2015 IEEE 54th Annual Conference on Decision and Control (CDC)
December 15-18, 2015. Osaka, Japan

Hierarchical Interface-Based Decentralized Supervisory Control*

Huailiang Liu[1], Ryan J. Leduc[2] and S. L. Ricker[3]

Abstract— **The Hierarchical Interface-Based Supervisory Control (HISC) framework was proposed to address challenges inherent in modeling the behavior of large, complex systems. Such systems are often characterized by decentralized or distributed architectures, where agents have only a partial view of the system behavior and cooperate to achieve the control objective, aspects unsupported by HISC. We introduce the Hierarchical Interface-Based Decentralized Supervisory Control (HIDSC) framework that extends HISC to decentralized control.**

In decentralized control, the specification must satisfy a property called co-observability. The verification of co-observability requires the (possibly intractable) construction of the complete system. To adapt this property for HIDSC, we propose a per-component definition of co-observability along with a verification strategy that does not require the construction of the complete system. We provide and prove the necessary and sufficient conditions for supervisor existence in this new framework and illustrate our approach with an example.

I. Introduction

One of the main challenges in the control of Discrete Event Systems (DES) is the combinatorial explosion of the product state space. The Hierarchical Interface-Based Supervisory Control (HISC) framework proposed in [4], [5] can alleviate the state-space explosion problem. HISC provides a set of local properties that can be used to verify global properties, such as nonblocking and controllability, so that the complete system model never needs to be constructed. The sufficient conditions of HISC allow the independent design and verification of different levels, ensuring that a change to one level of the hierarchy will not impact the others.

However, the current HISC framework does not support decentralized control problems which arise naturally through the investigation of a large variety of distributed systems, such as communication networks, integrated sensor networks, networked control systems and automated guided vehicular systems. These systems have many controllers that jointly control the distributed architecture. Further, these controllers may be supervisors with only a partial observation of the system. Also, due to the distributed nature of the system, controllers at different sites in the distributed system may see the effect of different sets of sensors and may control different sets of controllable events. The controllers must coordinate the disabling and enabling of events to realize the legal or desired behavior.

Decentralized control of DES focuses on problems where multiple agents each control and observe some events in a system and must together achieve some prescribed goal. The synthesis of decentralized supervisors requires that the specification satisfies a decentralized property called co-observability [12]. Nevertheless, when the system is very large and composed of many sub-systems, checking co-observability using the existing monolithic method [11] requires the construction of the complete system model, which may be intractable due to the state-space explosion problem.

To address the above problems, we propose an approach called the Hierarchical Interface-Based Decentralized Supervisory Control (HIDSC) framework that allows HISC to manage decentralized control problems. The proposed HIDSC framework is a scalable method that can mitigate the product state-space explosion problem, and make decentralized control scale better. We introduce a per-component co-observability definition which does not require the synchronization of all the components. We then prove that if a system is level-wise co-observable, then it is globally co-observable. This should allow more large problems to be solved. Further, we provide and prove the necessary and sufficient conditions for supervisor existence in HIDSC. We then apply our HIDSC approach to an illustrative example.

This paper is organized as follows. Section II presents a summary of the DES terminology that we will use in this paper. Section III discusses the HISC architecture. In Section IV, a new framework called the HIDSC architecture is introduced. Section V introduces the new level-wise co-observability definition and the HIDSC co-observability theorem. Section VI provides and proves a supervisor existence theorem. In Section VII, we illustrate our HIDSC approach with an example. We present conclusions in Section VIII.

II. Preliminaries

This section provides a review of the key concepts used in this paper. Readers unfamiliar with the notation and definitions may refer to [2].

Event sequences and languages are simple ways to describe DES behavior. Let Σ be a finite set of distinct symbols (*events*), and Σ^* be the set of all finite sequences of events plus ϵ, the *empty string*. A language L over Σ is any subset $L \subseteq \Sigma^*$.

The *concatenation* of two strings $s,\ t \in \Sigma^*$, is written as st. Languages and alphabets can also be concatenated. For $L \subseteq \Sigma^*$ and $\Sigma' \subseteq \Sigma$, the concatenation of the language and event set is defined as $L\Sigma' := \{s\sigma | s \in L,\ \sigma \in \Sigma'\}$. For

*This work was supported by Natural Sciences and Engineering Research Council of Canada (NSERC)

[1]Huailiang Liu and [2]Ryan J. Leduc are with the Department of Computing and Software, McMaster University, Hamilton, ON L8S 4K1, Canada {liuh26, leduc}@mcmaster.ca

[3]S. L. Ricker is with the Department of Mathematics and Computer Science, Mount Allison University, Sackville, NB E4L 1E6, Canada lricker@mta.ca

strings $s,\ t \in \Sigma^*$, we say that t is a *prefix* of s (written $t \leq s$) if $s = tu$, for some $u \in \Sigma^*$. We also say that t can be *extended* to s. The *prefix closure* $\overline{L}$ of a language $L \subseteq \Sigma^*$ is defined as follows: $\overline{L} := \{t \in \Sigma^* | t \leq s \text{ for some } s \in L\}$. A language L is said to be *prefix-closed* if $L = \overline{L}$.

Let $\Sigma = \Sigma_1 \cup \Sigma_2$, $L_1 \subseteq \Sigma_1^*$, and $L_2 \subseteq \Sigma_2^*$. For $i \in \{1, 2\}$, $s \in \Sigma^*$, and $\sigma \in \Sigma$. To capture the notion of partial observation, we define the *natural projection* $P_i : \Sigma^* \to \Sigma_i^*$ according to:

$P_i(\epsilon) := \epsilon$

$$P_i(\sigma) := \begin{cases} \epsilon, \text{ if } \sigma \notin \Sigma_i \\ \sigma, \text{ if } \sigma \in \Sigma_i \end{cases}$$

$P_i(s\sigma) := P_i(s)P_i(\sigma)$

Given any language $L \subseteq \Sigma^*$, the *natural projection of a language* L, is $P_i(L) := \{P_i(s) \mid s \in L\}$.

The *inverse projection* P_i^{-1}: $Pwr(\Sigma_i^*) \to Pwr(\Sigma^*)$ is defined over subsets of languages, where $Pwr(\Sigma_i^*)$ and $Pwr(\Sigma^*)$ denote all subsets of Σ_i^* and Σ^*, respectively. Given any $L_i \subseteq \Sigma_i^*$, the inverse projection of L_i is defined as: $P_i^{-1}(L_i) := \{s \mid P_i(s) \in L_i\}$.

A DES is represented as a tuple: $\mathbf{G} := (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$, with finite state set Q, finite alphabet set Σ, partial transition function $\delta :\ Q \times \Sigma \to Q$, initial state q_0, and the set of marker states Q_m. We use $\delta(q, \sigma)!$ to represent that δ is defined for $\sigma \in \Sigma$ at state $q \in Q$. Function δ can be extended to Σ^* by defining $\delta(q,\ \epsilon) := q$ and $\delta(q,\ s\sigma) := \delta(\delta(q,\ s),\ \sigma)$, provided that $q' = \delta(q,\ s)!$ and $\delta(q',\ \sigma)!$, for $s \in \Sigma^*$ and $q \in Q$. We will always assume that a DES has a finite state and event set, and is deterministic.

