subsystems have to mark the reduced alphabets in addition to the conditions which are required for nonblocking supervisor synthesis for the reduced system model.

Theorem 5.3: Let K be a separable specification and let S_i be such that $L_{\mathrm{m}}(S_i/G_i) = \kappa_{L(G_i)}(p_i(K), \Sigma_{i,\mathrm{uc}})$. Assume that G_i and G_k are mutually controllable for $i \neq k$ and $L_{\mathrm{m}}(\tilde{G}_i)$ marks Σ_i forall $i = 1, \ldots, n$. If the supervisors $\tilde{S}_i$ are consistent implementations of S_i, then the overall supervisor $\tilde{S}$ such that

$$L(\tilde{S}/\tilde{G}) := \|_i L(\tilde{S}_i/\tilde{G}_i)$$

is nonblocking and consistent.
First we need the following lemma.

Lemma 5.3 ([12]): The consistent implementation implies that if $s_i \in L(\tilde{S}_i/\tilde{G}_i)$ and $s_i u_i \in L(\tilde{G}_i)$ for $u_i \in (\tilde{\Sigma}_i - \Sigma_i)^*$, then $s_i u_i \in L(\tilde{S}_i/\tilde{G}_i)$. If additionally $s_i u_i \sigma \in L(\tilde{G}_i)$ for $\sigma \in \Sigma_i$ and $p_i^{\mathrm{dec}}(s_i u_i)\sigma \in L(S_i/G_i)$, then $s_i u_i \sigma \in L(\tilde{S}_i/\tilde{G}_i)$.

Based on this lemma, the proof of Theorem 5.3 is as follows:

Proof: For showing consistency, we observe that $p(L(\tilde{S}/\tilde{G})) = p(\|_i L(\tilde{S}_i/\tilde{G}_i)) = \|_i p_i^{\mathrm{dec}}(L(\tilde{S}_i/\tilde{G}_i)) = \|_i L(S_i/G_i) = L(S/G)$. Also, because of $L(S/G) \subseteq K$, it holds that $L(\tilde{S}/\tilde{G}) \subseteq p^{-1}(K)$.

For proving nonblocking control, it has to be shown that if $s \in L(\tilde{S}/\tilde{G})$, then $s \in \overline{L_{\mathrm{m}}(\tilde{S}/\tilde{G})}$. Now assume that $s \in L(\tilde{S}/\tilde{G})$. Then $\tilde{p}_i(s) \in L(\tilde{S}_i/\tilde{G}_i)$ for all $i = 1, \ldots, n$. Suppose that there is no $u_i \in (\tilde{\Sigma}_i - \Sigma_i)^*$ s.t. $\tilde{p}_i(s)u_i \in L_{\mathrm{m}}(\tilde{G}_i)$. As $\tilde{G}_i$ is nonblocking, there must be a string $v_i = \hat{v}_i \sigma \tilde{v}_i \in (\tilde{\Sigma}_i - \Sigma_i)^* \Sigma_i \tilde{\Sigma}_i^*$ s.t. $\tilde{p}_i(s)v_i \in L_{\mathrm{m}}(\tilde{G}_i)$. But as $L_{\mathrm{m}}(\tilde{G}_i)$ marks Σ_i, $\tilde{p}_i(s)\hat{v}_i \in L_{\mathrm{m}}(\tilde{G}_i)$, which contradicts the assumption. As i was chosen arbitrarily, it is true that $\forall i$, there is a $u_i \in (\tilde{\Sigma}_i - \Sigma_i)^*$ s.t. $\tilde{p}_i(s)u_i \in L_{\mathrm{m}}(\tilde{G}_i)$. Hence, for example the string $su_1 \cdots u_n \in \|_i \tilde{p}_i(s)u_i \subseteq \|_i L_{\mathrm{m}}(\tilde{G}_i) = L_{\mathrm{m}}(\tilde{G})$. Now using Lemma 5.3, we also have that $\tilde{p}_i(s)u_i \in L(\tilde{S}_i/\tilde{G}_i)$, $\forall i$, wich entails that $su_1 \cdots u_n \in \|_i \tilde{p}_i(s)u_i \subseteq \|_i L_i(\tilde{S}_i/\tilde{G}_i) = L(\tilde{S}/\tilde{G})$. Thus $su_1 \cdots u_n \in L_{\mathrm{m}}(\tilde{G}) \cap L(\tilde{S}/\tilde{G}) = L_{\mathrm{m}}(\tilde{S}/\tilde{G})$ and thus $s \in \overline{L_{\mathrm{m}}(\tilde{S}/\tilde{G})}$. ■

