THINK CONTINUOUS, ACT DISCRETE: DES TECHNIQUES FOR CONTINUOUS SYSTEMS

T. Moor[1], **J. Raisch**[2]

[1]Research School of Information Sciences and Engineering
Australian National University, Canberra
e-mail: *thomas.moor@anu.edu.au*

[2] Lehrstuhl für Systemtheorie technischer Prozesse
Otto-von-Guericke Universität, Magdeburg, Germany
and Max-Planck-Institut für Dynamik komplexer technischer Systeme, Magdeburg, Germany
e-mail: *raisch@mpi-magdeburg.mpg.de*

Keywords: hybrid systems, supervisory control, discrete abstractions, modular control, decentralised control.

Abstract. This contribution addresses the synthesis of discrete supervisors for continuous plants, with a particular emphasis on modular and decentralised control architecture. We follow a well known style of reasoning that has been originally developed in the context of DESs and demonstrate its applicability to the design of hybrid systems.

1 Introduction

Many technical processes which are naturally modelled by continuous dynamical systems are subject to discrete event control. Examples are complex production processes where high level control switches between distinct modes of continuous operation. Another example is the discrete allocation of resources to a process configuration that intrinsically depends on continuous dynamics. The resulting heterogeneous control systems, consisting of both continuous and discrete event components, are commonly referred to as hybrid systems [1, 3, 15].

The *overall* state space of a hybrid system is neither finite nor does it inherit the vector space structure of the continuous components. This is precisely why a hybrid system is neither directly amenable to the use of standard DES methods nor can traditional techniques from continuous control be applied. Finite discrete abstractions of the continuous dynamics seem a straightforward approach to escape this dilemma. This approach translates a hybrid control problem into a purely discrete one, which can subsequently be solved by adapting DES methods; e.g. [12, 18]. However, there remains the crucial question whether solutions on the abstraction level carry over to the actual hybrid control configuration.

When addressing language inclusion type control synthesis problems, the answer is affirmative if the abstraction is conservative; e.g. [8, 10]. In this paper, we extend our approach in order to address modular and decentralised control architectures. Our principle strategy is to adapt DES techniques to the typical input/output structure of continuous systems, where we use *J.C. Willems*' behavioural system theory [17] as a formal framework for our discussion. As our main result, we obtain methods for the design of general a class of modular/decentral hybrid systems.

The paper is organised as follows. In Section 2, we summarise our general framework for the discrete supervisory control of continuous systems. Section 3 provides results regarding abstraction based supervisory controller synthesis. In Section 4 and 5 we present extensions to modular and decentralised control, respectively.

2 Supervisory control of I/- behaviours

In this section, we summarise a fairly general framework for the discrete supervisory control of continuous systems [8, 10]. Our case is motivated by the following principle: regardless of the particularities of a specific plant, a discrete supervisor can only interact via discrete events. Hence, the plant must be equipped with suitable actuators and sensors that mediate between continuous signals and discrete events. If we assume that this external interface is given, the problem of supervisory controller synthesis can be discussed in analogy to *P.J. Ramadge* and *W.M. Wonham's* supervisory control theory for discrete event systems; e.g. [12, 18].

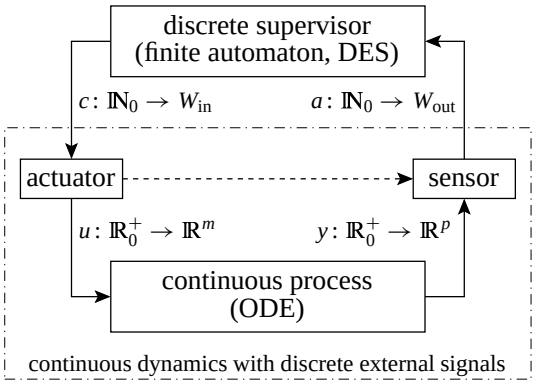

Figure 1: Hybrid closed-loop system

An example of a hybrid closed-loop configuration is depicted in Figure 1. Detailed mathematical models have been developed in e.g. [1, 3, 9, 10, 11, 15], including configurations based on *clock time* (events occur at a fixed sampling rate) and *logic time* (events may occur at any time, e.g. when continuous variables cross critical thresholds). In contrast to the basic DES framework, we will assume that the plant inherits the input/output structure from the underlying continuous system; i.e. the external signal space is a product composition of input and output components rather than a disjoint union of controllable and uncontrollable events. As we shall see in Section 3, this facilitates abstraction based supervisory controller synthesis methods.

Our discussion is set within *J.C. Willems'* behavioural systems theory; e.g. [17, 16]. There, a dynamical system is a model of a phenomenon and a behaviour is the set of all trajectories on which the phenomenon can —according to the model— possibly evolve. Similar to approaching DESs via the formal languages they generate, *Willems* suggests to discuss dynamical systems in terms of their behaviours.

Definition 2.1. (See [17], Def. II.1) A *dynamical system* is a triple $\Sigma = (T, W, \mathfrak{B})$, with $T \subseteq \mathbb{R}$ the *time axis*, W the *signal space*, and $\mathfrak{B} \subseteq W^T$ the *behaviour*. [1] □

In this paper, we focus attention on behaviours that externally evolve on discrete event signals; [2] i.e., we assume $T = \mathbb{N}_0$ and $|W| \in \mathbb{N}$. The traditional notion of inputs and outputs is captured by the following definition of *I/- behaviours*.

Definition 2.2. (see [17], Defs. VIII.1, VIII.4) A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ over the signal space $W = W_{\text{in}} \times W_{\text{out}}$ is an *I/- behaviour* if: [3] [4]

(i) the input is *free*, i.e. $\mathcal{P}_{\text{in}}\mathfrak{B} = W_{\text{in}}^{\mathbb{N}_0}$; and

(ii) the output *does not anticipate* the input, i.e. for all $k \in \mathbb{N}_0$, $\tilde{w}$, $\hat{w} \in \mathfrak{B}$ with $\mathcal{P}_{\text{in}}\tilde{w}|_{[0,k]} = \mathcal{P}_{\text{in}}\hat{w}|_{[0,k]}$ there exists a $w \in \mathfrak{B}$ such that $\mathcal{P}_{\text{out}}w|_{[0,k]} = \mathcal{P}_{\text{out}}\tilde{w}|_{[0,k]}$ and $\mathcal{P}_{\text{in}}w = \mathcal{P}_{\text{in}}\hat{w}$. □

Adapting *Ramadge* and *Wonham*'s supervisory control theory for DESs, the task of a supervisor is to restrict the plant behaviour $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$ such that the closed loop is guaranteed to evolve within a specified set of acceptable signals, denoted $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$. In analogy to the plant, the supervisor is represented by a behaviour $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$, denoting the set of external signals it can evolve on. The closed-loop

[1] The set of maps from T to W is denoted $W^T := \{w \mid w : T \to W\}$.

[2] $\mathbb{N}$ denotes the positive integers (without zero); let $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. We write $|A| \in \mathbb{N}$ to indicate that the set A has only finite many elements and $|A| \neq \emptyset$.

[3] Unlike *Willems*' I/O systems ([17], Def. VIII.3), we do not require the output to *process* the input. The slightly weaker notion of I/- behaviours is motivated by the target class of applications: we cannot assume that the discrete event interface reveals sufficient information about the underlying continuous dynamics such that the past of the discrete signal together with input would uniquely determine the future evolution.

[4] By $\mathcal{P}_{\text{in}}$ and $\mathcal{P}_{\text{out}}$ we denote the natural projections from $W^{\mathbb{N}_0}$ to the input and output component, respectively; i.e. $\mathcal{P}_{\text{in}}w = c$ and $\mathcal{P}_{\text{out}}w = a$ for $w = (c, a)$, $c \in W_{\text{in}}^{\mathbb{N}_0}$, $a \in W_{\text{out}}^{\mathbb{N}_0}$.

behaviour is the intersection $\mathfrak{B}_{\rm cl} = \mathfrak{B}_{\rm p} \cap \mathfrak{B}_{\rm sup}$, i.e. only those external signals can occur in the closed-loop configuration that are consistent with both plant and controller dynamics. The supervisor $\mathfrak{B}_{\rm sup}$ is said to *enforce the specification* $\mathfrak{B}_{\rm spec}$ if $\mathfrak{B}_{\rm cl} \subseteq \mathfrak{B}_{\rm spec}$. However, two conditions apply for the interconnection of $\mathfrak{B}_{\rm p}$ and $\mathfrak{B}_{\rm sup}$. We state and motivate these *admissibility criteria* in terms of behaviours.