For DES $\mathbf{G}$, its *closed behavior* is denoted by $L(\mathbf{G}) := \{s \in \Sigma^* | \delta(q_0,\ s)!\}$. The *marked behavior* of $\mathbf{G}$, is defined as $L_m(\mathbf{G}) := \{s \in L(\mathbf{G}) | \ \delta(q_o, s) \in Q_m\}$. A DES $\mathbf{G}$ is said to be *nonblocking* if $\overline{L_m(\mathbf{G})} = L(\mathbf{G})$.

Let $K \subseteq L_m(\mathbf{G}) \subseteq \Sigma^*$. We say that the language K is $L_m(\mathbf{G})$-closed if $K = \overline{K} \cap L_m(\mathbf{G})$. Thus K is $L_m(\mathbf{G})$-closed if it contains all of its prefixes that belong to $L_m(\mathbf{G})$.

The *synchronous product of languages* L_1 and L_2, denoted by $L_1||L_2$, is defined to be: $L_1||L_2 := P_1^{-1}(L_1) \cap P_2^{-1}(L_2)$.

If both L_1 and L_2 are over the same event set Σ, then their languages have the following property: $L_1||L_2 = P_1^{-1}(L_1) \cap P_2^{-1}(L_2) = L_1 \cap L_2$.

Let $\mathbf{G}_i = (Q_i, \Sigma_i, \delta_i, q_{0,i}, Q_{mi})$, $i = 1, 2$. We define the *synchronous product* $\mathbf{G}_1||\mathbf{G}_2$ as: $(Q_1 \times Q_2,\ \Sigma_1 \cup \Sigma_2,\ \delta,\ (q_{0,1},\ q_{0,2}),\ Q_{m1} \times Q_{m2})$, where $\delta((q_1, q_2),\ \sigma)$ is defined as:

$$\begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)), \text{if } \sigma \in \Sigma_1 \cap \Sigma_2, \delta_1(q_1, \sigma)!, \delta_2(q_2, \sigma)!; \\ (\delta_1(q_1, \sigma), q_2),\ \text{if } \sigma \in \Sigma_1 \backslash \Sigma_2 \text{ and } \delta_1(q_1, \sigma)!; \\ (q_1, \delta_2(q_2, \sigma)),\ \text{if } \sigma \in \Sigma_2 \backslash \Sigma_1 \text{ and } \delta_2(q_2, \sigma)!. \end{cases}$$

In supervisory control, the event set Σ is partitioned into two disjoint sets: the *controllable* event set Σ_c and the *uncontrollable* event set Σ_{uc}. Controllable events can be prevented from happening (disabled) by a supervisor, while uncontrollable events cannot be disabled.

Let K and $L = \overline{L}$ be languages over event set Σ. K is said to be *controllable* with respect to L and Σ_{uc} if and only if, $\overline{K}\Sigma_{uc} \cap L \subseteq \overline{K}$.

For *decentralized control,* there is an index set of $N > 1$ decentralized controllers, $D = \{1, ..., N\}$. These controllers have only a partial view of the system behavior and control only a subset of the controllable events. To describe events that each *decentralized controller* $i \in D$ controls, we use the notation $\Sigma_{c,i} \subseteq \Sigma_c$, where $\cup_{i=1}^N \Sigma_{c,i} = \Sigma_c$. We refer to the set of controllers that control $\sigma \in \Sigma_c$ as $D_c(\sigma) := \{i \in D \,|\sigma \in \Sigma_{c,i}\}$.

To describe events that each decentralized controller $i \in D$ observes, we use the notation $\Sigma_{o,i} \subseteq \Sigma_o$, where $\cup_{i=1}^N \Sigma_{o,i} = \Sigma_o$. We refer to the set of controllers that observe $\sigma \in \Sigma_o$ by $D_o(\sigma) := \{i \in D \,|\sigma \in \Sigma_{o,i}\}$. Correspondingly, the natural projection describing the partial view of each controller is denoted by $P_i :\ \Sigma^* \to \Sigma_{o,i}^*$, for $i \in D$.

For decentralized control with a *conjunctive architecture* [12], the fusion rule is the conjunction of all *local control decisions*, i.e., an event is globally enabled if not locally disabled. We use the conjunctive architecture in this paper.

Definition 1: Let $K \subseteq \Sigma^*$ be the desired language, $i \in D$, and $t \in L(\mathbf{G})$. Then the decision rule for a *local partial-observation decentralized supervisor* is a function: $\mathcal{S}_{P_i}(t) := (\Sigma \backslash \Sigma_{c,i}) \cup \{\sigma \in \Sigma_{c,i} \mid P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{G}) \neq \emptyset\}$. The *conjunction* of $\mathcal{S}_{P_i}$, $i \in D$, denoted by $\mathcal{S}_{Con}$, is defined as: $\mathcal{S}_{Con}(t) := \cap_{i=1}^N \mathcal{S}_{P_i}(t) = \cap_{i=1}^N \mathcal{S}_{P_i}(P_i(t))$.

We note that $\mathcal{S}_{P_i}(t) = \mathcal{S}_{P_i}(P_i(t))$ as the natural projection is idempotent, i.e., $P_i(t) = P_i(P_i(t))$.

Definition 2: Given $\mathbf{G}$ and $\mathcal{S}_{Con}$, the resulting *closed-loop system* is denoted by $\mathcal{S}_{Con}/\mathbf{G}$. The system's *closed behavior* $L(\mathcal{S}_{Con}/\mathbf{G})$, is recursively defined as follows:

I) $\epsilon \in L(\mathcal{S}_{Con}/\mathbf{G})$

II) $t \in L(\mathcal{S}_{Con}/\mathbf{G})$, $\sigma \in \mathcal{S}_{Con}(t)$, and $t\sigma \in L(\mathbf{G})$ if and only if $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{G})$.

Definition 3: Given $\mathbf{G}$ and $\mathcal{S}_{Con}$, we say $\mathcal{S}_{Con}$ is a *decentralized supervisory control* if the decision rule is defined as in Definition 1, and the resulting closed-loop system and closed behavior is defined as in Definition 2.

Definition 4: We say that $\mathcal{S}_{Con}$ is a *nonblocking decentralized supervisory control (NDSC)* for $\mathbf{G}$ if $\overline{L_m(\mathcal{S}_{Con}/\mathbf{G})} = L(\mathcal{S}_{Con}/\mathbf{G})$ where $L_m(\mathcal{S}_{Con}/\mathbf{G}) := L(\mathcal{S}_{Con}/\mathbf{G}) \cap L_m(\mathbf{G})$.

It is useful to introduce a generalization of NDSC in which the supervisory action also includes marking as well as control, since allowing supervisors to add marking information makes them more expressive.

Definition 5: Let $K \subseteq L_m(\mathbf{G})$. We say that $\mathcal{S}_{con}$ is a *marking nonblocking decentralized supervisory control (MNDSC)* for $(K, \mathbf{G})$ if $\overline{L_m(\mathcal{S}_{Con}/\mathbf{G})} = L(\mathcal{S}_{Con}/\mathbf{G})$ where $L_m(\mathcal{S}_{con}/\mathbf{G}) := L(\mathcal{S}_{con}/\mathbf{G}) \cap K$.