The second case is based on the notion of an $H-$observer.

Definition 5.2 (H-observer): Let $H \subseteq L = \overline{L} \subseteq \tilde{\Sigma}^*$ be languages and $p : \tilde{\Sigma}^* \to \Sigma^*$ be the natural projection on the alphabet $\Sigma \subseteq \tilde{\Sigma}$. p is called an $H-$observer if $\forall s \in L$ and $\forall \sigma \in (\Sigma \cup \{\varepsilon\})$:

$$p(s)\sigma \in p(H) \Rightarrow \exists u \in \tilde{\Sigma}^* \text{ s.t. } su \in H \wedge p(su) = p(s)\sigma.$$

In Theorem 5.4, the condition that all events in Σ_i must mark $L_{\mathrm{m}}(G_i)$ is reduced to the events in $\Sigma_{i,\mathrm{s}}$. This is compensated by requiring the decentralized projection p_i^{dec} to be a $L_{\mathrm{m}}(\tilde{G}_i)$-observer.[5]

Theorem 5.4: Let K be a separable specification and let S_i be such that $L_{\mathrm{m}}(S_i/G_i) = \kappa_{L(G_i)}(p_i(K), \Sigma_{i,\mathrm{uc}})$. Assume that G_i and G_k are mutually controllable for $i \neq k$ and $L_{\mathrm{m}}(G_i)$ marks $\Sigma_{i,\mathrm{s}}$. If p_i^{dec} is a $L_{\mathrm{m}}(\tilde{G}_i)$-observer and the supervisors $\tilde{S}_i$ are consistent implementations of S_i, then the overall

[5] If p_i^{dec} is not a $L_{\mathrm{m}}(\tilde{G}_i)$-observer, then [16] provides an algorithm to compute a $L_{\mathrm{m}}(\tilde{G}_i)$-observer with the coarsest equivalence kernel possible.

supervisor $\tilde{S}$ such that $L(\tilde{S}/\tilde{G}) := \|_i L(\tilde{S}_i/\tilde{G}_i)$ is nonblocking and consistent.
Lemma 5.4 supports the proof of Theorem 5.4.

Lemma 5.4: With the assumptions in Theorem 5.4, it holds that if $s \in L(\tilde{S}_i/\tilde{G}_i)$ and $p_i^{\mathrm{dec}}(s)t \in L_{\mathrm{m}}(S_i/G_i)$ for $t \in \Sigma_i^*$, then $\exists u_i \in \tilde{\Sigma}_i^*$ s.t. $su_i \in L_{\mathrm{m}}(\tilde{S}_i/\tilde{G}_i)$ and $p_i^{\mathrm{dec}}(su_i) = p_i^{\mathrm{dec}}(s)t$.