The first condition on behavioural interconnection ensures that the synchronisation of the external signal is performed "locally on the time axis": at any instance of time and independent of the past trajectory, it shall be clear on which value the two systems can agree without "getting stuck" in their future evolution. This requirement corresponds to the notion of *nonconflicting* languages in DES theory.

Definition 2.3. (See [8]) Two behaviours $\mathfrak{B}_{\rm p}$, $\mathfrak{B}_{\rm sup} \subseteq W^{\mathbb{N}_0}$ are said to be *nonconflicting* if $\mathfrak{B}_{\rm p}|_{[0,k]} \cap \mathfrak{B}_{\rm sup}|_{[0,k]} = (\mathfrak{B}_{\rm p} \cap \mathfrak{B}_{\rm sup})|_{[0,k]}$ for all $k \in \mathbb{N}_0$. □

Our second condition on behavioural interconnection specifically addresses I/- behaviours: the supervisor may enable or disable certain plant input events but must not impose restrictions on the plant output.

Definition 2.4. (See also [8]) [5] [6] [7] A behaviour $\mathfrak{B}_{\rm sup} \subseteq W^{\mathbb{N}_0}$, $W = W_{\rm in} \times W_{\rm out}$, is *generically implementable* if $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_{\rm sup}|_{[0,k]}$, $\tilde{w}|_{[0,k]} \in W^{k+1}$, and $\tilde{w}|_{[0,k]} \approx_{\rm y} w|_{[0,k]}$ implies $\tilde{w}|_{[0,k]} \in \mathfrak{B}_{\rm sup}|_{[0,k]}$. □

Our supervisory control problem is summarised as follows:

Definition 2.5. (See also [8], Def. 16) Given a plant behaviour $\mathfrak{B}_{\rm p} \subseteq W^{\mathbb{N}_0}$, $W = W_{\rm in} \times W_{\rm out}$, and a specification $\mathfrak{B}_{\rm spec} \subseteq W^{\mathbb{N}_0}$, we call the pair $(\mathfrak{B}_{\rm p},\ \mathfrak{B}_{\rm spec})$ a *supervisory control problem*. We say that

(i) a supervisor $\mathfrak{B}_{\rm sup} \subseteq W^{\mathbb{N}_0}$ is *admissible* w.r.t. the plant $\mathfrak{B}_{\rm p}$ if $\mathfrak{B}_{\rm p}$ and $\mathfrak{B}_{\rm sup}$ are nonconflicting. and $\mathfrak{B}_{\rm sup}$ is generically implementable;
(ii) a supervisor $\mathfrak{B}_{\rm sup} \subseteq W^{\mathbb{N}_0}$ *enforces the specification* $\mathfrak{B}_{\rm spec}$ if $\mathfrak{B}_{\rm cl} := \mathfrak{B}_{\rm p} \cap \mathfrak{B}_{\rm sup} \subseteq \mathfrak{B}_{\rm spec}$;
(iii) a supervisor $\mathfrak{B}_{\rm sup}$ is a *solution* of $(\mathfrak{B}_{\rm p},\ \mathfrak{B}_{\rm spec})$ if $\mathfrak{B}_{\rm sup}$ is admissible to $\mathfrak{B}_{\rm p}$ and enforces $\mathfrak{B}_{\rm spec}$;
(iv) a solution $\mathfrak{B}_{\rm sup}$ is *nontrivial* if it imposes a nontrivial closed-loop behaviour $\mathfrak{B}_{\rm cl} \neq \emptyset$. □

It is readily observed that $\mathfrak{B}_{\rm sup} = \emptyset$ is a trivial solution to any supervisory control problem. Moreover, $\mathfrak{B}_{\rm sup} = \emptyset$ is the *only* trivial solution:

Proposition 2.6. Let the supervisor $\mathfrak{B}_{\rm sup} \subseteq W^{\mathbb{N}_0}$ be admissible w.r.t. an I/- behaviour $\mathfrak{B}_{\rm p} \subseteq W^{\mathbb{N}_0}$, $W = W_{\rm in} \times W_{\rm out}$. If $\mathfrak{B}_{\rm sup} \neq \emptyset$, then $\mathfrak{B}_{\rm p} \cap \mathfrak{B}_{\rm sup} \neq \emptyset$.

Proof. Pick $w \in \mathfrak{B}_{\rm sup}$. As the input of $\mathfrak{B}_{\rm p}$ is free, there exists $\tilde{w} \in \mathfrak{B}_{\rm p}$ with $\mathcal{P}_{\rm in}\tilde{w} = \mathcal{P}_{\rm in}w$, and, hence, $\tilde{w}|_{[0,0]} \approx_{\rm y} w|_{[0,0]}$. From generic implementability we obtain $\tilde{w}|_{[0,0]} \in \mathfrak{B}_{\rm sup}|_{[0,0]} \cap \mathfrak{B}_{\rm p}|_{[0,0]}$. The nonconflicting property ensures that $\tilde{w}|_{[0,0]} \in (\mathfrak{B}_{\rm sup} \cap \mathfrak{B}_{\rm p})|_{[0,0]} \neq \emptyset$. Hence, $\mathfrak{B}_{\rm sup} \cap \mathfrak{B}_{\rm p} \neq \emptyset$. □

Whilst the trivial solution is unacceptable in almost any application context, it facilitates —very much in the spirit of [12, 18]— a set-theoretic lattice argument that establishes the unique existence of a *least restrictive* solution:

Theorem 2.7. (See also [8]) Let $(\mathfrak{B}_{\rm p},\ \mathfrak{B}_{\rm spec})$ be a supervisory control problem. The set of all solutions of $(\mathfrak{B}_{\rm p},\ \mathfrak{B}_{\rm spec})$ is a complete upper semi-lattice with the join operator "$\cup$" and the partial order "$\subseteq$". The supremal element $\mathfrak{B}_{\rm sup}^{\uparrow}$ of the lattice is referred to as the *least restrictive solution* of $(\mathfrak{B}_{\rm p},\ \mathfrak{B}_{\rm spec})$.

Proof. Given an arbitrary family of behaviours $(\mathfrak{B}_\alpha)_{\alpha\in A}$, $\mathfrak{B}_\alpha \subseteq W^{\mathbb{N}_0}$, $\alpha \in A$, let $\mathfrak{B}_\cup := \cup_{\alpha\in A}\mathfrak{B}_\alpha$. (i) If $\mathfrak{B}_{\rm p}$ and $\mathfrak{B}_\alpha$ are nonconflicting for each $\alpha \in A$ then so are $\mathfrak{B}_{\rm p}$ and $\mathfrak{B}_\cup$: pick any $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_{\rm p}|_{[0,k]} \cap \mathfrak{B}_\cup|_{[0,k]}$; then there exists an $\alpha \in A$ such that $w|_{[0,k]} \in \mathfrak{B}_{\rm p}|_{[0,k]} \cap \mathfrak{B}_\alpha|_{[0,k]}$ and hence $w|_{[0,k]} \in (\mathfrak{B}_{\rm p} \cap \mathfrak{B}_\alpha)|_{[0,k]} \subseteq (\mathfrak{B}_{\rm p} \cap \mathfrak{B}_\cup)|_{[0,k]}$. (ii) If $\mathfrak{B}_\alpha$ is generically implementable for each $\alpha \in A$

[5] The notion of *generic implementability* differs slightly from *implementability w.r.t. a particular plant* as defined in [8]. This adjustment leads to admissibility criteria that, as we will see in Proposition 3.2, are independent of particular plant dynamics. However, it can be shown that the two alternative notions of implementability lead to precisely the same closed-loop behaviours and, in this sense, the respective supervisory control problems are equivalent.

[6] The restriction operator $(\cdot)|_{[k_1,k_2]}\colon W^{\mathbb{N}_0} \to W^{k_2-k_1+1}$ is defined by $w|_{[k_1,k_2]} = (w(k_1),\ w(k_{k+1}),\ \ldots\ w(k_2))$ for all $k_1,\ k_2 \in \mathbb{N}_0$, $k_1 \leq k_2$, and all $w\colon \mathbb{N}_0 \to W$. For $k_1 > k_2$ let $w|_{[k_1,k_2]} := \epsilon$, $W^0 := \{\epsilon\}$, where ϵ denotes the empty string.