Definition 6: Let $\mathcal{S}_{Con}$ be a MNDSC for plant $\mathbf{G} = (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$ and $K \subseteq L_m(\mathbf{G})$, with $L_m(\mathcal{S}_{Con}/\mathbf{G}) = L(\mathcal{S}_{con}/\mathbf{G}) \cap K$ and $L(\mathcal{S}_{Con}/\mathbf{G}) = \overline{K}$. Let $\mathbf{H} = (X,\ \Sigma,\ \xi,\ x_0,\ X_m)$ be a specification automaton. We say that $\mathbf{H}||\mathbf{G}$ has *equivalent MNDSC behavior* with $\mathcal{S}_{Con}/\mathbf{G}$, if $K = L_m(\mathbf{H}) \cap L_m(\mathbf{G})$ and $\overline{K} = L(\mathbf{H}) \cap L(\mathbf{G})$. Alternatively, we say that $\mathbf{H}$ is an *equivalent theoretical implementation of MNDSC* $\mathcal{S}_{Con}$ *for* $\mathbf{G}$.

In this paper, we focus on MNDSC, which is a more expressive supervisory control paradigm. In particular, it will allow us to later introduce a decentralized supervisor existence result that relies on a closed-loop system $(\mathbf{H}||\mathbf{G})$ to be nonblocking, instead of the existing results that require K to be $L_m(\mathbf{G})$-closed. This is essential to adapting decentralized control to the HISC approach as HISC provides a scalable method to verify nonblocking but not $L_m(\mathbf{G})$-closure.

We note that in decentralized control, there is no real implementation of the centralized supervisor $\mathbf{H}$. The above MNDSC $\mathcal{S}_{Con}$, defined as the control policy of the conjunction of a group of decentralized supervisors, is the real supervisory control. Further, for an HISC system, $\mathbf{H}$ will correspond to the theoretical flat supervisor of the system defined in Section III, and will be used to determine if the flat system is nonblocking.

The following is the definition of co-observability adapted from [12], [1], [10]. Co-observability is a necessary condition to synthesize decentralized controllers which ensure that the supervised system generates exactly the behavior of specification K.

Definition 7: Let K, $L = \overline{L}$ be languages over event set Σ. Let $D = \{1, ..., N\}$ be an index set. Let $\Sigma_{c,i} \subseteq \Sigma$ and $\Sigma_{o,i} \subseteq \Sigma$ be sets of controllable and observable events, respectively, for $i \in D$, where $\Sigma_c = \cup_{i=1}^{N} \Sigma_{c,i}$ and $D_c(\sigma) := \{i \in D \,|\sigma \in \Sigma_{c,i}\}$. Let $P_i : \Sigma^* \to \Sigma_{o,i}^*$ be a natural projection. A language K is said to be *co-observable* with respect to L, $\Sigma_{o,i}$, $\Sigma_{c,i}$, $i \in D$, if and only if

$(\forall t \in \overline{K} \cap L)\ (\forall \sigma \in \Sigma_c)\ t\sigma \in L\backslash\overline{K} \Rightarrow (\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L = \emptyset$.

Note that, when $D = \{1\}$, this property is called *observability* [7]. Since, in practice, the specification $\overline{K}$ is not necessarily a subset of L, we do not require that $K \subseteq L$, as is traditionally done. Instead of checking all strings in $\overline{K}$, we check all strings in $\overline{K} \cap L$.

If an event σ needs to be disabled (i.e., $t \in \overline{K}$, $t\sigma \in L\backslash\overline{K}$), then at least one of the controllers that control σ must unambiguously know to disable σ (i.e., $P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L = \emptyset$). In accordance with the conjunctive architecture, a global disablement decision will be to disable sigma.

In the following when there is no ambiguity, instead of saying that K is co-observable with respect to L, $\Sigma_{o,i}$, $\Sigma_{c,i}$, $i \in D$, we will say that K is co-observable w.r.t. L.

III. HISC Architecture

The HISC approach decomposes a system into a high-level subsystem which communicates with $n \geq 1$ parallel low-level subsystems through separate *interfaces* that restrict the interaction of the subsystems. The high-level subsystem communicates with each low-level subsystem through a separate interface.

In HISC there is a master-slave relationship. A high-level subsystem sends a command to a particular low-level subsystem, which then performs the indicated task and returns a response (answer). Fig 1 shows conceptually the structure and information flow of the system. This style of interaction

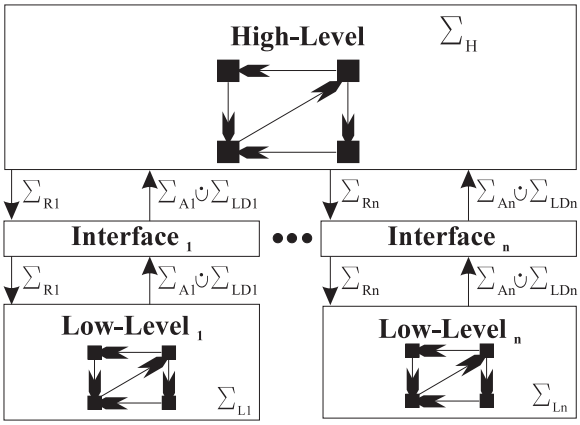

Fig. 1. Interface Block Diagram with Low Data Events.

is enforced by an interface that mediates communication between the two subsystems. All system components, including the interfaces, are modeled as automata.

To restrict information flow and decouple the subsystems, the system alphabet is partitioned into pairwise disjoint alphabets:

$$\Sigma := \Sigma_H \,\dot\cup \bigdot_{j=1,\ldots,n} [\Sigma_{L_j} \dot\cup \Sigma_{R_j} \dot\cup \Sigma_{A_j} \dot\cup \Sigma_{LD_j}] \qquad (1)$$

where we use $\dot\cup$ to represent disjoint union.

The events in Σ_H are called *high-level events* and the events in Σ_{L_j} are the j^{th} *low-level events* ($j \in \{1, \ldots, n\}$) as these events appear only in the high-level and j^{th} low-level subsystem models, $\mathbf{G}_H$ and $\mathbf{G}_{L_j}$, respectively.

Subsystem $\mathbf{G}_H$ is defined over event set $\Sigma_H \dot\cup (\dot\cup_{j\in\{1,\ldots,n\}} [\Sigma_{R_j} \dot\cup \Sigma_{A_j} \dot\cup \Sigma_{LD_j}])$ while $\mathbf{G}_{L_j}$ is defined over event set $\Sigma_{L_j} \dot\cup \Sigma_{R_j} \dot\cup \Sigma_{A_j} \dot\cup \Sigma_{LD_j}$. We model the j^{th} interface by DES $\mathbf{G}_{I_j}$, which is defined over events that are common to both levels of the hierarchy, namely $\Sigma_{R_j} \dot\cup \Sigma_{A_j} \dot\cup \Sigma_{LD_j}$. For the remainder of this paper, the index $j \in \{1, \ldots, n\}$. We define our *flat system* to be $\mathbf{G} = \mathbf{G}_H||\mathbf{G}_{I_1}||\mathbf{G}_{L_1}|| \ \ldots ||\mathbf{G}_{I_n}||\mathbf{G}_{L_n}$. By flat system we mean the equivalent DES if we ignore the interface structure.

The events in Σ_{R_j}, called *request events*, represent commands sent from the high-level subsystem to the j^{th} low-level subsystem. The events in Σ_{A_j} are *answer events* and represent the low-level subsystem's responses to the request events. The events in Σ_{LD_j} are called *low data events* which provide a means for a low-level to send information (data) through the interface. Request, answer, and low data events are collectively known as the set of LD *interface events*, defined as $\Sigma_I := \dot\cup_{k\in\{1,\ldots,n\}} [\Sigma_{R_k} \dot\cup \Sigma_{A_k} \dot\cup \Sigma_{LD_k}]$, and $\mathbf{G}_{I_j}$ is an LD interface [5].