Proof: Assume that $s \in L(\tilde{S}_i/\tilde{G}_i)$ and $p_i^{\mathrm{dec}}(s)t \in L_{\mathrm{m}}(S_i/G_i)$ for $t \in \Sigma_i^*$. There are two cases.
1. $t = \varepsilon$. As p_i^{dec} is a $L_{\mathrm{m}}(\tilde{G}_i)$-observer, there is a $u_i \in (\tilde{\Sigma}_i - \Sigma_i)^*$ s.t. $su_i \in L_{\mathrm{m}}(\tilde{G}_i)$. Because of Lemma 5.3, $su_i \in L(\tilde{S}_i/\tilde{G}_i)$. Together, $su_i \in L(\tilde{S}_i/\tilde{G}_i) \cap L_{\mathrm{m}}(\tilde{G}_i) = L_{\mathrm{m}}(\tilde{S}_i/\tilde{G}_i)$.
2. $t = \sigma_1 \cdots \sigma_m$. As p_i^{dec} is a $L_{\mathrm{m}}(\tilde{G}_i)$-observer, there is a $u_i = v_0\sigma_1 v_1 \cdots \sigma_m v_m \in \tilde{\Sigma}_i^*$ s.t. $su_i \in L_{\mathrm{m}}(\tilde{G}_i)$ and $p_i^{\mathrm{dec}}(u_i) = t$, i.e. $v_j \in (\tilde{\Sigma}_i - \Sigma_i)^*$ for all $j = 0, \ldots, m$. Successive application of Lemma 5.3 implies $su_i \in L(\tilde{S}_i/\tilde{G}_i)$. Thus, $su_i \in L(\tilde{S}_i/\tilde{G}_i) \cap L_{\mathrm{m}}(\tilde{G}_i) = L_{\mathrm{m}}(\tilde{S}_i/\tilde{G}_i)$. ■

Proof of Theorem 5.4:
Consistency follows from the proof of Theorem 5.3.

Now assume that $s \in L(\tilde{S}/\tilde{G})$. Then $s_i := \tilde{p}_i(s) \in L(\tilde{S}_i/\tilde{G}_i)$ and $p_i^{\mathrm{dec}}(s_i) \in L(S_i/G_i)$. As S_i is a nonblocking supervisor, there is a string $t \in \Sigma_i^*$ s.t. $p_i^{\mathrm{dec}}(s_i)t \in L_{\mathrm{m}}(S_i/G_i)$ and s.t. all its predecessors are not marked, i.e. $\forall t' < t$ we have that $p_i^{\mathrm{dec}}(s_i)t' \notin L_{\mathrm{m}}(S_i/G_i)$. Then it holds that $t \in (\Sigma_i - \Sigma_{i,\mathrm{s}})^*$ (otherwise there would be a marked predecessor string as $L_{\mathrm{m}}(G_i)$ marks $\Sigma_{i,\mathrm{s}}$). Because of Lemma 5.4, there is a $u_i \in \tilde{\Sigma}_i^*$ s.t. $su_i \in L_{\mathrm{m}}(\tilde{S}_i/\tilde{G}_i)$ and $p_i^{\mathrm{dec}}(su_i) = p_i^{\mathrm{dec}}(s)t$. Furthermore, as $p_i^{\mathrm{dec}}(u_i) = t \subseteq (\Sigma_i - \Sigma_{i,\mathrm{s}})^*$, it turns out that $u_i \in (p_i^{\mathrm{dec}})^{-1}(t) \subseteq (\tilde{\Sigma}_i - \Sigma_{i,\mathrm{s}})^*$. As i was arbitrary, such u_i exists for all $i = 1, \ldots, n$. Hence, for example the string $su_1 \cdots u_n \in \underline{\|_i \tilde{p}_i(s) u_i} \subseteq \|_i L_{\mathrm{m}}(\tilde{S}_i/\tilde{G}_i) = L_{\mathrm{m}}(\tilde{S}/\tilde{G})$ and consequently $s \in L_{\mathrm{m}}(\tilde{S}/\tilde{G})$. ■

The reduced structural decentralized control architecture is depicted in Figure 5.

