A Hierarchical Architecture for Nonblocking Control of Decentralized Discrete Event Systems

K. Schmidt, T. Moor and S. Perk

***Abstract*— This contribution investigates the hierarchical control of decentralized DES which are synchronized by shared events. A multi-level hierarchical control architecture providing hierarchical consistency is introduced. Moreover, it allows for composition of decentralized subsystems on the high-level of the hierarchy, and hence reduces the computational complexity of supervisory control synthesis for language inclusion specifications. In this context, a crucial issue is the nonblocking operation of the overall system. In our main theorem, marked state acceptance and marked state controllability are identified as sufficient conditions for this desirable property.**

I. Introduction

Recent approaches for reducing the computational effort of supervisor synthesis algorithms assume or impose a particular control architecture, such that product compositions of individual subsystems can be either avoided or at least postponed to a more favorable stage in the design process [5], [6], [8], [15]. Our contribution uses a hierarchical control architecture [3], [4], [7], [9], [10], [13], [16].

This paper extends previous results [11], [12], using the natural language projection as a high-level abstraction. We recall from [12] that, if a particular supervisor implementation is chosen, this abstraction complies with hierarchical consistency (see also [16]). It is also demonstrated that the hierarchical architecture can be extended to decentralized systems. Moreover, we show in [11] that a nonblocking closed loop is implied by the following conditions: local nonblocking, marked state consistency and circular closed loop behavior. The first two conditions are *structural* in that they refer to the open loop only.

In contrast to previous work, this contribution introduces marked state controllability as an additional structural property of hierarchical and decentralized systems and proves nonblocking low-level control by imposing structural conditions only.

The outline of the paper is as follows. Basic notations and definitions of supervisory control theory are recalled in Section II. Section III introduces the notion of marked state acceptance, marked state controllability and local nonblocking combined with hierarchical control and proves nonblocking control for the overall closed loop. In Section IV, the architecture is extended to form a decentralized and hierarchical control architecture.

Klaus Schmidt, Thomas Moor and Sebastian Perk are with the Lehrstuhl für Regelungstechnik, Universität Erlangen-Nürnberg, Germany {klaus.schmidt,thomas.moor}@rt.eei.uni-erlangen.de

II. Preliminaries

We recall basics from supervisory control theory. [14], [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 $\epsilon \in \Sigma^*$, i.e. $s\epsilon = \epsilon s = s$ 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 $H = \overline{H}$.

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: (1) let $p_i(\epsilon) := \epsilon$; (2) 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 $\delta(x, \epsilon) := x$ 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. For any string $s \in L(G)$, $\Sigma(s) := \{\sigma | s\sigma \in L(G)$ is the set of eligible events after s. A formal definition of the synchronous composition of two automata G_1 and G_2 can be taken from e.g. [2]. Note that $L(G_1 || G_2) = L(G_1) || L(G_2)$.

In a supervisory control context, we write $\Sigma = \Sigma_c \cup \Sigma_u$, $\Sigma_c \cap \Sigma_u = \emptyset$, to distinguish *controllable* (Σ_c) and *uncontrollable* (Σ_u) events. A *control pattern* is a set γ, $\Sigma_u \subseteq \gamma \subseteq \Sigma$, 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 (1) $\epsilon \in L(S/G)$ and (2) $s\sigma \in L(S/G)$ 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 $L_m(S/G) :=$

$L(S/G) \cap L_m(G)$. The closed-loop system is *nonblocking* if $\overline{L_m(S/G)} = L(S/G)$, i.e. if each string in $L(S/G)$ is the prefix of a marked string in $L_m(S/G)$.

A language H is said to be controllable w.r.t. $L(G)$ 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. $L(G)$, which is formally defined as $\kappa_{L(G)}(E) := \cup\{K \in \mathcal{C}(L(G))|\ K \subseteq E\}$. A supervisor S that leads to a closed-loop behavior $\kappa_{L(G)}(E)$ is said to be *maximally permissive*. A maximally permissive supervisor can be realized on the basis of a generator of $\kappa_{L(G)}(E)$. The latter can be computed from G and a generator of E. The computational complexity is of order $O(N^2M^2)$, where N and M are the number of states in G and the generator of E, respectively.

A language E is L_m*-closed* if $\overline{E} \cap L_m = E$ and the set of $L_m(G)$-closed languages is denoted $\mathcal{F}_{L_m(G)}$. For specifications $E \in \mathcal{F}_{L_m(G)}$, the plant $L(G)$ is nonblocking under maximally permissive supervision.

III. Nonblocking Hierarchical Control

Monolithic supervisory control faces the problem of a very high computational effort for large systems. One method for reducing this effort is hierarchical control, where an abstracted (high-level) plant model is used for supervisor synthesis on a higher level. Then, the high-level control has to be implemented in the lower level. In this work, the event-based control scheme proposed in [16] is used (compare Figure 2). The detailed plant model G and the supervisor S^{lo} form a low-level closed-loop system, indicated by Con^{lo} (control action) and Inf^{lo} (feedback information). Similarly, the high-level closed loop consists of an abstracted plant model G^{hi} and a high-level supervisor S^{hi}. The two levels are interconnected via Com^{hilo}, imposing high-level control on S^{lo} and Inf^{lohi} which drives the abstract plant G^{hi} in accordance to the detailed model.