To simplify notation in our exposition, we bring in the following event sets, natural projections, and languages.

$\Sigma_{I_j} := \Sigma_{R_j} \dot\cup \Sigma_{A_j} \dot\cup \Sigma_{LD_j}$, $P_{I_j} : \Sigma^* \to \Sigma_{I_j}^*$,
$\Sigma_{IL_j} := \Sigma_{L_j} \cup \Sigma_{I_j}$, $P_{IL_j} : \Sigma^* \to \Sigma_{IL_j}^*$,
$\Sigma_{IH} := \Sigma_H \cup \bigcup_{k\in\{1,\ldots,n\}} \Sigma_{I_k}$, $P_{IH} : \Sigma^* \to \Sigma_{IH}^*$,

$\mathcal{H} := P_{IH}^{-1}(L(\mathbf{G}_H)), \mathcal{L}_j := P_{IL_j}^{-1}(L(\mathbf{G}_{L_j})) \subseteq \Sigma^*$,
$\mathcal{I} := \cap_{k\in\{1,...,n\}}\mathcal{I}_k, \mathcal{I}_j := P_{I_j}^{-1}(L(\mathbf{G}_{I_j}))$.

We define the *high-level plant* to be $\mathbf{G}_H^p$, and the *high-level supervisor* to be $\mathbf{S}_H$ (both defined over event set Σ_{IH}). Similarly, the j^{th} *low-level plant* and *supervisor* are $\mathbf{G}_{L_j}^p$ and $\mathbf{S}_{L_j}$ (defined over Σ_{IL_j}). The high-level subsystem and the j^{th} low-level subsystem are then $\mathbf{G}_H := \mathbf{G}_H^p||\mathbf{S}_H$ and $\mathbf{G}_{L_j} := \mathbf{G}_{L_j}^p||\mathbf{S}_{L_j}$, respectively. We note that in HISC systems, interfaces are always supervisors.

We can now define our *flat supervisor* and *plant* as well as some other useful languages as follows:

Plant $:= \mathbf{G}_H^p||\mathbf{G}_{L_1}^p||\ldots||\mathbf{G}_{L_n}^p$,
Sup $:= \mathbf{S}_H||\mathbf{S}_{L_1}||\ldots||\mathbf{S}_{L_n}||\mathbf{G}_{I_1}||\ldots||\mathbf{G}_{I_n}$,
$\mathcal{H}^p := P_{IH}^{-1}L(\mathbf{G}_H^p), \mathcal{S}_H := P_{IH}^{-1}L(\mathbf{S}_H) \subseteq \Sigma^*$,
$\mathcal{L}_j^p := P_{IL_j}^{-1}L(\mathbf{G}_{L_j}^p), \mathcal{S}_{L_j} := P_{IL_j}^{-1}L(\mathbf{S}_{L_j}) \subseteq \Sigma^*$.

IV. HIDSC Architecture

In Section III, we described a system composed of plant DES $\mathbf{G}_H^p$, $\mathbf{G}_{L_1}^p, \ldots, \mathbf{G}_{L_n}^p$, supervisor DES $\mathbf{S}_H, \mathbf{S}_{L_1}, \ldots, \mathbf{S}_{L_n}$, and interface DES $\mathbf{G}_{I_1}, \ldots, \mathbf{G}_{I_n}$. Although the level-wise controllability condition [4], [5] does effectively limit the high-level supervisor to events in Σ_{IH}, and the j^{th} low-level supervisor to events in Σ_{IL_j}, it requires the HISC structure and does not allow further restrictions outside of this structure. In order to allow decentralized supervisors within components, we need to extend the HISC structure. We will now introduce the *Hierarchical Interface-based Decentralized Supervisory Control (HIDSC) architecture.*

HIDSC is an extension of HISC from centralized control to a decentralized architecture. In the HIDSC framework, all the HISC supervisors are replaced by corresponding *specification DES*, which are our requirements of the legal behavior of the system. In HIDSC, we will replace supervisor $\mathbf{S}_H$ by specification DES $\mathbf{F}_H$ (defined over Σ_{IH}), and we will replace supervisor $\mathbf{S}_{L_j}$ by specification DES $\mathbf{F}_{L_j}$ (defined over Σ_{IL_j}). Typically, $\mathbf{F}_H$ will express system-wide constraints about how the components interact and what tasks the low-levels should perform. $\mathbf{F}_{L_j}$ expresses how the j^{th} low-level will perform the tasks (requests) given to it by the high-level. For each component, there is a different index set of decentralized controllers.

Definition 8: The n^{th} *degree decentralized specification interface system* with respect to the alphabet partition given by (1) is composed of plant DES $\mathbf{G}_H^p$, $\mathbf{G}_{L_1}^p, \ldots, \mathbf{G}_{L_n}^p$, specification DES $\mathbf{F}_H, \mathbf{F}_{L_1}, \ldots, \mathbf{F}_{L_n}$, interface DES $\mathbf{G}_{I_1}, \ldots, \mathbf{G}_{I_n}$, and high-level and low-level decentralized controllers. The high-level decentralized controllers have an index set $D_H := \{N_{H,1}, \ldots, N_{H,n_0}\}$. The event set for $\mathbf{G}_H^p$, $\mathbf{F}_H$ and the corresponding decentralized controllers is Σ_{IH}. For $i \in D_H$, $\Sigma_{H,c,i} \subseteq \Sigma_c \cap \Sigma_{IH}$ and $\Sigma_{H,o,i} \subseteq \Sigma_o \cap \Sigma_{IH}$ are the corresponding controllable and observable event subsets for the high-level decentralized controllers. Each low-level component has an index set $D_{L_j} := \{N_{L_j,1}, \ldots, N_{L_j,n_j}\}$ for its own decentralized controllers. For $j \in \{1, \ldots, n\}$, the event set of each low-level component $\mathbf{G}_{L_j}^p$, $\mathbf{F}_{L_j}$ and the corresponding decentralized controllers is Σ_{IL_j}. For $i \in D_{L_j}$, $\Sigma_{L_j,c,i} \subseteq \Sigma_c \cap \Sigma_{IL_j}$ and $\Sigma_{L_j,o,i} \subseteq \Sigma_o \cap \Sigma_{IL_j}$ are the corresponding controllable and observable event subsets for the low-level decentralized controllers. The index set for all decentralized controllers in the system is $D := D_H \dot{\cup} \dot{\bigcup}_{j=1}^{n} D_{L_j} = \{1, ..., N\}$.

For the rest of this section, we will refer to such a system as an n^{th} degree decentralized specification interface system Ψ, or simply Ψ. Note that in Ψ, we do not specify the index of decentralized controllers by $\{1, ..., n_0\}$, $\{1, ..., n_j\}$, etc., because once combined they would overlap. We create the system index set using disjoint union.

The *flat system* $\mathbf{G}$ is the synchronization of all the plant, specification, and interface components in the whole system together, i.e., $\mathbf{G} = \mathbf{G}_H^p \,||\, \mathbf{G}_{L_1}^p \,||\, \ldots \,||\, \mathbf{G}_{L_n}^p \,||\, \mathbf{F}_H \,||\, \mathbf{F}_{L_1} \,||\, \ldots \,||\, \mathbf{F}_{L_n} \,||\, \mathbf{G}_{I_1} \,||\, \ldots \,||\, \mathbf{G}_{I_n}$. We use the term *flat system* to mean the overall system ignoring the HIDSC structure.