[7] We use $\tilde{w}|_{[0,k]} \approx_{\rm y} w|_{[0,k]}$ as an abbreviation for the two restricted trajectories to be identical up to the last output event, i.e. $\mathcal{P}_{\rm in}\tilde{w}|_{[0,k]} = \mathcal{P}_{\rm in}w|_{[0,k]}$ and $\mathcal{P}_{\rm out}\tilde{w}|_{[0,k-1]} = \mathcal{P}_{\rm out}w|_{[0,k-1]}$.

then so is $\mathfrak{B}_\cup$: pick any $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_\cup|_{[0,k]}$, $\tilde{w}|_{[0,k]} \approx_y w|_{[0,k]}$; then there exists $\alpha \in A$ such that $w|_{[0,k]} \in \mathfrak{B}_\alpha|_{[0,k]}$ and hence $\tilde{w}|_{[0,k]} \in \mathfrak{B}_\alpha|_{[0,k]} \subseteq \mathfrak{B}_\cup|_{[0,k]}$. (iii) Clearly, if $\mathfrak{B}_\alpha$ enforces the $\mathfrak{B}_{\text{spec}}$ for each $\alpha \in A$ then so does $\mathfrak{B}_\cup$. □

The least restrictive supervisor $\mathfrak{B}^\uparrow_{\text{sup}}$ contains all other solutions. In particular, $\mathfrak{B}^\uparrow_{\text{sup}}$ is a nontrivial solution if and only if a nontrivial solution exists.

3 Abstraction based synthesis

If both $\mathfrak{B}_{\text{p}}$ and $\mathfrak{B}_{\text{spec}}$ were realised by finite automata, a realisation of the least restrictive solution $\mathfrak{B}^\uparrow_{\text{sup}}$ to the problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$ could be computed with a slight modification of standard DES tools [12]. However, since hybrid plant behaviours $\mathfrak{B}_{\text{p}}$ almost never have a finite realisation, we instead work with an approximation $\mathfrak{B}_{\text{ca}}$ that is realised by a finite automaton. We say $\mathfrak{B}_{\text{ca}}$ is an *abstraction* of $\mathfrak{B}_{\text{p}}$ if $\mathfrak{B}_{\text{p}} \subseteq \mathfrak{B}_{\text{ca}}$. Under this condition, we will guarantee that solutions for $\mathfrak{B}_{\text{ca}}$ carry over to $\mathfrak{B}_{\text{p}}$. The following notion of *completeness* plays a key role in our discussion:

Definition 3.1. (See [17], Def. II.4) A behaviour $\mathfrak{B} \subseteq W^{\mathbb{N}_0}$ is *complete* if for all $w \in W^{\mathbb{N}_0}$:

$$w \in \mathfrak{B} \iff \forall\, k \in \mathbb{N}_0 :\ w\big|_{[0,k]} \in \mathfrak{B}\big|_{[0,k]}. \quad (1)$$

□

Behaviours that are realised by finite state machines are complete [5]. As another example, finite-dimensional discrete-time linear systems are seen to induce a complete external behaviour [17]. However, not all behaviours are complete and, in the case of our target configuration in Figure 1, completeness can not be determined by a finite computational procedure. For now, we use completeness as a convenient formal vehicle, subject to further justification.

Proposition 3.2. Let $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\text{in}} \times W_{\text{out}}$, be an I/- behaviour and let $\mathfrak{B}_{\text{sup}} \subseteq W^{\mathbb{N}_0}$ be generically implementable. If both $\mathfrak{B}_{\text{p}}$ and $\mathfrak{B}_{\text{sup}}$ are complete, then they are nonconflicting.

Proof. Pick any $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_{\text{p}}|_{[0,k]} \cap \mathfrak{B}_{\text{sup}}|_{[0,k]}$. Without loss of generality, we may assume $w \in \mathfrak{B}_{\text{sup}}$. Pick some $\tilde{w} \in \mathfrak{B}_{\text{p}}$ with $\tilde{w}|_{[0,k]} = w|_{[0,k]}$. Using the I/- property of $\mathfrak{B}_{\text{p}}$ and the generic implementability of $\mathfrak{B}_{\text{sup}}$, we can construct $\hat{w} \in W^{\mathbb{N}_0}$ with $\hat{w}|_{[0,k+1]} \in \mathfrak{B}_{\text{p}}|_{[0,k+1]} \cap \mathfrak{B}_{\text{sup}}|_{[0,k+1]}$ and $\hat{w}|_{[0,k]} = w|_{[0,k]}$. As we have started with an arbitrary $w|_{[0,k]} \in \mathfrak{B}_{\text{p}}|_{[0,k]} \cap \mathfrak{B}_{\text{sup}}|_{[0,k]}$, we can iterate our construction to obtain a sequence of trajectories $(w^\kappa)_{\kappa \in \mathbb{N}_0}$, $w^\kappa \in W^{\mathbb{N}_0}$, with $w^{\kappa+1}|_{[0,k+\kappa]} = w^\kappa|_{[0,k+\kappa]}$ for all $\kappa \in \mathbb{N}_0$. Hence, the sequence converges pointwise and we denote the limit by $w^\infty \in W^{\mathbb{N}_0}$; i.e. $w^\infty(j) := \lim_{\kappa\to\infty} w^\kappa(j)$, for all $j \in \mathbb{N}_0$. Then, $w^\infty|_{[0,\kappa]} \in \mathfrak{B}_{\text{p}}|_{[0,\kappa]} \cap \mathfrak{B}_{\text{sup}}|_{[0,\kappa]}$ for all $\kappa \in \mathbb{N}_0$. Completeness of $\mathfrak{B}_{\text{p}}$ and $\mathfrak{B}_{\text{sup}}$ then implies $w^\infty \in \mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}}$. Hence, $w|_{[0,k]} = w^\infty|_{[0,k]} \in (\mathfrak{B}_{\text{p}} \cap \mathfrak{B}_{\text{sup}})|_{[0,k]}$. □

From the above proposition, we derive our main result on abstraction based synthesis:

Theorem 3.3. Let $\mathfrak{B}_{\text{ca}} \subseteq W^{\mathbb{N}_0}$ be an abstraction of a plant $\mathfrak{B}_{\text{p}} \subseteq \mathfrak{B}_{\text{ca}}$ and let $\mathfrak{B}_{\text{sup}}$ be a complete and nontrivial solution of $(\mathfrak{B}_{\text{ca}}, \mathfrak{B}_{\text{spec}})$, $\mathfrak{B}_{\text{spec}} \subseteq W^{\mathbb{N}_0}$. If $\mathfrak{B}_{\text{p}}$ is a complete I/- behaviour, then $\mathfrak{B}_{\text{sup}}$ is a nontrivial solution of $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$.

Proof. Generic implementability does not depend on the particular plant, and, by Proposition 3.2, we obtain that $\mathfrak{B}_{\text{sup}}$ is admissible w.r.t. $\mathfrak{B}_{\text{p}}$. Clearly, $\mathfrak{B}_{\text{sup}}$ enforces the specification for $\mathfrak{B}_{\text{ca}}$, and, hence, $\mathfrak{B}_{\text{sup}}$ solves $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec}})$. Nontriviality is a consequence of Proposition 2.6. □

Recall that, in general, we cannot tell whether a hybrid system is complete. However, [6] identifies a general class of I/- behaviours which together with a generically implementable complete supervisor possess the nonconflicting property; this class of behaviours includes our target configuration depicted in Figure 1 and does not necessarily require plant completeness. Given a plant $\mathfrak{B}_{\text{p}}$ that possibly fails to be complete, let $\mathfrak{B}_{\text{ca}} := \{w \in W^{\mathbb{N}_0} \mid \forall\, k \in \mathbb{N}_0 : w|_{[0,k]} \in \mathfrak{B}_{\text{p}}|_{[0,k]}\} \supseteq \mathfrak{B}_{\text{p}}$ denote it's *completion*: it can be shown that $\mathfrak{B}_{\text{ca}}$ is the smallest complete superset of $\mathfrak{B}_{\text{p}}$; e.g. [16]. Appealing to the results in [6], we henceforth may assume that $\mathfrak{B}_{\text{p}}$ is complete, where, if need be, we substitute the original plant behaviour by its completion. One is then left to ensure that the candidate supervisor is complete. This clearly is the case for any supervisor that is realised by a finite state machine. Moreover, completeness of least restrictive supervisors is implied by the completeness of the specification, regardless of the plant

being complete or not:

Proposition 3.4. Let $\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec}} \subseteq W^{\mathbb{N}_0}$. If $\mathfrak{B}_{\mathrm{spec}} \subseteq W^{\mathbb{N}_0}$ is complete, then the least restrictive solution $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ of $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec}})$ is also complete.