A. Hierarchical Control Problem

The natural projection is used as a method for hierarchical abstraction. For other methods consult [3], [4], [7], [16].

Definition 3.1 (Projected System): Let $G = (X, \Sigma, \delta, x_0, X_m)$ be a nonblocking DES and $\Sigma^{hi} \subseteq \Sigma$ an abstraction alphabet. Also let $p^{hi} : \Sigma^* \rightarrow (\Sigma^{hi})^*$ be the natural projection. The high-level language is defined by $L^{hi} := p^{hi}(L(G))$. The projected marked language is $L_m^{hi} := p^{hi}(L_m(G))$ and G^{hi} is the canonical recognizer s.t. $L_m(G^{hi}) = L_m^{hi}$ and $L(G^{hi}) = L^{hi}$. With $\Sigma = \Sigma_u \dot{\cup} \Sigma_c$, high-level uncontrollable and controllable events are $\Sigma_u^{hi} := p^{hi}(\Sigma_u)$ and $\Sigma_c^{hi} := p^{hi}(\Sigma_c)$, respectively. The tuple (G, θ, G^{hi}) is called a projected system.

The interconnection of low- and high-level supervisors with the plant is defined as follows.

Definition 3.2 (Hierarchical Control System): Referring to Definition 3.1, a *hierarchical control system (HCS)* $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$ consists of a projected system (G, p^{hi}, G^{hi}) which is equipped with a *high-level supervisor* S^{hi} and a *low-level supervisor* S^{lo}, where S^{hi} and S^{lo} fulfill the following conditions:

- S^{hi}: $L^{hi} \rightarrow \Gamma^{hi}$ with the high-level control patterns $\Gamma^{hi} := \{\gamma|\Sigma_u^{hi} \subseteq \gamma \subseteq \Sigma^{hi}\}$.
- $S^{lo} : L(G) \rightarrow \Gamma$. S^{lo} is called valid if $p^{hi}(L(S^{lo}/G)) \subseteq L(S^{hi}/G^{hi})$.

This contribution investigates the implementation of a non-blocking low-level supervisor in case the abstraction alphabet and the high-level supervisor are already known. Thus we focus on translating a non-blocking high-level supervisor S^{hi} to a valid nonblocking controller S^{lo} in the low level. This is fomalized in Definition 3.3.

Definition 3.3 (Hierarchical Control Problem): Given a HCS $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$, compute a valid low-level supervisor S^{lo} as in Definition 3.2 such that the low-level controlled language of the HCS, $L(S^{lo}/G)$, is nonblocking.

B. Hierarchical Consistency

Hierarchical consistency has been used as a powerful tool for showing nonblocking behavior of hierarchical control architectures [3], [4], [7], [16]. This property ensures that the desired high-level behavior can be implemented in the low level. Thus it imposes certain requirements on the translation of the high-level supervisor to the low-level controller. For our particular abstraction, we can guarantee hierarchical consistency by a specific supervisor implementation which is based on the following definitions.

The set of entry strings contains all low-level strings which are just projected to a given high-level string.

Definition 3.4 (Entry Strings): Let (G, p^{hi}, G^{hi}) be a projected system. The set of entry strings of $s^{hi} \in L^{hi}$ is

$$\begin{aligned} L_{en,s^{hi}} \quad := \quad & \{s \in L(G)|p^{hi}(s) = s^{hi} \wedge \\ & \nexists s' < s \text{ s.t. } p^{hi}(s) = s^{hi}\} \subseteq \Sigma^* \end{aligned}$$

The local (marked) language consists of (marked) strings which are reachable by local strings $u \in (\Sigma - \Sigma^{hi})^*$ from a given string s.

Definition 3.5 (Local Languages): Let (G, p^{hi}, G^{hi}) be a projected system and let $s \in L(G)$ for $s^{hi} := p^{hi}(s) \in L^{hi}$. The local language $L_{s,s^{hi}}$ is

$$L_{s,s^{hi}} := \{u\sigma|su\sigma \in L(G) \wedge p^{hi}(su) = s^{hi} \wedge \sigma \in \Sigma\}$$

and the locally marked language $L_{s,s^{hi},\gamma^{hi}}$ for the control pattern $\gamma^{hi} \in \Gamma^{hi}$ is[1]

$$L_{s,s^{hi},\gamma^{hi}} := \{u\sigma \in L_{s,s^{hi}} | su\sigma \in L_m(G)\} \cup \{u\sigma | u \in L_{s,s^{hi}} \wedge \sigma \in \gamma^{hi} \wedge su\sigma \in L(G)\}.$$

Using the above definitions, a consistent implementation of a high-level supervisor can be introduced.