It is important to note that for an HIDSC system, we would first design level-wise supervisors for the original HISC system while ignoring any decentralized restrictions. We would then use the HISC structure to verify that the system is nonblocking and controllable. We would next use these level-wise supervisors (which include the system's interface DES) as "specifications" for the design of the per-component decentralized supervisors specified by the HIDSC system. The final system would not contain any of these specification DES, just the resulting decentralized controllers that would provide us with equivalent closed-loop behavior (see Corollary 1 in Section VI).

V. HIDSC Co-observability Definition and Theorem

The main focus here is to verify co-observability in an HIDSC system Ψ without explicitly constructing the flat system. We will only perform a per-component co-observability verification, but guarantee that the whole system is co-observable.

To aid in defining our per-component co-observability definition and HIDSC co-observability theorem, we specify some decentralized notations for Ψ.

We use $D_{H,c}(\sigma) := \{i \in D_H \,|\sigma \in \Sigma_{H,c,i}\}$ to denote the set of decentralized controllers in the high-level that can control the event σ.

We use $D_{H,o}(\sigma) := \{i \in D_H \,|\sigma \in \Sigma_{H,o,i}\}$ to denote the set of decentralized controllers in the high-level that can observe the event σ. Correspondingly, $P_{H,i} : \Sigma^* \to (\Sigma_{H,o,i})^*$ is the natural projection describing the partial view of controller $i \in D_H$.

For $j \in \{1, \ldots, n\}$, $D_{L_j,c}(\sigma) := \{i \in D_{L_j} \,|\sigma \in \Sigma_{L_j,c,i}\}$ is the set of decentralized controllers in the j^{th} low-level component that can control the event σ.

For $j \in \{1, \ldots, n\}$, $D_{L_j,o}(\sigma) := \{i \in D_{L_j} \,|\sigma \in \Sigma_{L_j,o,i}\}$ is the set of decentralized controllers in the j^{th} low-level component that can observe the event σ. Correspondingly, $P_{L_j,i} : \Sigma^* \to (\Sigma_{L_j,o,i})^*$ is the natural projection describing the partial view of controller $i \in D_{L_j}$.

Further, we introduce a few languages used for the HIDSC co-observability definition and theorem.

$\mathcal{F}_H := P_{IH}^{-1}(L(\mathbf{F}_H))$, $\mathcal{F}_{L_j} := P_{IL_j}^{-1}(L(\mathbf{F}_{L_j}))$,

$\mathcal{F} := \mathcal{F}_H \cap \mathcal{F}_{L_1} \cap \ldots \cap \mathcal{F}_{L_n}$, $\mathcal{P} := \mathcal{H}^p \cap \mathcal{L}_1^p \cap \ldots \cap \mathcal{L}_n^p$.

Language $\mathcal{F}_H$ represents the behavior of the specification automata in the high-level subsystem, while $\mathcal{F}_{L_j}$ represents the behavior of the specification automata for component j in the low-level subsystem. Language $\mathcal{F}$ represents the global specification of the flat system, and $\mathcal{P}$ represents the behavior of the flat plant.

A. HIDSC Co-observability Definition

We now present the per-component level-wise co-observability definition for HIDSC system Ψ.

Definition 9: Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. Then Ψ is *level-wise co-observable* if for all $j \in \{1, \ldots, n\}$ the following conditions hold:

I) $(\forall t \in \mathcal{F}_H \cap \mathcal{H}^p \cap \mathcal{I})(\forall \sigma \in \Sigma_c) t\sigma \in (\mathcal{H}^p \cap \mathcal{I}) \backslash \mathcal{F}_H \Rightarrow$
$(\exists i \in D_{H,c}(\sigma))\ P_{H,i}^{-1}[P_{H,i}(t)]\sigma \cap \mathcal{F}_H \cap \mathcal{H}^p \cap \mathcal{I} = \emptyset$

II) $(\forall t \in \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p)(\forall \sigma \in \Sigma_c) t\sigma \in \mathcal{L}_j^p \backslash (\mathcal{F}_{L_j} \cap \mathcal{I}_j) \Rightarrow$
$(\exists i \in D_{L_j,c}(\sigma)) P_{L_j,i}^{-1}[P_{L_j,i}(t)]\sigma \cap \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p = \emptyset$.

Definition 9 states that HIDSC system Ψ is *level-wise co-observable* if the system is co-observable at the high-level component and at each low-level component.

We note that the interfaces are treated as specifications at the low-level and treated as plants at the high-level. This is done this way because interfaces represent the behavior provided by its low-level and the information needed to verify that it is co-observable is typically present at the low-level but not the high-level. To avoid having to repeat this information at the high-level, we use the results of [8] that allow us to treat supervisors as if they are plants once we verify they are co-observable. By treating interfaces as plants at the high-level, we allow the high-level supervisor to be more permissive in general as there will typically be fewer strings that can cause co-observability verification to fail.

Definition 10: Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. Then Ψ is *globally co-observable* if

$(\forall t \in \mathcal{F} \cap \mathcal{I} \cap \mathcal{P})\ (\forall \sigma \in \Sigma_c)\ t\sigma \in \mathcal{P} \backslash (\mathcal{F} \cap \mathcal{I}) \Rightarrow (\exists i \in D_c(\sigma)) P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F} \cap \mathcal{I} \cap \mathcal{P} = \emptyset$.

Definition 10 states that for HIDSC system Ψ, if the global specification $\mathcal{F}$ synchronized with interface $\mathcal{I}$ is co-observable with respect to the flat plant system $\mathcal{P}$, then Ψ is globally co-observable.

Note that Definition 10 is the property we want to verify but we will do so by using our per-component co-observability definition. We do not need to combine the flat specifications, interfaces and plant components together. This potentially saves computation and helps to alleviate the state-space explosion problem.

B. HIDSC Co-observability Theorem

The following is the HIDSC co-observability theorem which states that the level-wise co-observability property is sufficient to guarantee that the flat system is co-observable.

Theorem 1: Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. If Ψ is level-wise co-observable then Ψ is globally co-observable.

Proof: See proof in [9]. ■

C. Complexity Analysis

The following is the complexity analysis of our verification method to which we compare monolithic verification.

In monolithic verification, the n low-level subsystems are composed directly with the high-level system without using the interface structure. The size of the state space for the monolithic method is the size of the product state space of $\mathbf{G}_H \,||\, \mathbf{G}_{L_1} \,||\, \ldots \,||\, \mathbf{G}_{L_n}$. If the size of the state space of $\mathbf{G}_H$ is bounded by N_H, and each of the size of the state space of $\mathbf{G}_{L_j}$ is bounded by N_L, then the size of the state space of the monolithic method is bounded by $N_H N_L^n$.

Our method verifies each component separately, therefore the size of the state space is bounded by the size of the component combined with its interface. For $j = 1, ..., n$, if we assume that the size of the state space of $\mathbf{G}_{I_j}$ is bounded by N_I, then each low-level subsystem $\mathbf{G}_{L_j}||\mathbf{G}_{I_j}$ is bounded by $N_L N_I$. The high-level subsystem $\mathbf{G}_H||\mathbf{G}_{I_1}||\ldots||\mathbf{G}_{I_n}$ is bounded by $N_H N_I^n$. Therefore, our method is bounded by the larger of $N_H N_I^n$ and $N_L N_I$. Typically in an HIDSC design, the size of the high-level is the limiting factor. This means that as long as $N_I \ll N_L$, we should achieve significant computational savings [6].