Proof (Outline). Let $\mathfrak{B}' := \{w \in W^{\mathbb{N}_0} \mid \forall\, k \in \mathbb{N}_0 : w|_{[0,k]} \in \mathfrak{B}^{\uparrow}_{\mathrm{sup}}|_{[0,k]}\}$. Obviously, $\mathfrak{B}'$ is complete and $\mathfrak{B}^{\uparrow}_{\mathrm{sup}} \subseteq \mathfrak{B}'$. The key observation is that $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}|_{[0,k]} = \mathfrak{B}'|_{[0,k]}$ for all $k \in \mathbb{N}_0$, from which one can deduce that $\mathfrak{B}'$ is a solution of $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec}})$. As $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ is least restrictive, we conclude $\mathfrak{B}^{\uparrow}_{\mathrm{sup}} = \mathfrak{B}'$. □

4 Modular supervisory control

Setting up an overall supervisor by combining a number of individual supervisors is referred to as *modular supervisory control*. There are two potential benefits from modular supervisors. First, it may turn out that the synthesis of individual supervisors and their combination is computationally cheaper than the direct synthesis of an overall supervisor. Second, given a plant, one may set up a library of supervisors which can be combined in order to suit various applications for that plant. The modular control architecture we consider in this section is illustrated in Figure 2 and corresponds to the concept modularity that has been proposed for DESs; e.g. [2, 13, 18]. It is apparent from the picture that the hybrid character and the issue of modularity are well seperated. We therefor expect that basic principles from DES theory carry over to the behavioural framework and thus can be employed to establish modular controller synthesis for hybrid systems.

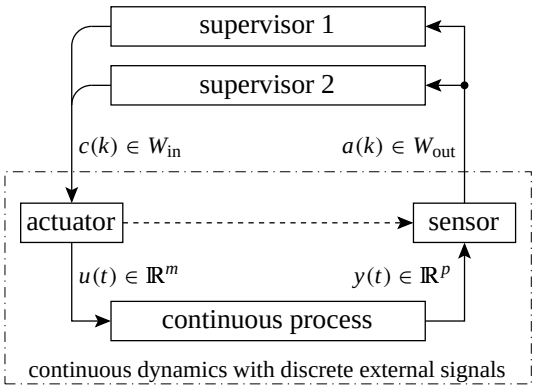

Figure 2: Modular control architecture

Given a plant $\mathfrak{B}_{\mathrm{p}}$, we assume that the problem of supervisory control has been solved for two specifications $\mathfrak{B}_{\mathrm{spec1}}$ and $\mathfrak{B}_{\mathrm{spec2}}$ individually. That is, two supervisors $\mathfrak{B}_{\mathrm{sup1}}$ and $\mathfrak{B}_{\mathrm{sup2}}$ have been established, both admissible w.r.t. the plant and —when connected individually— enforcing the desired specifications $\mathfrak{B}_{\mathrm{spec1}}$ and $\mathfrak{B}_{\mathrm{spec2}}$, respectively. Here, a sensible question to ask is under which circumstances and how these two supervisors can be combined into an overall supervisor $\mathfrak{B}_{\mathrm{sup}}$ such that both specifications are enforced simultaneously, i.e. the closed-loop behaviour $\mathfrak{B}_{\mathrm{cl}} = \mathfrak{B}_{\mathrm{p}} \cap \mathfrak{B}_{\mathrm{sup}}$ must be a subset of the intersection $\mathfrak{B}_{\mathrm{spec}} := \mathfrak{B}_{\mathrm{spec1}} \cap \mathfrak{B}_{\mathrm{spec2}}$. A natural starting point here is to run both supervisors $\mathfrak{B}_{\mathrm{sup1}}$ and $\mathfrak{B}_{\mathrm{sup2}}$ in parallel; i.e. we choose $\mathfrak{B}_{\mathrm{sup}} = \mathfrak{B}_{\mathrm{sup1}} \cap \mathfrak{B}_{\mathrm{sup2}}$ to be our candidate supervisor. Trivially, this $\mathfrak{B}_{\mathrm{sup}}$ enforces the combined specification $\mathfrak{B}_{\mathrm{spec}}$. However, while the property of admissibility for supervisors is preserved under union, the corresponding statement does not hold true for intersections. Thus, we ask for a criterion which guarantees our candidate $\mathfrak{B}_{\mathrm{sup}}$ to be admissible w.r.t. the plant $\mathfrak{B}_{\mathrm{p}}$. Furthermore, as on the realization level we intend to run $\mathfrak{B}_{\mathrm{sup1}}$ and $\mathfrak{B}_{\mathrm{sup2}}$ in parallel, we require that they do not conflict as long as the trajectory evolves within the plant behaviour. As we will show, the latter requirement will imply admissibility, and we start our discussion with a formal definition of the relative nonconflicting property.

Definition 4.1. (See [7], Sec. 4) Two behaviours $\mathfrak{B}_{\mathrm{sup1}} \in W^{\mathbb{N}_0}$ and $\mathfrak{B}_{\mathrm{sup2}} \in W^{\mathbb{N}_0}$ are *nonconflicting relative to* $\mathfrak{B}_{\mathrm{p}} \in W^{\mathbb{N}_0}$ if for all $k \in \mathbb{N}_0$:

$$\begin{aligned} & \mathfrak{B}_{\mathrm{p}}\big|_{[0,k]} \cap \mathfrak{B}_{\mathrm{sup1}}\big|_{[0,k]} \cap \mathfrak{B}_{\mathrm{sup2}}\big|_{[0,k]} \\ = \ & \mathfrak{B}_{\mathrm{p}}\big|_{[0,k]} \cap (\mathfrak{B}_{\mathrm{sup1}} \cap \mathfrak{B}_{\mathrm{sup2}})\big|_{[0,k]}\,. \end{aligned}$$

□

We expect that a least restrictive supervisor will "take no action outside the plant behaviour". The following proposition formalises this line of thought, and leads to a characterisation of the relative nonconflicting property for least restrictive supervisors.

Proposition 4.2. Let $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ denote the least restrictive solution of a supervisory control problem $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec}})$. Then for all $k \in \mathbb{N}_0$ and all $w \in W^{\mathbb{N}_0}$, $w|_{[0,k]} \in \mathfrak{B}^{\uparrow}_{\mathrm{sup}}|_{[0,k]} \setminus \mathfrak{B}_{\mathrm{p}}|_{[0,k]}$ implies $w \in \mathfrak{B}^{\uparrow}_{\mathrm{sup}}$.

Proof (Outline). Pick arbitrary $k \in \mathbb{N}_0$, $w|_{[0,k]} \in$

$\mathfrak{B}^{\uparrow}_{\mathrm{sup}}|_{[0,k]} \setminus \mathfrak{B}_{\mathrm{p}}|_{[0,k]}$. Let $\mathfrak{B}_{\mathrm{sup}} := \mathfrak{B}^{\uparrow}_{\mathrm{sup}} \cup \{v \in W^{\mathbb{N}_0} \mid v|_{[0,k]} = w|_{[0,k]}\}$. It can be seen that $\mathfrak{B}_{\mathrm{sup}}$ solves $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})$; further details are omitted for brevity. Clearly, $\mathfrak{B}_{\mathrm{sup}} \supseteq \mathfrak{B}^{\uparrow}_{\mathrm{sup}}$. As $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ is least restrictive, this implies $\mathfrak{B}_{\mathrm{sup}} = \mathfrak{B}^{\uparrow}_{\mathrm{sup}}$. From the definition of $\mathfrak{B}_{\mathrm{sup}}$ we obtain $w \in \mathfrak{B}_{\mathrm{sup}}$. □

We can now show that two least restrictive supervisors that have been designed for the same plant are relatively nonconflicting if and only if they are nonconflicting in the ordinary, "non-relative" sense:

Proposition 4.3. Let $\mathfrak{B}_{\mathrm{p}} \subseteq W^{\mathbb{N}_0}$ be a plant behaviour, and let $\mathfrak{B}_{\mathrm{spec1}}$, $\mathfrak{B}_{\mathrm{spec2}} \subseteq W^{\mathbb{N}_0}$ be two complete specifications. Denote the least restrictive solutions of the supervisory control problems $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec1}})$, $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec2}})$ by $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$, respectively. Then $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ are nonconflicting if and only if they are nonconflicting relative to $\mathfrak{B}_{\mathrm{p}}$.