Definition 3.6 (Consistent Implementation): Given a projected system (G, p^{hi}, G^{hi}) and a supervisor S^{hi}, we define the consistent implementation S^{lo}. For $s \in L(G)$, let $s^{hi} := p^{hi}(s)$ and $s_{en} \in L_{en,s^{hi}}$, $u \in (\Sigma - \Sigma^{hi})^*$ s.t. $s = s_{en}u$. Then

$$S^{lo}(s) := \begin{cases} \text{if } \Sigma^{hi}(s^{hi}) \cap S^{hi}(s^{hi}) \neq \emptyset \text{ then} \\ \quad S^{hi}(s^{hi}) \cup (\Sigma - \Sigma^{hi}) \\ \text{else} \\ \quad \{\sigma | u\sigma \in \kappa_{L_{s_{en},s^{hi}}}(L_{s_{en},s^{hi},S^{hi}(s^{hi})})\} \cup \Sigma_u \end{cases}$$

For any low-level string, the consistent supervisor allows all low-level events as long as the high-level supervisor allows some successor event after the corresponding high-level string. If this is not the case, there might be strings in the low-level behavior which cannot be extended to a marked state. Because of this reason, the low-level controller implements the maximally permissive and nonblocking behavior for the relevant local automaton.

By the following lemma, it is shown that the consistent implementation is indeed an admissible low-level supervisor.

Lemma 3.1 (Admissible Supervisor Implementation): Let (G, p^{hi}, G^{hi}) be a projected system and S^{hi} an admissible supervisor with a consistent implementation S^{lo}. Then S^{lo} is admissible, i.e. $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$ is a HCS.

As mentioned above, our construction leads to a hierarchically consistent closed-loop behavior, i.e. the projected behavior of the low-level supervised plant equals the controlled high-level behavior (see also [3], [4], [7], [16]).

Theorem 3.1 (Hierarchical Consistency): Let $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$ be a HCS. If S^{lo} is the consistent implementation of S^{hi}, then the HCS is *hierarchically consistent*, i.e. $p^{hi}(L(S^{lo}/G)) = L(S^{hi}/G^{hi})$.

The proof of this Theorem is obmitted due to lack of space. Note that up to now, no structural properties of the control system have been used.

C. Nonblocking Control

Nonblocking control of the low-level is an essential prerequisite of hierarchical control. In contrast to the above construction, nonblocking behavior does not come for free. We identify structural properties which imply nonblocking behavior of the hierarchical control system.

[1]Note that $L_{s,s^{hi},\Sigma^{hi}(s^{hi})} \subseteq L_{s,s^{hi}}$.

The set of exit strings contains all low-level strings which have a high-level successor event.

Definition 3.7 (Exit Strings): Let (G, p^{hi}, G^{hi}) be a projected system and assume $s^{hi} \in L^{hi}$. The set of exit strings of s^{hi} is

$$L_{ex,s^{hi}} := \{s \in L(G) | p^{hi}(s) = s^{hi} \wedge \exists \sigma^{hi} \in \Sigma^{hi} \text{ s.t. } s\sigma^{hi} \in L(G)\} \subseteq \Sigma^*.$$

Marked state acceptance guarantees that if a marked high-level string is traversed, then a marked string has also been passed in the low level.

Definition 3.8 (Marked State Acceptance): Let (G, p^{hi}, G^{hi}) be a projected system. The string $s_m^{hi} \in L_m^{hi}$ is marked state accepting[2] if for all $s_{ex} \in L_{s_m^{hi},ex}$

$$\exists s' \leq s_{ex} \text{ with } p^{hi}(s') = s_m^{hi} \text{ and } s' \in L_m(G).$$

Let $L_m^p \subseteq L_m^{hi}$. (G, p^{hi}, G^{hi}) is marked state accepting w.r.t L_m^p if s_m^{hi} is marked state accepting for all $s_m^{hi} \in L_m^p$.

For locally nonblocking DES it holds that after any low-level string, there is a local path to all successor events of the corresponding high-level string.[3] This property is closely related to the observer property in [10] and [14].

Definition 3.9 (Locally Nonblocking DES): Let (G, p^{hi}, G^{hi}) be a projected system and let $L_m^p \subseteq L_m^{hi}$. $s^{hi} \in \overline{L_m^p}$ is locally nonblocking w.r.t L_m^p if for all $s \in L(G)$ with $p^{hi}(s) = s^{hi}$ and $\forall \sigma \in \Sigma^{hi}(s^{hi})$ with $p^{hi}(s)\sigma \in \overline{L_m^p}$, $\exists u_\sigma \in (\Sigma - \Sigma^{hi})^*$ s.t. $su_\sigma\sigma \in L(G)$. (G, p^{hi}, G^{hi}) is locally nonblocking w.r.t. L_m^p if s^{hi} is locally nonblocking w.r.t. L_m^p $\forall s^{hi} \in \overline{L_m^p}$.

Marked state controllability guarantees nonblocking low-level control for the case that no high-level event is possible after a marked high-level string.

Definition 3.10 (Marked State Controllability): Let (G, p^{hi}, G^{hi}) be a projected system. Let $L_m^p \subseteq L_m^{hi}$ and $s^{hi} \in L_m^p$ with $\gamma^{hi} \in \Gamma^{hi}$ s.t. $\gamma^{hi} \cap \Sigma^{hi}(s^{hi}) = \emptyset$. s^{hi} is marked state controllable if for all $s_{en} \in L_{en,s^{hi}}$ $\kappa_{L_{s_{en},s^{hi}}}(L_{s_{en},s^{hi},\gamma^{hi}}) \neq \emptyset$. (G, p^{hi}, G^{hi}) is marked state controllable w.r.t. L_m^p if s^{hi} is marked state controllable $\forall s^{hi} \in L_m^p$.