VI. MNDSC Supervisor Existence Theorem

We now present the marking nonblocking decentralized supervisory control (MNDSC) existence theorem, which shows that there exists an MNDSC to achieve the specification if and only if K is controllable and co-observable.

Theorem 2: Let **Plant** $:= (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$, $K \subseteq L_m(\textbf{Plant})$, and $K \neq \emptyset$. There exists an MNDSC $\mathcal{S}_{Con}$ for (K, **Plant**) such that $L_m(\mathcal{S}_{Con}/\textbf{Plant}) = K$ if and only if K is controllable and co-observable with respect to $L(\textbf{Plant})$.

Proof: See proof in [9]. ■

Note that in Theorem 2 we do not require that K be $L_m(\mathbf{G})$-closed which is assumed by traditional decentralized control [2]. This will allow us to apply the result to our HIDSC system as we have an HISC nonblocking result but not an HISC $L_m(\mathbf{G})$-closed result.

We will now relate Theorem 2 to our HIDSC system and nonblocking. In essence, we are requiring Ψ to have equivalent MNDSC behavior with $\mathcal{S}_{Con}/$**Plant**, which ensures our HIDSC system implementation will be nonblocking.

Corollary 1: Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. Let **Plant** $:= \mathbf{G}_H^p||\mathbf{G}_{L_1}^p||\ldots||\mathbf{G}_{L_n}^p$, **Spec** $:= \mathbf{F}_H||\mathbf{F}_{L_1}||\ldots||\mathbf{F}_{L_n}||\mathbf{G}_{I_1}||\ldots||\mathbf{G}_{I_n}$. Let $L_m(\textbf{Spec}) \cap L_m(\textbf{Plant}) \neq \emptyset$. There exists an MNDSC $\mathcal{S}_{Con}$ for ($L_m(\textbf{Spec}) \cap L_m(\textbf{Plant})$, **Plant**) such that $L_m(\mathcal{S}_{Con}/\textbf{Plant}) = L_m(\textbf{Spec}) \cap L_m(\textbf{Plant})$, and $L(\mathcal{S}_{Con}/\textbf{Plant}) = L(\textbf{Spec}) \cap L(\textbf{Plant})$, if and only if $L_m(\textbf{Spec}) \cap L_m(\textbf{Plant})$ is controllable

and co-observable with respect to $L(\textbf{Plant})$, and $\overline{L_m(\textbf{Spec}) \cap L_m(\textbf{Plant})} = L(\textbf{Spec}) \cap L(\textbf{Plant})$.

Proof: See proof in [9]. ■

For HIDSC system Ψ, Corollary 1 tells us that the marked behavior of our MNDSC and flat plant is equal to $L_m(\textbf{Spec}) \cap L_m(\textbf{Plant})$ and their closed behavior is equal to $L(\textbf{Spec}) \cap L(\textbf{Plant})$. To apply Corollary 1, we need to first show that Ψ is co-observable, nonblocking, and controllable. For scalability, we want to verify all these global properties using only per-component properties. Theorem 1 shows us that level-wise co-observability gives us global co-observability. From [4], [5], we know that the HISC LD level-wise nonblocking, LD interface consistent, and level-wise controllability properties together imply that our flat system is nonblocking and controllable. We can thus verify all needed global properties using per-component check. As we never need to construct the full system model, this offers potentially great computational savings.

VII. Example

To demonstrate the HIDSC method, we adapt a small manufacturing system from [4], that was originally modeled as an HISC system, shown in Fig 2. The system is composed of three manufacturing units running in parallel, a testing unit, material feedback, a packaging unit, plus three buffers to insure the proper flow of material.

Fig 3 shows which DES belong to the high level subsystem ($\mathbf{G}_H$), high-level plant ($\mathcal{G}_H$), the high-level specification automata ($\mathcal{S}_H$), the j^{th} low-level subsystem ($\mathbf{G}_{L_j}$), the j^{th} low-level plant ($\mathcal{G}_{L_j}$), the j^{th} low-level specification automata ($\mathcal{S}_{L_j}$), and the j^{th} interface DES ($\mathbf{G}_{I_j}$), $j = \text{I, II, III}$. We note that the three low-level subsystems shown in Fig 2 and 3 are identical up to relabeling. Fig 4 shows the low-level subsystems in detail.

Controllable events are those with a slash on the transition arrow, marked states are states with an unlabeled incoming arrow, and initial states are states with an unlabeled outgoing arrow.

A. A Manufacturing System as an HIDSC

Originally this example was modeled as an HISC system. We will now adapt it to an HIDSC system. Typically, we would only do this if the system had an inherent distributed nature forcing us to implement supervisors with partial observations and partial controllability beyond the compartmentalized limitation imposed by the HISC structure.

We define the alphabet partition $\Sigma := [\dot\cup_{k\in\{\text{I,II,III}\}}(\Sigma_{L_k} \dot\cup \Sigma_{R_k} \dot\cup \Sigma_{A_k})] \dot\cup \Sigma_H$ below:

$$\Sigma_H = \{\textit{take\_item}, \textit{package}, \textit{allow\_exit}, \textit{new\_part}, \textit{part\_f\_obuff}, \textit{part\_passes}, \textit{part\_fails}, \textit{ret\_inbuff}, \textit{deposit\_part}\}$$

$$\Sigma_{R_j} = \{\textit{part\_ent-j}\}$$

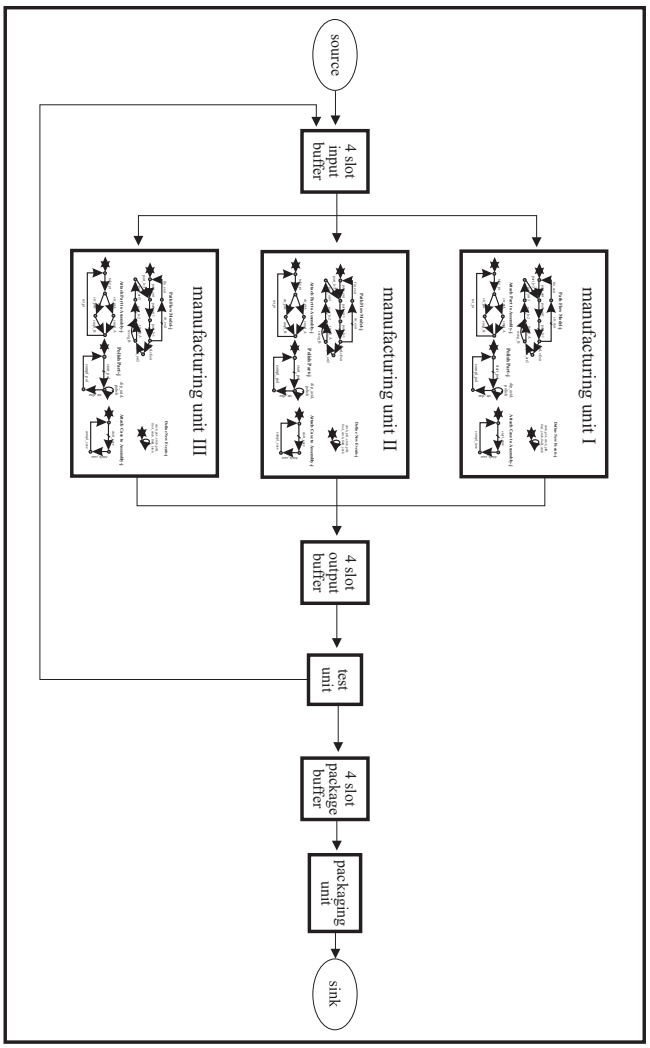

Fig. 2. Block Diagram of Parallel Plant System.