Proof. Assume that $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ are nonconflicting relative to $\mathfrak{B}_{\mathrm{p}}$, and pick arbitrary $k \in \mathbb{N}_0$, $w \in |_{[0,k]} \in \mathfrak{B}^{\uparrow}_{\mathrm{sup1}}|_{[0,k]} \cap \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}|_{[0,k]}$. In the case of $w \in |_{[0,k]} \in \mathfrak{B}_{\mathrm{p}}|_{[0,k]}$, the relative nonconflicting property gives us $w|_{[0,k]} \in (\mathfrak{B}^{\uparrow}_{\mathrm{sup1}} \cap \mathfrak{B}^{\uparrow}_{\mathrm{sup2}})|_{[0,k]}$. In the case of $w|_{[0,k]} \notin \mathfrak{B}_{\mathrm{p}}|_{[0,k]}$, we assume without loss of generality that $w \in \mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and apply Proposition 4.2 to the supervisor $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$: $w|_{[0,k]} \in \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}|_{[0,k]} \setminus \mathfrak{B}_{\mathrm{p}}|_{[0,k]}$ gives us $w \in \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$. Thus, in both cases we have that $w|_{[0,k]} \in (\mathfrak{B}^{\uparrow}_{\mathrm{sup1}} \cap \mathfrak{B}^{\uparrow}_{\mathrm{sup2}})|_{[0,k]}$. Hence, $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ are nonconflicting. The converse implication follows immediately from Definition 4.1. □

Note that for a hybrid plant we can conduct this discussion on the abstraction level, where completeness of the substitute plant and the individual supervisors can be assumed. By the above proposition, our individual supervisors then must be nonconflicting if we want to apply the combined supervisor $\mathfrak{B}_{\mathrm{sup}} := \mathfrak{B}^{\uparrow}_{\mathrm{sup1}} \cap \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$. In fact, this condition also secures generic implementability of $\mathfrak{B}_{\mathrm{sup}}$:

Proposition 4.4. Let $\mathfrak{B}_{\mathrm{sup1}} \subseteq W^{\mathbb{N}_0}$, $\mathfrak{B}_{\mathrm{sup2}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\mathrm{in}} \times W_{\mathrm{out}}$ be generically implementable and nonconflicting. Then $\mathfrak{B}_{\mathrm{sup}} = \mathfrak{B}_{\mathrm{sup1}} \cap \mathfrak{B}_{\mathrm{sup2}}$ is generically implementable.

Proof. Pick arbitrary $k \in \mathbb{N}_0$, $w|_{[0,k]} \in \mathfrak{B}_{\mathrm{sup}}|_{[0,k]}$, $\tilde{w}|_{[0,k]} \in W^{k+1}$, $\tilde{w}|_{[0,k]} \approx_{\mathrm{y}} w|_{[0,k]}$. Then $w|_{[0,k]} \in \mathfrak{B}_{\mathrm{sup1}}|_{[0,k]} \cap \mathfrak{B}_{\mathrm{sup2}}|_{[0,k]}$, and we obtain from generic implementability of $\mathfrak{B}_{\mathrm{sup1}}$ and $\mathfrak{B}_{\mathrm{sup2}}$ that $\tilde{w}|_{[0,k]} \in \mathfrak{B}_{\mathrm{sup1}}|_{[0,k]} \cap \mathfrak{B}_{\mathrm{sup2}}|_{[0,k]}$. Then, nonconflictingness implies $\tilde{w}|_{[0,k]} \in (\mathfrak{B}_{\mathrm{sup1}} \cap \mathfrak{B}_{\mathrm{sup2}})|_{[0,k]}$. □

Admissibility is readily obtained for any complete I/- plant behaviour $\mathfrak{B}_{\mathrm{p}}$, since under this hypothesis $\mathfrak{B}_{\mathrm{p}}$ is nonconflicting in the interconnection with any complete generically implementable supervisor; see Proposition 3.2. We summarise the results of this section by the following corollary.

Corollar 4.5. Let $\mathfrak{B}_{\mathrm{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\mathrm{in}} \times W_{\mathrm{out}}$, be a complete I/- behaviour. Let $\mathfrak{B}_{\mathrm{spec1}}, \mathfrak{B}_{\mathrm{spec2}} \subseteq W^{\mathbb{N}_0}$ be complete. Denote the least restrictive solutions of $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec1}})$ and $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec2}})$ by $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ respectively. Then the following are equivalent:

(i) $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ are nonconflicting;
(ii) $\mathfrak{B}^{\uparrow}_{\mathrm{sup}} = \mathfrak{B}^{\uparrow}_{\mathrm{sup1}} \cap \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ is the least restrictive solution of $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec1}} \cap \mathfrak{B}_{\mathrm{spec2}})$, and $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ are nonconflicting relative to $\mathfrak{B}_{\mathrm{p}}$.

Proof. We show "(i) ⇒ (ii)". From Proposition 4.4, we obtain generic implementability of $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$. Proposition 3.4 ensures completeness of $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$. As completeness is retained under intersection, $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ is also complete. Thus, Proposition 3.2 provides admissibility of $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ w.r.t. $\mathfrak{B}_{\mathrm{p}}$. Clearly, $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ enforces $\mathfrak{B}_{\mathrm{spec}} := \mathfrak{B}_{\mathrm{spec1}} \cap \mathfrak{B}_{\mathrm{spec2}}$ and, hence, solves $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})$. Any solution $\mathfrak{B}_{\mathrm{sup}}$ of $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})$ is also a solution of $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec1}})$ and $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec2}})$. This implies $\mathfrak{B}_{\mathrm{sup}} \subseteq \mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$, $\mathfrak{B}_{\mathrm{sup}} \subseteq \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$, and, hence, $\mathfrak{B}_{\mathrm{sup}} \subseteq \mathfrak{B}^{\uparrow}_{\mathrm{sup}}$. Therefore $\mathfrak{B}^{\uparrow}_{\mathrm{sup}}$ is the least restrictive solution of $(\mathfrak{B}_{\mathrm{p}}, \mathfrak{B}_{\mathrm{spec}})$. The relative nonconflicting property in (ii) is seen by Proposition 4.3. The latter also gives "(ii) ⇒ (i)". □

Under the hypothesis of the above corollary, we may run the individual supervisors $\mathfrak{B}^{\uparrow}_{\mathrm{sup1}}$ and $\mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ in parallel on the plant $\mathfrak{B}_{\mathrm{p}}$ and by this achieve both specifications $\mathfrak{B}_{\mathrm{spec1}}$ and $\mathfrak{B}_{\mathrm{spec2}}$ simultaneously if and only if the individual supervisors are nonconflicting. Furthermore, if the latter condition is fulfilled and the combined supervisor $\mathfrak{B}^{\uparrow}_{\mathrm{sup}} := \mathfrak{B}^{\uparrow}_{\mathrm{sup1}} \cap \mathfrak{B}^{\uparrow}_{\mathrm{sup2}}$ imposes an empty closed-loop behaviour, we conclude that the trivial supervisor is the only solution that enforces both specifications.

5 Decentralised supervisory control

As in modular control, we are looking for multiple supervisors that enforce their individual specifications simultaneously. In contrast to the previous section, decentralised control addresses scenarios in which each controller has its own interface to the plant: the individual supervisors may not share the same measurement information nor may they have the same set of controls at their disposal. The principle aim is to address applications in which communication constraints enforce a decentralised control architecture, although computational benefits for controller synthesis also play a role in the DES literature on this topic; e.g. [4, 14, 19]. In this section, we address the hybrid decentralised control architecture shown in Figure 3. Motivated by the underlying continuous dynamics and by our results for the monolithic case we carefully discuss the input/output structure from the perspectives of the individual supervisors.

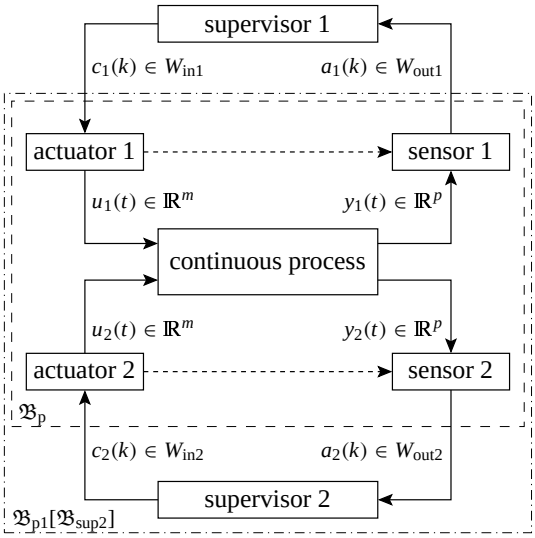

Figure 3: Decentralised control architecture

As in the previous sections, our plant model is an I/-behaviour $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\text{in}} \times W_{\text{out}}$. To represent the two separate interfaces, we assume that $W_{\text{in}} = W_{\text{in1}} \times W_{\text{in2}}$ and $W_{\text{out}} = W_{\text{out1}} \times W_{\text{out2}}$, where $W_1 = W_{\text{in1}} \times W_{\text{out1}}$ and $W_2 = W_{\text{in2}} \times W_{\text{out2}}$ are the external signal spaces of the respective interface/supervisor. Throughout this section we refer to this particular structure of W, including the proposed notation. For a time driven hybrid control system where the sampling instances are uniform for both interfaces, the hypothesis of $\mathfrak{B}_{\text{p}}$ to be an I/-behaviour is readily justified by the traditional notion of inputs and outputs of the underlying continuous dynamics. However, this is not the case for an event driven plant. In the latter case, one needs to distinguish the discrete time axis for each interface, as the k-th measurement event $a_2(k)$ form sensor 2, in general, occurs at a different instant of real time as $a_1(k)$. For simplicity —not only of notation— we restrict formal treatment to the time driven case.