The above definitions are illustrated by short example. Consider the low-level automaton and the corresponding high-level automaton in Figure 1.[4]

Example 3.1: The entry strings of the high-level string α are $L_{en,\alpha} = \{(fg)^*\alpha\}$, the exit strings of α are $L_{\alpha,ex} = \{(fg)^*\alpha(ab+cd+ce)\}$, the local language $L_{\alpha,\alpha}$ is $L_{\alpha,\alpha} =$

[2]Note that $s^{hi} \in L^{hi} - L_m^{hi} \Rightarrow (p^{hi})^{-1}(s^{hi}) \cap L_m(G) = \emptyset$.

[3]There are weaker conditions for nonblocking hierarchical control. This restrictive condition is used as it directly extends to decentralized systems.

[4]Controllable events are marked with a tick and dotted lines are used for high-level events.

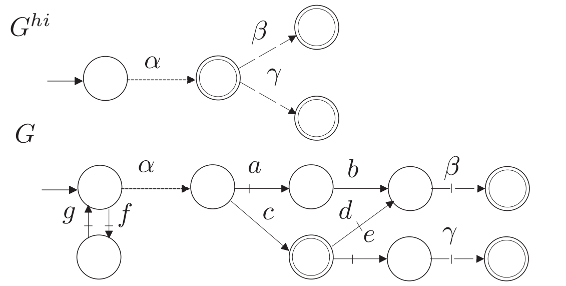

Fig. 1. low-level automaton G and projected automaton G^{hi}

$\{c, cd, cd\beta, ce, ce\gamma, a, ab, ab\beta\}$ and the locally marked language $L_{\alpha,\alpha,\{\beta,\gamma,\alpha\}}$ is $L_{\alpha,\alpha,\{\beta,\gamma,\alpha\}} = \{c, cd\beta, ce\gamma, ab\beta\}$.

For investigating marked state acceptance, it is readily observed that the exit string $s = \alpha ab$ does not have a marked predecessor string $s' < s$ with $p^{hi}(s') = \alpha$. Thus the high-level string α is not marked state accepting. Local nonblocking w.r.t. $L(G^{hi})$ is also not true, as it is not possible to find a local string leading to the occurrence of the high-level event γ after the string αab. For checking marked state controllability, we assume that the high-level supervisor disables γ and β after the high-level string α. A low-level supervisor disabling e and d after $(fg)^*\alpha c$ and a after $(fg)^*\alpha$ leads to nonblocking behavior of the controlled low-level plant and at the same time the controlled local language after the entry string α equals $\kappa_{L_{\alpha,\alpha}}(L_{\alpha,\alpha,\{\alpha\}})$.

With the above properties as conditions, the main theorem of this section establishes nonblocking control for hierarchical control systems.

Theorem 3.2 (Nonblocking Hierarchical Control): Let $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$ be a hierarchical control system with a marked state accepting, marked state controllable and locally nonblocking projected system (G, p^{hi}, G^{hi}) w.r.t. L_m^{hi}. Also let S^{hi} be a high-level supervisor with a consistent implementation S^{lo}. Then S^{lo} solves the hierarchical control problem in Definition 3.3 and the HCS is hierarchically consistent.

The proof is based on the subsequent lemmas.

Lemma 3.2: Let (G, p^{hi}, G^{hi}) be a locally nonblocking projected system w.r.t. L_m^p and let S^{hi} be a nonblocking high-level supervisor s.t. $L_m(S^{hi}/G^{hi}) \subseteq L_m^p$. Also let S^{lo} be a consistent implementation of S^{hi}. Then $(S^{lo}/G, p^{hi}, S^{hi}/G^{hi})$ is a locally nonblocking projected system w.r.t. $L_m(S^{hi}/G^{hi})$.

Lemma 3.3: Let $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$ be a hierarchical control system with a locally nonblocking projected system (G, p^{hi}, G^{hi}) w.r.t. L_m^p s.t. $L_m(S^{hi}/G^{hi}) \subseteq L_m^p \subseteq L_m^{hi}$ and let S^{lo} be a consistent implementation. Assume $s \in L(S^{lo}/G)$ for some $s^{hi} \in L(S^{hi}/G^{hi})$. If $t \in (\Sigma^{hi})^*$ s.t. $s^{hi}t \in L(S^{hi}/G^{hi})$ then $\exists u \in \Sigma^*$ with $p^{hi}(u) = t$ and $su \in L(S^{lo}/G) \cap L_{en,s^{hi}t}$.

Lemma 3.4: Let $(G, p^{hi}, G^{hi}, S^{hi}, S^{lo})$ be a hierarchical control system with a marked state accepting projected system (G, p^{hi}, G^{hi}) w.r.t. L_m^p s.t. $L_m(S^{hi}/G^{hi}) \subseteq L_m^p \subseteq L_m^{hi}$ and let S^{lo} be a consistent implementation. Also let $s_{en} \in L_{en,s^{hi}} \cap L(S^{lo}/G)$ for $s^{hi} \in L_m(S^{hi}/G^{hi})$. Then $\exists u \in (\Sigma - \Sigma^{hi})^*$ s.t. $s_{en}u \in L_m(S^{lo}/G)$.