$$\Sigma_{A_j} = \{\textit{fin\_exit-j}\}$$

$$\Sigma_{L_j} = \{\textit{start\_pol-j}, \textit{attch\_ptA-j}, \textit{attch\_ptB-j}, \textit{start\_case-j}, \textit{comp\_pol-j}, \textit{finA\_attch-j}, \textit{finB\_attch-j}, \textit{compl\_case-j}, \textit{part\_arr1-j}, \textit{part\_lv1-j}, \textit{partLvExit-j}, \textit{str\_exit-j}, \textit{part\_arr2-j}, \textit{recog\_A-j}, \textit{recog\_B-j}, \textit{part\_lv2-j}, \textit{part\_arr3-j}, \textit{part\_lv3-j}, \textit{take\_pt-j}, \textit{str\_ptA-j}, \textit{str\_ptB-j}, \textit{compl\_A-j}, \textit{compl\_B-j}, \textit{ret\_pt-j}, \textit{dip\_acid-j}, \textit{polish-j}, \textit{str\_rlse-j}\}$$

Our first step is to replace the existing supervisors with specification automata. Thus let $\mathbf{F}_H = \mathcal{S}_H$ and $\mathbf{F}_{L_j} = \mathcal{S}_{L_j}$, $j = \text{I, II, III}$.

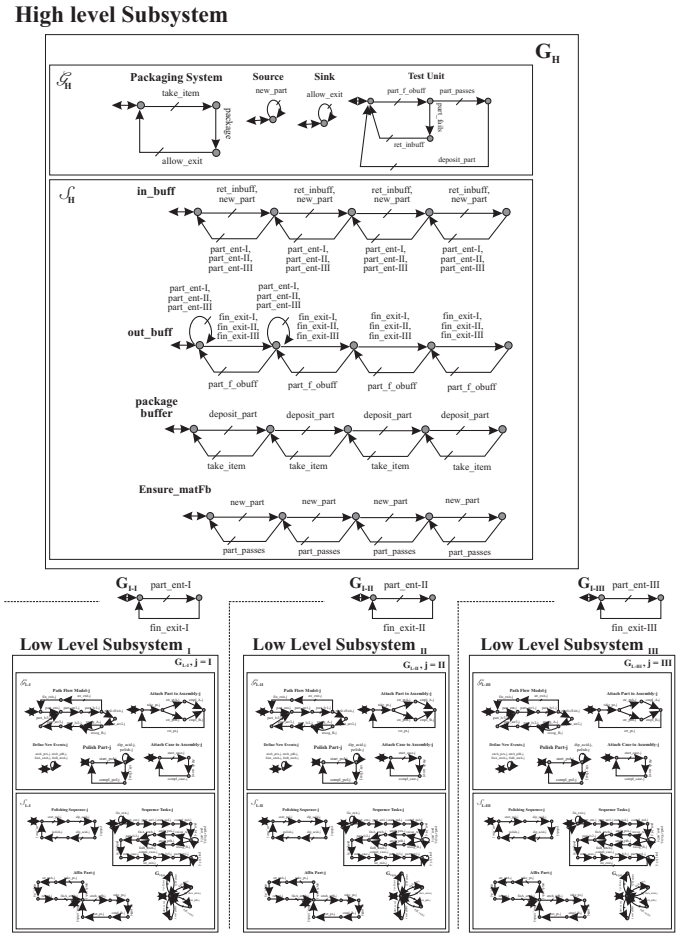

Fig. 3. *Complete Parallel System.*

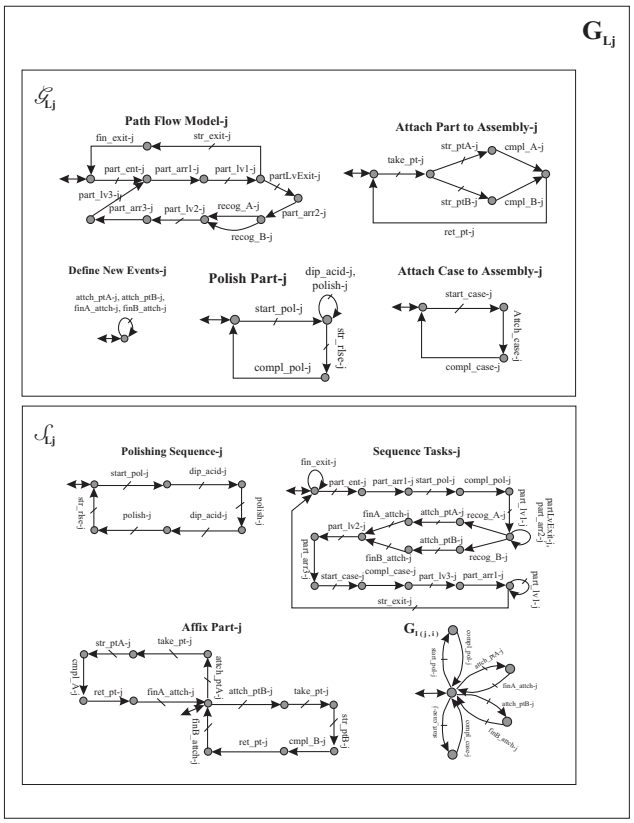

Fig. 4. *Low Level Subsystem j.*

Next, we design five decentralized controllers (H1, H2, L1, L2 and L3) to define our HIDSC problem.

Controller H1 can only observe interface events, and can only control controllable interface events (i.e., $\Sigma_c \cap \Sigma_I$). Hence events in $\Sigma_H \cup \Sigma_{L_I} \cup \Sigma_{L_{II}} \cup \Sigma_{L_{III}}$ are all unobservable and uncontrollable to H1, therefore can be safely ignored.

Controller H2 can only observe events in Σ_H, and can only control controllable events in Σ_H. Therefore, events in the low-level components and interfaces can be safely ignored.

In each low-level subsystem, a controller can only observe and control events in its own subsystem. For example, controller L1 can only observe and control events in $\mathbf{G}_{L_I}$ and $\mathbf{G}_{I_I}$, i.e., events in Σ_{IL_I}. Therefore high-level events and other low-levels can be safely ignored. Analogously, this is also true for controllers L2 and L3.

The index sets of decentralized controllers for each component are: D_H={H1, H2}, D_{L_I}={L1}, $D_{L_{II}}$={L2}, $D_{L_{III}}$={L3}.

We now define the *flat system,* the *flat plant,* and the *flat specification automata* as follows:

Plant $:= \mathcal{G}_H || \mathcal{G}_{L_{\text{I}}} || \mathcal{G}_{L_{\text{II}}} || \mathcal{G}_{L_{\text{III}}}$

Spec $:= \mathbf{F}_H || \mathbf{F}_{L_I} || \mathbf{F}_{L_{II}} || \mathbf{F}_{L_{III}} || \mathbf{G}_{I_{\text{I}}} || \mathbf{G}_{I_{\text{II}}} || \mathbf{G}_{I_{\text{III}}}$

B. Co-observability Verification for the Decentralized System

We need to verify whether $L_m(\mathbf{Spec})$ is co-observable w.r.t. $L(\mathbf{Plant})$. We can then conclude, in combination with checking controllability, by Theorem 2 that there exists an MNDSC decentralized supervisory control. By Theorem 1, we know it is sufficient to verify level-wise co-observability.