Given two specifications $\mathfrak{B}_{\text{spec1}} \subseteq W_1^{\mathbb{N}_0}$, $\mathfrak{B}_{\text{spec2}} \subseteq W_2^{\mathbb{N}_0}$, we aim at the design of two supervisors $\mathfrak{B}_{\text{sup1}} \subseteq W_1^{\mathbb{N}_0}$ and $\mathfrak{B}_{\text{sup2}} \subseteq W_1^{\mathbb{N}_0}$ that, when connected to the plant via their interfaces, do enforce the respective specifications. Formally, we define the overall closed-loop behaviour by

$$\mathfrak{B}_{\text{cl}} := \mathfrak{B}_{\text{p}} \cap (\mathfrak{B}_{\text{sup1}} \times_\pi \mathfrak{B}_{\text{sup2}}) , \qquad (2)$$

where the product operator "$\times_\pi$" is the usual product composition plus the obvious re-arrangement of components to fit the scheme imposed by $W = W_{\text{in1}} \times W_{\text{in2}} \times W_{\text{out1}} \times W_{\text{out2}}$. [8] The pair of supervisors $(\mathfrak{B}_{\text{sup1}}, \mathfrak{B}_{\text{sup2}})$ enforces the specification given by $(\mathfrak{B}_{\text{spec1}}, \mathfrak{B}_{\text{spec2}})$ if the inclusion $\mathfrak{B}_{\text{cl}} \subseteq \mathfrak{B}_{\text{spec}} := \mathfrak{B}_{\text{spec1}} \times_\pi \mathfrak{B}_{\text{spec2}}$ is fulfilled. Note that by our product composition of specifications we cannot *explicitly* demand that the two supervisors cooperate. However, the supervisors interact via the plant, and our specification can *implicitly* enforce the supervisors to share common plant resources in a cooperative fashion to achieve the overall control objective.

In the following definition of solutions to the decentralised control problem we pragmatically copy the admissibility conditions from the ordinary supervisory control problem, subject to further justification.

Definition 5.1. The triple $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec1}}, \mathfrak{B}_{\text{spec2}})$ is said to be a *decentralised control problem,* where the I/- behaviour $\mathfrak{B}_{\text{p}} \subseteq W^{\mathbb{N}_0}$ is the plant and $\mathfrak{B}_{\text{spec1}} \subseteq W_1^{\mathbb{N}_0}$, $\mathfrak{B}_{\text{spec2}} \subseteq W_2^{\mathbb{N}_0}$, are the specifications for the respective signals. The pair of supervisors $(\mathfrak{B}_{\text{sup1}}, \mathfrak{B}_{\text{sup2}})$ is a *solution* of $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec1}}, \mathfrak{B}_{\text{spec2}})$ if $\mathfrak{B}_{\text{sup1}} \times_\pi \mathfrak{B}_{\text{sup2}}$ solves the supervisory control problem $(\mathfrak{B}_{\text{p}}, \mathfrak{B}_{\text{spec1}} \times_\pi \mathfrak{B}_{\text{spec2}})$. □

[8] Throughout this section, we use the subscript "π" to indicate the appropriate permutation.

The immediate benefit of the above definition is that one can draw from Theorem 3.3 for abstraction based approaches, and this includes the convenient assumption of completeness of $\mathfrak{B}_\mathrm{p}$ on the abstraction level. However, we need to justify our pragmatic admissibility conditions from the perspective of the individual supervisors. Are the individual supervisors of a solution in the sense of Definition 5.1 generically implementable? Is the plant from the perspective provided by each individual interface an I/- behaviour? We will obtain affirmative answers to both questions. The first one is straightforward:

Proposition 5.2. Let $\mathfrak{B}_{\mathrm{sup}1} \subseteq W_1$ and $\mathfrak{B}_{\mathrm{sup}2} \subseteq W_2^{\mathbb{N}_0}$ be non-empty supervisors, and let $\mathfrak{B}_{\mathrm{sup}} := \mathfrak{B}_{\mathrm{sup}1} \times_\pi \mathfrak{B}_{\mathrm{sup}2}$. Then $\mathfrak{B}_{\mathrm{sup}}$ is generically implementable if and only if both $\mathfrak{B}_{\mathrm{sup}1}$ and $\mathfrak{B}_{\mathrm{sup}2}$ are generically implementable.

Proof (Outline). We appeal to elementary properties of product composition and restriction operator. □

To answer the second question, we derive a plant model from the perspective of each individual supervisor, where symmetry allows us to restrict attention to the first supervisor. For the interface provided for $\mathfrak{B}_{\mathrm{sup}1}$, a plant model $\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}] \subseteq W_1^{\mathbb{N}_0}$ shall include all trajectories $w_1 \in W_1^{\mathbb{N}_0}$ which can possibly occur under the restrictions imposed by $\mathfrak{B}_{\mathrm{sup}2}$ on $\mathfrak{B}_\mathrm{p}$. Formally, we define

$$\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}] := \{w_1 \in W_1^{\mathbb{N}_0} \mid \exists\, w_2 \in \mathfrak{B}_{\mathrm{sup}2} : (w_1,\, w_2)_\pi \in \mathfrak{B}_\mathrm{p}\}\,. \quad (3)$$

The above formula is based on an intersection and a projection. In general, these operations are readily seen *not* to preserve the input/output structure of $\mathfrak{B}_\mathrm{p}$; e.g., if the behaviour $\mathfrak{B}_{\mathrm{sup}2}$ consists of one trajectory only, this imposes a rather restrictive condition on the measurement readings from sensor 2, which in turn restricts the input signals to $\mathfrak{B}_\mathrm{p}$, such that the input of $\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$ cannot be expected to be free. The following two propositions develop conditions for $\mathfrak{B}_{\mathrm{sup}2}$ that ensure that $\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$ is an I/- behaviour.

We first consider a simplified problem in which the behavioural restriction arises only from a fixed input signal rather than an entire behaviour $\mathfrak{B}_{\mathrm{sup}2}$.

Proposition 5.3. Let $\mathfrak{B}_\mathrm{p} \subseteq W^{\mathbb{N}_0}$ be an I/- behaviour (w.r.t. $W = W_\mathrm{in} \times W_\mathrm{out}$). For $c_2 \in W_{\mathrm{in}2}^{\mathbb{N}_0}$ define

$$\mathfrak{B}'_{\mathrm{p}1}[c_2] := \{(c_1,\, a_1,\, a_2) \in (W_{\mathrm{in}1} \times W_{\mathrm{out}1} \times W_{\mathrm{out}2})^{\mathbb{N}_0} \mid (c_1,\, c_2,\, a_1,\, a_2) \in \mathfrak{B}_\mathrm{p}\}\,. \quad (4)$$

Then $\mathfrak{B}'_{\mathrm{p}1}[c_2]$ is an I/- behaviour (w.r.t. $W_{\mathrm{in}1} \times W_\mathrm{out}$). If $\mathfrak{B}_\mathrm{p}$ is complete, so is $\mathfrak{B}'_{\mathrm{p}1}[c_2]$. By uniform substitution we obtain the corresponding definition and properties of $\mathfrak{B}'_{\mathrm{p}2}[c_1]$.