Finally Theorem 3.2 can be proven.

Proof: Lemma 3.1 provides hierarchical consistency.

For proving nonblocking behavior, it has to be shown that $\forall s \in L(S^{lo}/G)$, $\exists u \in \Sigma^*$ s.t. $su \in L_m(S^{lo}/G)$. Because of hierarchical consistency, $s^{hi} := p^{hi}(s) \in L(S^{hi}/G^{hi})$. As S^{hi} is nonblocking, $\exists t \in (\Sigma^{hi})^*$ s.t. $s^{hi}t \in L_m(S^{hi}/G^{hi})$. There are two cases. First let $S^{hi}(s^{hi}) \cap \Sigma^{hi}(s^{hi}) = \emptyset$. Writing $s = s_{en}u$ with $s_{en} \in L_{en,s^{hi}}$ and $u \in (\Sigma - \Sigma^{hi})^*$ and noting that $s \in L(S^{lo}/G)$, we have that $u \in \kappa_{L_{s_{en},s^{hi}}}(L_{s_{en},s^{hi},S^{hi}(s^{hi})})$. Thus $\exists \bar{u} \in (\Sigma - \Sigma^{hi})^*$ s.t. $s\bar{u} \in L_m(G)$ and also $s\bar{u}' \in L(S^{lo}/G)$ for all $\bar{u}' \leq \bar{u}$ because of Definiton 3.6. Thus $su \in L_m(S^{lo}/G)$. Now let $S^{hi}(s^{hi}) \cap \Sigma^{hi}(s^{hi}) \neq \emptyset$. Then $\exists t \neq \epsilon$ s.t. $s^{hi}t \in L_m(S^{hi}/G^{hi})$. Because of Lemma 3.2 and Lemma 3.3, $\exists u' \in \Sigma^*$ s.t. $su' \in L(S^{lo}/G)$ and $p^{hi}(u') = t$. W.l.o.g., let $su' \in L_{en,s^{hi}t}$. Considering that $s^{hi}t \in L_m(S^{hi}/G^{hi})$ and $su' \in L_{en,s^{hi}t}$, $\exists u'' \in (\Sigma - \Sigma^{hi})^*$ s.t. $su'u'' \in L_m(S^{lo}/G)$ because of Lemma 3.4. In both cases, $s \in \overline{L_m(S^{lo}/G)}$. □

Thus the hierarchical control architecture guarantees hierarchical consistency if a consistent implementation of the low-level supervisor is chosen. If, in addition to that, the structural properties local nonblocking, marked state acceptance and marked state controllability are fulfilled, the hierarchical control system is also nonblocking.

IV. Decentralized Approach

One drawback of the purely hierarchical approach is the fact that the overall low-level model has to be computed before performing the abstraction, again leading to a high computational effort. Because of this, the decentralized nature of composed systems shall be exploited in the sequel. To this end, our method readily extends to decentralized systems. At first some useful notions are introduced.

A. Decentralized Control System

Definition 4.1 (Decentralized Control System): A *decentralized control system* $||_{i=1}^n G_i$ (DCS) consists of subsystems, modelled by finite state automata $G_i, i = 1, \ldots, n$ over the respective alphabets Σ_i. The overall system is defined as $G := ||_{i=1}^n G_i$ over the alphabet $\Sigma := \bigcup_{i=1}^n \Sigma_i$. The controllable and uncontrollable events are $\Sigma_{i,c} := \Sigma_i \cap \Sigma_c$ and $\Sigma_{i,u} := \Sigma_i \cap \Sigma_u$, respectively, where $\Sigma_c \dot{\cup} \Sigma_u = \Sigma$ and $\Sigma_c \cap \Sigma_u = \emptyset$. For brevity and convenience, let $L := L(G)$, $L_m := L_m(G)$, $L_i := L(G_i)$, and $L_{i,m} := L_m(G_i)$.

The projected decentralized control system corresponds to the projected control system in the monolithic approach.

Definition 4.2 (Projected Decentralized Control System): Let $\|_{i=1}^n G_i$ be a DCS, let Σ^{hi} s.t. $\bigcup_{i,j,i\neq j}(\Sigma_i \cap \Sigma_j) \subseteq \Sigma^{hi} \subseteq \Sigma$, $i = 1,\dots,n$[5] and let $p^{hi} : \Sigma^* \to (\Sigma^{hi})^*$ be the natural projection. A *projected decentralized control system* $(\|_{i=1}^n G_i, p^{hi}, \|_{i=1}^n G_i^{hi})$ (PDCS) is composed of finite state automata G_i^{hi}, $i = 1,\dots,n$, such that $L_i^{hi} := L(G_i^{hi}) = p^{hi}(L_i)$ and $L_{i,m}^{hi} := L_m(G_i^{hi}) = p^{hi}(L_{i,m})$. The language of the PDCS is denoted $L^{hi} := L(G^{hi})$ and its marked language is $L_m^{hi} := L_m(G^{hi})$ where the overall automaton is $G^{hi} := \|_{i=1}^n G_i^{hi}$.