The following steps for level-wise co-observability verification are:

Step 1. Verify whether the first low-level subsystem satisfies its portion of the level-wise co-observable definition, i.e., whether $L(\mathbf{F}_{L_{\text{I}}} || \mathbf{G}_{I_{\text{I}}})$ is co-observable w.r.t. $L(\mathbf{G}_{L_{\text{I}}})$,$\Sigma_{L,c,i}$, $\Sigma_{L,o,i}$ for $i \in D_{L_I}$.

L1 is the only decentralized supervisor for D_{L_I}. We find that controller L1 can observe all the events in the first low-level subsystem, i.e., $\Sigma_{L_1,o,1} = \{$*part_ent-I*, *fin_exit-I*$\} \dot{\cup} \Sigma_{L_I} = \Sigma_{IL_I}$. Further, controller L1 can control all the events in the first low-level subsystem, i.e., $\Sigma_{L_1,c,1} = \{$*part_ent-I*$\} \dot{\cup} \Sigma_{L_I} \cap \Sigma_c = \Sigma_{IL_I} \cap \Sigma_c$. Therefore, the first low-level component trivially satisfies its portion of the level-wise co-observable definition.

Step 2. Verify whether the second low-level subsystem satisfies its portion of the level-wise co-observable definition, i.e., whether $L(\mathbf{F}_{L_{\text{II}}} || \mathbf{G}_{I_{\text{II}}})$ is co-observable w.r.t. $L(\mathbf{G}_{L_{\text{II}}})$, $\Sigma_{L,c,i}$, $\Sigma_{L,o,i}$ for

$i \in D_{L_{II}}$. We note that since the second low-level subsystem is identical up to relabeling with the first one, this step does not need to be verified.

Step 3. Verify whether the third low-level subsystem satisfies its portion of the level-wise co-observable definition. Similar to Step 2, this step does not need to be verified.

Step 4. Verify whether the high-level subsystem satisfies its portion of the level-wise co-observable definition, i.e., verifying whether $L(\mathbf{F}_H)$ is co-observable w.r.t. $L(\mathcal{G}_H||\mathbf{G}_{I_{\rm I}}||\mathbf{G}_{I_{\rm II}}||\mathbf{G}_{I_{\rm III}})$, $\Sigma_{H,c,i}$, $\Sigma_{H,o,i}$, for $i \in D_H$.

In the high-level subsystem, each controllable event in Σ_{IH} can be controlled by either H1 or H2, but not both. In other words, $\Sigma_{H,c,1} \cap \Sigma_{H,c,2} = \emptyset$, $\Sigma_{H,o,1} \cap \Sigma_{H,o,2} = \emptyset$.

Observability is a special case of co-observability, and if a system is observable for each controller independently, then it is definitely co-observable [2], [12]. This means the problem can be reduced to whether the high-level is observable for H1 and H2 separately.

Using the DES design software TCT [13], we find that the high-level is observable for H1 and H2 separately. When we verify H1 by TCT, we specify that all controllable and observable events are within Σ_I. Correspondingly, when we verify H2, we specify that events in Σ_I are all uncontrollable and unobservable. We thus conclude that the high-level satisfies its portion of the level-wise co-observable definition.

By completing Steps 1-4, we conclude that the decentralized system is level-wise co-observable, thus globally co-observable by Theorem 1.

Using our software tool DESpot [3], we verified that the system is level-wise controllable, LD level-wise nonblocking, and LD interface consistent. We can thus conclude by [5] that our flat system is controllable and nonblocking. We conclude by Corollary 1 that there exists a marking nonblocking decentralized supervisory control $\mathcal{S}_{Con}$ for **Plant**, and that **Spec**||**Plant** has equivalent MNDSC behavior with $\mathcal{S}_{Con}$/**Plant**. This means that since **Spec**||**Plant** is nonblocking, $\mathcal{S}_{Con}$/**Plant** is also nonblocking.

C. Complexity Analysis for the Decentralized System

Applying DESpot to the small manufacturing system example, we found that the state size of the entire system was 2.78×10^{10}. However, the high-level state size was 3.12×10^3 and the low-level state size was 5.5×10^2. This is a potential savings of about seven orders of magnitude.

The computational complexity to verify co-observability using the monolithic approach in [11] is $O(|\Sigma|^{n+2}|Y|^{n+2})$, where Σ is the event set, Y is the state space, and n is the number of decentralized controllers. Substituting in the small manufacturing system example, verifying co-observability using the above method gives a complexity bounded by $|42|^{5+2}|2.78 \times 10^{10}|^{5+2} = 2.96 \times 10^{84}$. Using our method, the complexity is bounded by $|15|^{2+2}|3.12 \times 10^3|^{2+2} = 4.8 \times 10^{18}$. The potential computation saving is a 65 order of magnitude reduction.

VIII. Conclusion

The existing HISC method does not support decentralized control. In this paper, we extended HISC to the decentralized architecture HIDSC. We introduced per-component co-observability verification which avoids the explicit construction of the complete system model. We then proved that if a system is level-wise co-observable, it is also globally co-observable. This verification method should be very useful for decentralized systems with many components, and should allow us to work with large distributed systems. Further, we provided a supervisory control existence theorem for HIDSC systems, and proved the necessary and sufficient conditions for decentralized control in HIDSC. Finally, we use an example to demonstrate the HIDSC approach.

References

[1] G. Barrett and S. Lafortune, "Decentralized supervisory control with communicating controllers," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1620–1638, 2000.

[2] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems (second edition)*. Springer, 2008.

[3] DESpot, "The official website for the despot project," [Online]. Available: http://www.cas.mcmaster.ca/~leduc/DESpot.html, 2014.

[4] R. J. Leduc, "Hierarchical interface-based supervisory control," Ph.D. dissertation, Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ont., 2002, [ONLINE] Available: http://www.cas.mcmaster.ca/~leduc.

[5] ——, "Hierarchical interface-based supervisory control with data events," *International Journal of Control*, vol. 82, no. 5, pp. 783–800, 2009.

[6] R. J. Leduc, M. Lawford, and W. M. Wonham, "Hierarchical interface-based supervisory control-part II: parallel case," *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1336–1348, 2005.

[7] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Infomation Sciience*, vol. 44, pp. 173–198, 1988.

[8] H. Liu, R. J. Leduc, R. Malik, and S. L. Ricker, "Incremental verification of co-observability in discrete-event systems," in *Proc. of 2014 American Control Conference*, June 2014, pp. 5446–5452.

[9] H. Liu, R. J. Leduc, and S. L. Ricker, "Decentralized control using the hierarchical interface-based supervisory control approach," Technical Reports CAS-14-10-RL. Department of Computing and Software, McMaster University [Online]. Available: http://www.cas.mcmaster.ca/~leduc, December 2014.

[10] L. Ricker and B. Caillaud, "Mind the gap: Expanding communication options in decentralized discrete-event control," *Automatica*, vol. 47, no. 11, pp. 2364–2372, 2011.

[11] K. Rudie and J. C. Willems, "The computational complexity of decentralized discrete-event control problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 7, pp. 1313–1319, 1995.

[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] TCT, "The official website for the design software: TCT," [Online]. Available: http://www.control.utoronto.ca/~wonham/, 2014.