Proof. Clearly, the input of $\mathfrak{B}'_{\mathrm{p}1}[c_2]$ is free. To show that the output of $\mathfrak{B}'_{\mathrm{p}1}[c_2]$ does not anticipate the input, pick any $k \in \mathbb{N}_0$, $(\tilde{c}_1,\, \tilde{a}_1,\, \tilde{a}_2)$, $(\hat{c}_1,\, \hat{a}_1,\, \hat{a}_2) \in \mathfrak{B}'_{\mathrm{p}1}[c_2]$ with $\tilde{c}_1|_{[0,k]} = \hat{c}_1|_{[0,k]}$. Then $(\tilde{c}_1,\, c_2,\, \tilde{a}_1,\, \tilde{a}_2)$, $(\hat{c}_1,\, c_2,\, \hat{a}_1,\, \hat{a}_2) \in \mathfrak{B}_\mathrm{p}$. By hypothesis, $\mathfrak{B}_\mathrm{p}$ is an I/- behaviour. Hence, there exist $(a_1,\, a_2) \in W_\mathrm{out}^{\mathbb{N}_0}$ with $(a_1,\, a_2)|_{[0,k]} = (\hat{a}_1,\, \hat{a}_2)|_{[0,k]}$ and $(\tilde{c}_1,\, c_2,\, a_1,\, a_2) \in \mathfrak{B}_\mathrm{p}$. This implies $(\tilde{c}_1,\, a_1,\, a_2) \in \mathfrak{B}'_{\mathrm{p}1}[c_2]$, completing the proof of $\mathfrak{B}'_{\mathrm{p}1}[c_2]$ to be an I/- behaviour. We show that if $\mathfrak{B}_\mathrm{p}$ is complete so is $\mathfrak{B}'_{\mathrm{p}1}[c_2]$. Pick any $k \in \mathbb{N}_0$, $(c_1,\, a_1,\, a_2) \in (W_{\mathrm{in}1} \times W_\mathrm{out})^{\mathbb{N}_0}$, and observe that $(c_1,\, a_1,\, a_2)|_{[0,k]} \in \mathfrak{B}'_{\mathrm{p}1}[c_2]|_{[0,k]}$ is equivalent to $(c_1,\, c_2,\, a_1,\, a_2)|_{[0,k]} \in \mathfrak{B}_\mathrm{p}|_{[0,k]}$. Hence, completeness of $\mathfrak{B}_\mathrm{p}$ implies completeness of $\mathfrak{B}'_{\mathrm{p}1}[c_2]$. □

Proposition 5.4. Let $\mathfrak{B}_\mathrm{p} \subseteq W^{\mathbb{N}_0}$ be a complete I/- behaviour (w.r.t. $W = W_\mathrm{in} \times W_\mathrm{out}$) and let $\mathfrak{B}_{\mathrm{sup}2} \subseteq W_2^{\mathbb{N}_0}$, $\mathfrak{B}_{\mathrm{sup}2} \neq \emptyset$, be complete and generically implementable (w.r.t. $W_{\mathrm{in}2} \times W_{\mathrm{out}2}$). Then $\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$ is an I/- behaviour (w.r.t. $W_{\mathrm{in}1} \times W_{\mathrm{out}1}$)

Proof. For the scope of this proof, let $\mathfrak{B}'_{\mathrm{sup}2} := \mathfrak{B}_{\mathrm{sup}2} \times_\pi W_{\mathrm{out}1}^{\mathbb{N}_0} \subseteq (W_{\mathrm{in}2} \times W_{\mathrm{out}1} \times W_{\mathrm{out}2})^{\mathbb{N}_0}$. Observe the generic implementability and completeness of $\mathfrak{B}_{\mathrm{sup}2}$ carry over to $\mathfrak{B}'_{\mathrm{sup}2}$. We now show that the input of $\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$ is free. Pick an arbitrary $c_1 \in W_{\mathrm{in}1}^{\mathbb{N}_0}$ and focus attention of $\mathfrak{B}'_{\mathrm{p}2}[c_1]$. From Proposition 5.3 we obtain that $\mathfrak{B}'_{\mathrm{p}2}[c_1]$ is a complete I/- behaviour. Hence, by Proposition 2.6, we have $\mathfrak{B}'_{\mathrm{p}2}[c_1] \cap \mathfrak{B}'_{\mathrm{sup}2} \neq \emptyset$. Thus, there exists $(c_2,\, a_1,\, a_2)$ with $(c_2,\, a_2) \in \mathfrak{B}_{\mathrm{sup}2}$ and $(c_1,\, c_2,\, a_1,\, a_2) \in \mathfrak{B}_\mathrm{p}$. This implies $(c_1,\, a_1) \in \mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$, concluding the proof of the input being free. We show that for $\mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$ the input does not anticipate the output. Pick arbitrary $k \in \mathbb{N}_0$, $(\tilde{c}_1,\, \tilde{a}_1)$, $(\hat{c}_1,\, \hat{a}_1) \in \mathfrak{B}_{\mathrm{p}1}[\mathfrak{B}_{\mathrm{sup}2}]$ with $\tilde{c}_1|_{[0,k]} =$

$\hat{c}_1|_{[0,k]}$. Then there exist $(\tilde{c}_2,\ \tilde{a}_2) \in \mathfrak{B}_{\mathrm{sup2}}$ with $(\tilde{c}_1, \tilde{c}_2,\ \tilde{a}_1,\ \tilde{a}_2) \in \mathfrak{B}_{\mathrm{p}}$. As $\mathfrak{B}_{\mathrm{p}}$ is an I/- behaviour, we can find a_1 and a_2 such that $(\hat{c}_1, \tilde{c}_2,\ a_1,\ a_2) \in \mathfrak{B}_{\mathrm{p}}$ and $(a_1,\ a_2)|_{[0,k]} = (\tilde{a}_1,\ \tilde{a}_2)|_{[0,k]}$. Clearly, $(\tilde{c}_2,\ a_1,\ a_2)|_{[0,k]} \in \mathfrak{B}'_{\mathrm{sup2}}|_{[0,k]}$. Focus attention on $\mathfrak{B}'_{\mathrm{p2}}[\hat{c}_1]$ and observe that $(\tilde{c}_2,\ a_1,\ a_2)|_{[0,k]} \in \mathfrak{B}'_{\mathrm{p2}}[\hat{c}_1]|_{[0,k]}$. From Proposition 5.3 we obtain that $\mathfrak{B}'_{\mathrm{p2}}[\hat{c}_1]$ is a complete I/- behaviour. Together with Proposition 3.2 this implies that $\mathfrak{B}'_{\mathrm{p2}}[\hat{c}_1]$ and $\mathfrak{B}'_{\mathrm{sup2}}$ are nonconflicting, and, hence, $(\tilde{c}_2,\ a_1,\ a_2)|_{[0,k]} \in (\mathfrak{B}'_{\mathrm{p2}}[\hat{c}_1] \cap \mathfrak{B}'_{\mathrm{sup2}})|_{[0,k]}$. Therefore we can pick $(\tilde{c}'_2,\ a'_1,\ a'_2) \in \mathfrak{B}'_{\mathrm{p2}}[\hat{c}_1] \cap \mathfrak{B}'_{\mathrm{sup2}}$ with $(\tilde{c}'_2,\ a'_1,\ a'_2)|_{[0,k]} = (\tilde{c}_2,\ a_1,\ a_2)|_{[0,k]}$. In particular, we have $(\hat{c}_1,\ \tilde{c}'_2,\ a'_1,\ a'_2) \in \mathfrak{B}_{\mathrm{p}}$ and $(\tilde{c}'_2,\ a'_2) \in \mathfrak{B}_{\mathrm{sup2}}$, and, hence, $(\hat{c}_1,\ a'_1) \in \mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}_{\mathrm{sup2}}]$. Observe $a'_1|_{[0,k]} = a_1|_{[0,k]} = \tilde{a}_1|_{[0,k]}$, concluding the proof that $\mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}_{\mathrm{sup2}}]$ is nonanticipating. □

It is tempting to directly use synthesis procedures that address the ordinary supervisor control problem to solve the decentralised version. However, this will in general *not* result in a supervisor $\mathfrak{B}_{\mathrm{sup}}$ that can be decomposed in the required fashion $\mathfrak{B}_{\mathrm{sup}} = \mathfrak{B}_{\mathrm{sup1}} \times_\pi \mathfrak{B}_{\mathrm{sup2}}$. The following general property of solutions to a synthesis problem will play a key role in the deduction of a solution procedure for the decentralised case: roughly speaking, a supervisor is a subset of its specification if the specification only restricts the plant input.

Proposition 5.5. Let $\mathfrak{B}_{\mathrm{sup}}$ be a complete solution to the control problem $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec}})$. If $\mathfrak{B}_{\mathrm{p}} \subseteq W^{\mathbb{N}_0}$, $W = W_{\mathrm{in}} \times W_{\mathrm{out}}$, is a complete I/- behaviour and if $\mathfrak{B}_{\mathrm{spec}} = \mathcal{U} \times W_{\mathrm{out}}^{\mathbb{N}_0}, \mathcal{U} \subseteq W_{\mathrm{in}}^{\mathbb{N}_0}$, then $\mathfrak{B}_{\mathrm{sup}} \subseteq \mathfrak{B}_{\mathrm{spec}}$.