As the decentralized subsystems are synchronized via shared events, the feasible shared behavior of each subsystem can be different from their independent behavior, i.e. it is possible that $p_i(L^{hi}) \subset L_i^{hi}$. The feasible projected marked sublanguage of a subsystem represents its reachable marked strings in the synchronized behavior.

Definition 4.3 (Feasible Projected Marked Sublanguage): Let $(\|_{i=1}^n G_i, p^{hi}, \|_{i=1}^n G_i^{hi})$ be a PDCS and let $p_i : \Sigma^* \to \Sigma_i^*$ be the natural projection. For $i = 1,\dots,n$, the feasible projected marked sublanguage (FPMS) $L_{i,m}^f$ of $L_{i,m}^{hi}$ is defined as $L_{i,m}^f := p_i(L_m^{hi})$.

B. Hierarchical and Decentralized Control

The hierarchical and decentralized control system applies the above hierarchical architecture to decentralized systems.

Definition 4.4: A Hierarchical and Decentralized Control System (HDCS) $(\|_{i=1}^n G_i, S_i, \|_{i=1}^n G_i^c, p^{hi}, \|_{i=1}^n G_i^{hi}, S^{hi}, S^{lo})$ consists of the following entities:

- A detailed plant model is a decentralized control system $\|_{i=1}^n G_i$ as in Definition 4.1.
- Nonblocking low-level controllers are denoted $S_i\colon L_i \to \Gamma_i$, where Γ_i are the respective control patterns. Low-level closed-loop languages are denoted $L_i^c := L(S_i/G_i)$, $L_{i,m}^c := L_i^c \cap L_{i,m}$, $L^c := \|_{i=1}^n L_i^c$, $L_m^c := \|_{i=1}^n L_{i,m}^c = L^c \cap L_m$. Also let G^c be a generator such that $L^c = L(G^c)$, $L_m^c = L_m(G^c)$.
- (G^c, p^{hi}, G^{hi}) is a projected system with the natural projection $p^{hi} : \Sigma^* \to (\Sigma^{hi})^*$, where $\bigcup_{i,j,i\neq j}^{n}(\Sigma_i \cap \Sigma_j) \subseteq \Sigma^{hi} \subseteq \Sigma$ and the high level marking is chosen as[6] $L_m^{hi} := p^{hi}(L_m^c)$. High-level controllable events are defined as $\Sigma_c^{hi} := \Sigma_c \cap \Sigma^{hi}$ and $\Sigma_u^{hi} := \Sigma_u \cap \Sigma^{hi}$.
- The high-level supervisor is denoted $S^{hi}\colon L^{hi} \to \Gamma^{hi}$ with the high-level closed-loop language $L(S^{hi}/G^{hi})$ and a valid low-level supervisor $S^{lo}\colon L^c \to \Gamma$ must fulfill $p^{hi}(L(S^{lo}/G^c)) \subseteq L(S^{hi}/G^{hi})$.
- a decentralized implementation of S^{lo} consists of supervisors S_i^{lo} s.t. $L(S^{hi}/G^{hi})||\big(\|_{i=1}^n L(S_i^{lo}/G_i^c)\big) = L(S^{lo}/G^c)$.

[5]This means that Σ^{hi} contains all shared events.
[6]By construction L_m^{hi} is regular.

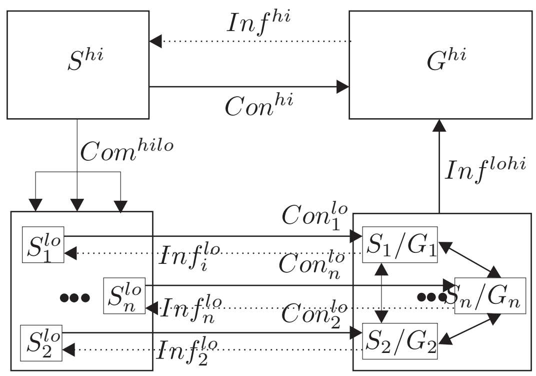

Fig. 2. Control Scheme for HDCS

The construction of the high-level plant is facilitated by composing the projected subsystems instead of composing the subsystems before projecting to the high level.

Lemma 4.1 (High Level Plant): Let $(\|_{i=1}^n G_i, p^{hi}, \|_{i=1}^n G_i^{hi})$ be a projected decentralized control system. Then the high level closed and marked languages are $L^{hi} = p^{hi}(\|_{i=1}^n L_i^c) = \|_{i=1}^n L_i^{hi}$ and $L_m^{hi} = p^{hi}(\|_{i=1}^n L_{i,m}^c) = \|_{i=1}^n L_{i,m}^{hi}$, respectively.

The mutual controllability condition [8] guarantees that the projections of a high-level supervisor to the subsystems is controllable with respect to the respective subsystem.

Lemma 4.2 (Mutual Controllability): Let $(\|_{i=1}^n G_i, S_i, \|_{i=1}^n G_i^c, p^{hi}, \|_{i=1}^n G_i^{hi}, S^{hi}, S^{lo})$ be a HDCS. If L_i^{hi} and L_j^{hi} are mutually controllable for $i,j = 1,\dots,n$, i.e.

$$L_j^{hi}(\Sigma_u^{hi} \cap \Sigma_i \cap \Sigma_j) \cap p_j\big((p_i^{hi})^{-1}(L_i^{hi})\big) \subseteq L_j^{hi},$$

then $\exists S_i^{hi} : (\Sigma_i^{hi})^* \to \Gamma_i^{hi}$ s.t. $L(S_i^{hi}/G_i^{hi}) = p_i(L(S^{hi}/G^{hi}))$ for all $i = 1,\dots,n$.