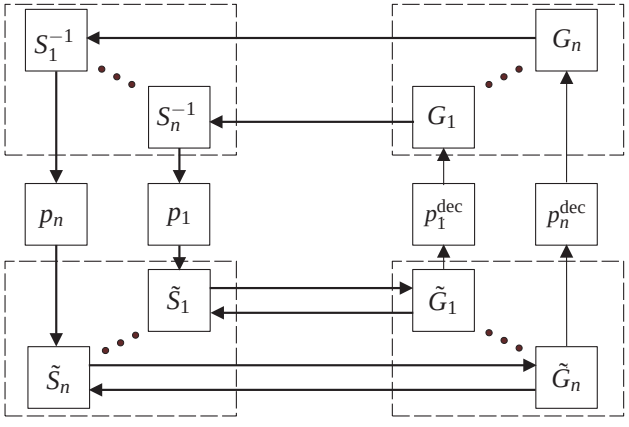

Fig. 5. Reduced decentralized architecture

VI. Conclusions

We have developed two methods exploiting the structure of concurrent systems for the supervisor synthesis without composition of the overall plant. In our approach, the computational complexity is further reduced by using reduced system models for supervisor computation. Our modular approach can be applied to prefix-closed non-separable specifications and results in modular supervisors in a conjunctive architecture. Additionaly, we elaborated a decentralized approach which is feasible for specifications that are separable but not necessarily prefix-closed. We provide two different sets of conditions which guarantee nonblocking control of the original system. It has to be noted that although maximally permissive supervisors could be computed for the reduced system models, the supervisors for the original system need not be maximally permissive. In further work, we want to investigate conditions which also guarantee maximally permissive supervisors for the original system.

References

[1] K. Akesson, H. Flordal, and M. Fabian. Exploiting modularity for synthesis and verification of supervisors. In *Proc. of the IFAC*, barcelona, Spain, July 2002.

[2] C.G Cassandras and S. Lafortune. Introduction to discrete event systems. *Kluwer*, 1999.

[3] A.E.C. da Cunha, J.E.R. Cury, and B.H. Krogh. An assume guarantee reasoning for hierarchical coordination of discrete event systems. *Workshop on Discrete Event Systems*, 2002.

[4] B. Gaudin and H. Marchand. Modular supervisory control of a class of concurrent discrete event systems. *Workshop on Discrete Event Systems*, 2004.

[5] J. Komenda, J. van Schuppen, B. Gaudin and H. Marchand. Modular supervisory control with general indecomposable specification languages. *Conference on Decision on Control*, 2005.

[6] S. Jiang and R. Kumar. Decentralized control of discrete-event systems with specializations to local control and concurrent systems. *IEEE Transactions on Systems, Man and Cybernetics*, 30(5):653–660, 2002.

[7] R.J. Leduc. Hierarchical interface based supervisory control. *PhD thesis, Department of Electrical and Computer Engineering, University of Toronto*, 2002.

[8] S-H. Lee and K.C. Wong. Structural decentralised control of concurrent DES. *European Journal of Control*, 35:1125–1134, October 2002.

[9] C. Ma. Nonblocking supervisory control of state tree structures. *Ph.D. Dissertation, Department of Electrical and Computer Engineering, University of Toronto*, 2004.

[10] M.H.de Querioz and J.E.R. Cury. Modular supervisory control of large scale discrete event systems. *Workshop on Discrete Event Systems*, 2000.

[11] K. Rohloff and S. Lafortune. The control and verification of similar agents operating in a broadcast network environment. In *42nd IEEE Conference on Decision and Control*, Hawaii, USA, 2003.

[12] K. Schmidt. Hierarchical control of decentralized discrete event systems: Theory and application. *Phd-thesis, Lehrstuhl für Regelungstechnik, Universität Erlangen-Nürnberg*, 2005.

[13] K. Schmidt, T. Moor, and S. Perk. A hierarchical architecture for nonblocking control of discrete event systems. *Mediterranean Conference on Control and Automation*, 2005.

[14] K. Schmidt, J. Reger, and T. Moor. Hierarchical control of structural decentralized DES. *Workshop on Discrete Event Systems*, 2004.

[15] Y. Willner and M. Heymann. Supervisory control of concurrent discrete-event systems. *International Journal of Control*, 54(5):1143–1169, 1991.

[16] K. Wong and W.M. Wonham. On the computation of observers in discrete-event systems. *Discrete Event Dynamic Systems*, 14(1):55–107, 2004.

[17] K.C. Wong and W.M. Wonham. Hierarchical control of discrete-event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 1996.

[18] W.M Wonham. Notes on control of discrete event systems. *Department of Electrical Engineering, University of Toronto*, 2004.