With the mutual controllability condition in addition to the conditions needed in Theorem 3.2, the main theorem of this paper states nonblocking decentralized control for hierarchical and decentralized control systems.

Theorem 4.1 (Main Result): Let $(\|_{i=1}^n G_i, S_i, \|_{i=1}^n G_i^c, p^{hi}, \|_{i=1}^n G_i^{hi}, S^{hi}, S^{lo})$ be a HDCS and for $i = 1,\dots,n$ let $L_{i,m}^f$ be the feasible projected marked sublanguage. Assume that the high-level languages L_i^{hi} are mutually controllable and all projected systems $(G_i^c, p^{hi}, G_i^{hi})$, $i = 1,\dots,n$ are marked state accepting, marked state controllable and locally nonblocking w.r.t. $L_{i,m}^f$. Let S_i^{hi} be supervisors s.t. $L(S_i^{hi}/G_i^{hi}) = p_i(L(S^{hi}/G^{hi}))$ and define S_i^{lo} as consistent implementations of S_i^{hi}. Then, the low-level supervisor S^{lo} s.t. $L(S^{lo}/G^c) = L(S^{hi}/G^{hi})||\big(||_{i=1}^n L(S_i^{lo}/G_i^c)\big)$ is nonblocking and the HDCS is hierarchically consistent.

Proof: For proving hierarchical consistency, we first note that the language $p_i(L(S^{hi}/G^{hi}))$ is controllable w.r.t. L_i^{hi} for all $i = 1, \ldots, n$ because of Lemma 4.2 and conclude that the hierarchical control systems $(G_i^c, p^{hi}, G_i^{hi}, S_i^{lo}, S_i^{hi})$ are hierarchically consistent as a consistent implementation S_i^{lo} of S_i^{hi} is chosen. Also it holds that $p^{hi}(||_{i=1}^n L(S_i^{lo}/G_i^c)) = ||_{i=1}^n p^{hi}(L(S_i^{lo}/G^c)) = ||_{i=1}^n L(S_i^{hi}/G_i^{hi})$ because of Lemma 4.1 as $(||_{i=1}^n S_i^{lo}/G_i^c, p^{hi}, S_i^{hi}/G_i^{hi})$ is a projected decentralized control system. Considering the definition of S_i^{hi}, we arrive at $p^{hi}(||_{i=1}^n L(S_i^{lo}/G_i^c)) = ||_{i=1}^n p_i(L(S^{hi}/G^{hi}))$. As $L(S^{hi}/G^{hi}) \subseteq ||_{i=1}^n p_i(L(S^{hi}/G^{hi}))$, it holds that $L(S^{hi}/G^{hi}))||p^{hi}(||_{i=1}^n L(S_i^{lo}/G_i^c)) = L(S^{hi}/G^{hi}))||\big(||_{i=1}^n p_i(L(S^{hi}/G^{hi}))\big) = L(S^{hi}/G^{hi})$ which proves hierarchical consistency of the hierarchical and decentralized control system with the decentralized supervisor implementation.

For proving nonblocking behavior, we first note that it holds that $L(S^{lo}/G^c) \neq \emptyset$ as $L(S^{hi}/G^{hi}) \neq \emptyset$ and $p^{hi}(L(S^{lo}/G^c)) = L(S^{hi}/G^{hi})$. Now assume that $s \in L(S^{lo}/G^c)$ and $s^{hi} = p^{hi}(s) \in L(S^{hi}/G^{hi})$. It has to be shown that $s \in \overline{L_m(S^{lo}/G^c)}$. Because of the definition of S^{lo}, $s_i := p_i(s) \in L(S_i^{lo}/G_i^c)$ and $s_i^{hi} := p_i(s^{hi}) \in L(S_i^{hi}/G_i^{hi})$. Let $\mathcal{I}_0 := \{i | \nexists u_i \in (\Sigma_i - \Sigma^{hi})^* \text{ s.t. } s_i u_i \in L_m(S_i^{lo}/G_i^c)\}$. The following algorithm is performed to find an appropriate string leading to a marked state in the high-level.

1. $k = 1$, $\mathcal{I} = \mathcal{I}_0$.
2. choose $i_k \in \mathcal{I}$.
3. find $t_k \in (\Sigma^{hi})^*$ s.t. $s^{hi} t_1 \cdots t_k \in L_m(S^{hi}/G^{hi})$ and $p_{i_k}(t_k) \neq \epsilon$.
4. remove all j with $p_j(t_{i_k}) \neq \epsilon$ from $\mathcal{I}$.
5. **if** $\mathcal{I} = \emptyset$: set $k^* = k$ and **terminate**
   else $k := k + 1$ and **go to** 2.

First note that the string t_k in 3. always exists. In case that $\Sigma_i^{hi}(s_i^{hi}) \cap S_i^{hi}(s_i^{hi}) = \emptyset$ for some i it holds that $i \notin \mathcal{I}$, as there must be $u_i \in (\Sigma_i - \Sigma^{hi})^*$ s.t. $s_i u_i \in L_m(S_i^{lo}/G_i^c)$ because of the consistent implementation. Thus, for all $i \in \mathcal{I}$, it holds that $\Sigma_i^{hi}(s_i^{hi}) \cap S_i^{hi}(s_i^{hi}) \neq \emptyset$. Thus $\exists t_i \neq \epsilon$ s.t. $s_i^{hi} t_i \in L_m(S_i^{hi}/G_i^{hi})$. Also $\exists t \in (\Sigma^{hi})^*$ with $p_i(t) = t_i \neq \epsilon$ and $s^{hi} t \in L_m(S^{hi}/G^{hi})$ as $L_m(S_i^{hi}/G_i^{hi}) = p_i(L_m(S^{hi}/G^{hi}))$. Also note that the algorithm terminates as $\mathcal{I}$ is a finite index set which is reduced in every step. Now define $t := t_1 \cdots t_{k^*}$. $s^{hi} t \in L_m(S^{hi}/G^{hi})$ and $\forall i, s_i^{hi} p_i(t) \in L_m(S_i^{hi}/G_i^{hi})$. Because of Lemma 3.3, $\forall i \in \mathcal{I}_0$, $\exists u_i \in \Sigma_i^*$ s.t. $s_i u_i \in L(S_i^{lo}/G_i^c) \cap L_{en, s_i^{hi} p_i(t)}$. Then, because of Lemma 3.4, $\exists \bar{u}_i \in (\Sigma - \Sigma^{hi})^*$ s.t. $s_i u_i \bar{u}_i \in L_m(S_i^{lo}/G_i^c)$. For $i \notin \mathcal{I}_0$, we define $u_i := \epsilon$ and we note that $\exists \bar{u}_i \in (\Sigma - \Sigma^{hi})^*$ s.t. $s_i u_i \bar{u}_i \in L_m(S_i^{lo}/G_i^c)$ by definition of $\mathcal{I}_0$. Then $\forall u \in ||_{i=1}^n s_i u_i \bar{u}_i$, it holds that $su \in ||_{i=1}^n L_m(S_i^{lo}/G_i^c)$ and $p^{hi}(su) = s^{hi} t \in L_m(S^{hi}/G^{hi})$ and thus $su \in L_m(S^{hi}/G^{hi})||\big(||_{i=1}^n L_m(S_i^{lo}/G_i^c)\big) = L_m(S^{lo}/G^c)$. Hence $s \in \overline{L_m(S^{lo}/G^c)}$. $\square$

Hence, the proposed hierarchichal and decentralized control architecture readily extends the hierarchical architecture presented in Section III and nonblocking and hierarchically consistent behavior of the control system is guaranteed.

V. Conclusions

In this contribution, a hierarchical control architecture was introduced and applied to decentralized discrete event systems. It has been shown that the architecture automatically provides hierarchical consistency and in combination with a particular supervisor implementation also guarantees nonblocking behavior of the control system. In contrast to [11], which needs conditions on the specification languages, only structural system properties are used in this paper. The method has been applied to an automated manufacturing system example and the supervisor synthesis yields nonblocking decentralized supervisors of manageable size [1].

References

[1] http://www.rt.e-technik.uni-erlangen.de/fgdes/.

[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. *WODES*, 2002.

[4] P. Hubbard and P.E. Caines. Dynamical consistency in hierarchical supervisory control. *IEEE TAC*, 2002.

[5] S. Jiang, V. Chandra, and R. Kumar. Decentralized control of discrete event systems with multiple local specializations. *Proc. ACC*, 2001.

[6] J. Komenda and J. H. van Schuppen. Decentralized control with coalgebra. *ECC*, 2003.

[7] R.J. Leduc, W.M. Wonham, and M. Lawford. Hierarchical interface-based supervisory control: Parallal case. *Allerton Conf. on Comm., Contr. and Comp.*, 2001.

[8] S-H. Lee and K.C. Wong. Stuctural decentralised control of concurrent DES. *EJC*, 2002.

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

[10] K.Q. Pu. Modeling and control of discrete event systems with hierarchical abstraction. *Master Thesis, Department of Electrical and Computer Engineering, University of Toronto*, 2000.

[11] K. Schmidt, S. Perk, and T. Moor. Nonblocking hierarchical control of decentralized systems. *Technical Report, Lehrstuhl für Regelungstechnik, Universität Erlangen Nürnberg, accepted for IFAC World Congress*, 2004.

[12] K. Schmidt, J. Reger, and T. Moor. Hierarchical control of structural decentralized DES. *WODES*, 2004.

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

[14] W.M Wonham. Supervisory control of discrete event systems. *Department of Electrical & Computer Engineering, University of Toronto*, 2004.

[15] T. Yoo and S. Lafortune. A generalized framework for decentralized supervisory control of discrete event systems. *WODES*, 2000.

[16] H. Zhong and W.M. Wonham. On the consistency of hierarchical supervision in discrete-event systems. *IEEE TAC*, 1990.