Proof. If $\mathfrak{B}_{\mathrm{sup}} = \emptyset$ then the claim is trivially true. For the case of $\mathfrak{B}_{\mathrm{sup}} \neq \emptyset$, we provide a proof by contradiction. Let $(c,\ a) \in \mathfrak{B}_{\mathrm{sup}} \setminus \mathfrak{B}_{\mathrm{spec}}$. Since the input of $\mathfrak{B}_{\mathrm{p}}$ is free and since $\mathfrak{B}_{\mathrm{sup}}$ is generically implementable, we can pick $a^0 \in W_{\mathrm{out}}^{\mathbb{N}_0}$ with $(c,\ a^0)|_{[0,0]} \in \mathfrak{B}_{\mathrm{p}}|_{[0,0]} \cap \mathfrak{B}_{\mathrm{sup}}|_{[0,0]}$. We inductively construct a sequence $(a^\kappa)_{\kappa \in \mathbb{N}_0}$ with

(i) $a^{\kappa+1}|_{[0,\kappa]} = a^\kappa|_{[0,\kappa]}$, and
(ii) $(c,\ a^\kappa)|_{[0,\kappa]} \in \mathfrak{B}_{\mathrm{p}}|_{[0,\kappa]} \cap \mathfrak{B}_{\mathrm{sup}}|_{[0,\kappa]}$,

for all κ. Suppose we are given a^κ with (i) and (ii). From (ii) we can pick $(\tilde{c}^{\kappa+1},\ \tilde{a}^{\kappa+1}) \in \mathfrak{B}_{\mathrm{p}}$ with $(\tilde{c}^{\kappa+1},\ \tilde{a}^{\kappa+1})|_{[0,\kappa]} = (c,\ a^\kappa)|_{[0,\kappa]}$. Using the I/- property of $\mathfrak{B}_{\mathrm{p}}$, we derive existence of an $a^{\kappa+1}$ with (i) and $(c,\ a^{\kappa+1}) \in \mathfrak{B}_{\mathrm{p}}$. From (ii) and generic implementability of $\mathfrak{B}_{\mathrm{sup}}$ we obtain $(c,\ a^{\kappa+1})|_{[0,\kappa+1]} \in \mathfrak{B}_{\mathrm{sup}}|_{[0,\kappa+1]}$, concluding the iterative construction. Property (i) ensures that the sequence converges pointwise to a limit $a^\infty \in W_{\mathrm{out}}^{\mathbb{N}_0}$. From (ii) and the completeness hypothesis, we obtain $(c,\ a^\infty) \in \mathfrak{B}_{\mathrm{p}} \cap \mathfrak{B}_{\mathrm{sup}}$. Clearly, $(c,\ a^\infty) \notin \mathfrak{B}_{\mathrm{spec}}$, and this contradicts with $\mathfrak{B}_{\mathrm{sup}}$ to solve $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec}})$. □

We conclude this section with the following theorem that provides a solution procedure to the decentralised control problem:

Theorem 5.6. Given a decentralised control problem $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec1}},\ \mathfrak{B}_{\mathrm{spec2}})$, where the plant $\mathfrak{B}_{\mathrm{p}}$ is a complete I/- behaviour, let:

(i) $\mathfrak{B}'_{\mathrm{spec2}} := \mathcal{P}_{\mathrm{in1}}\mathfrak{B}_{\mathrm{spec2}} \times W_{\mathrm{out2}}^{\mathbb{N}_0}$;
(ii) $\mathfrak{B}_{\mathrm{sup1}}$ solve $(\mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}'_{\mathrm{spec2}}],\ \mathfrak{B}_{\mathrm{spec1}})$;
(iii) $\mathfrak{B}_{\mathrm{sup2}}$ solve $(\mathfrak{B}_{\mathrm{p2}}[\mathfrak{B}_{\mathrm{sup1}}],\ \mathfrak{B}_{\mathrm{spec2}})$.

If both $\mathfrak{B}_{\mathrm{sup1}}$ and $\mathfrak{B}_{\mathrm{sup2}}$ are nontrivial and complete, and if $\mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}_{\mathrm{sup2}}]$ and $\mathfrak{B}_{\mathrm{p2}}[\mathfrak{B}_{\mathrm{sup1}}]$ are complete, then $\mathfrak{B}_{\mathrm{sup1}} \times_\pi \mathfrak{B}_{\mathrm{sup2}}$ solves $(\mathfrak{B}_{\mathrm{p}},\ \mathfrak{B}_{\mathrm{spec1}},\ \mathfrak{B}_{\mathrm{spec2}})$.

Proof. Clearly, $\mathfrak{B}_{\mathrm{spec2}} \subseteq \mathfrak{B}'_{\mathrm{spec2}}$. Hence, $\mathfrak{B}_{\mathrm{sup2}}$ solves $(\mathfrak{B}_{\mathrm{p2}}[\mathfrak{B}_{\mathrm{sup1}}],\ \mathfrak{B}'_{\mathrm{spec2}})$. We apply Proposition 5.5 to the control problem $(\mathfrak{B}_{\mathrm{p2}}[\mathfrak{B}_{\mathrm{sup1}}],\ \mathfrak{B}'_{\mathrm{spec2}})$ and obtain $\mathfrak{B}_{\mathrm{sup2}} \subseteq \mathfrak{B}'_{\mathrm{spec2}}$. Hence, $\mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}_{\mathrm{sup2}}] \subseteq \mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}'_{\mathrm{spec2}}]$. Using Proposition 5.4 and Theorem 3.3, this implies that $\mathfrak{B}_{\mathrm{sup1}}$ solves $(\mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}_{\mathrm{sup2}}],\ \mathfrak{B}_{\mathrm{spec1}})$. Now pick any overall closed-loop trajectory $w = (w_1,\ w_2)_\pi \in \mathfrak{B}_{\mathrm{p}} \cap (\mathfrak{B}_{\mathrm{sup1}} \times_\pi \mathfrak{B}_{\mathrm{sup2}})$ and observe $w_1 \in \mathfrak{B}_{\mathrm{p1}}[\mathfrak{B}_{\mathrm{sup2}}] \cap \mathfrak{B}_{\mathrm{sup1}} \subseteq \mathfrak{B}_{\mathrm{spec1}}$ and $w_2 \in \mathfrak{B}_{\mathrm{p2}}[\mathfrak{B}_{\mathrm{sup1}}] \cap \mathfrak{B}_{\mathrm{sup2}} \subseteq \mathfrak{B}_{\mathrm{spec2}}$. Hence, $w \in \mathfrak{B}_{\mathrm{spec1}} \times_\pi \mathfrak{B}_{\mathrm{spec2}}$. This shows that $\mathfrak{B}_{\mathrm{sup}}$ enforces the specification. Since completeness is retained under the cross product, we obtain from Proposition 5.2 admissibility of $\mathfrak{B}_{\mathrm{sup}}$ w.r.t. $\mathfrak{B}_{\mathrm{p}}$. □

6 Conclusions

We have addressed the problem of supervisory controller synthesis for continuous plants with a discrete event interface, using results from earlier work in which we have shown that this class of problems can be treated by a plant abstraction step and an adaption

of known results from DES theory. The distinguishing feature of our setting is an input/output structure based on product composition of the respective components. This is the normal case for continuous control systems and, hence, it is well motivated by the considered class of plant models. The present paper continues this line of thought in a discussion of modular and decentralised supervision of continuous systems. Our results demonstrate that not only the most fundamental DES techniques can be put to use in a hybrid systems context, but also that more advanced principles can provide a rich source of inspiration for discrete control of continuous systems.

Acknowledgements. This work has benefited from discussions with J.M. Davoren and B.D.O. Anderson (RSISE, Australian National University).

References

[1] R. Alur, C. Courcoubetis, T.A. Henzinger, and P.-H. Ho. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. In R.L. Grossman, A. Nerode, A.P. Ravn, and H. Rischel, editors, *Hybrid Systems IV*, LNCS 736, pages 209–229. Springer-Verlag, 1993.

[2] Y.-L. Cheng, S. Lafortune, and F. Lin. Incremental model evolution and reusability of supervisors for discrete event systems. *Automatica*, 36:243–259, 2000.

[3] X. Koutsoukos, P.J. Antsaklis, J.A. Stiver, and M.D. Lemmon. Supervisory control of hybrid systems. *Proceedings of the IEEE*, 88:1026–1049, July 2000.

[4] S.-H. Lee and K.C. Wong. Decentralised control of concurrent discrete-event systems with non-prefix closed local specification. In *IEEE Proc. of the 36th International Conference on Decision and Control*, pages 2958–2963, 1997.

[5] T. Moor. *Approximationsbasierter Entwurf diskreter Steuerungen für gemischtwertige Regelstrecken*, volume 2 of *Forschungsberichte aus dem Max-Planck-Institut für Dynamik komplexer technischer Systeme*. Shaker-Verlag, Aachen, Germany, 2000. Also PhD thesis, Fachbereich Elektrotechnik, Universität der Bundeswehr Hamburg.

[6] T. Moor, J.M. Davoren, and B.D.O. Anderson. Robust hybrid control from a behavioural perspective. Technical report, RSISE, Australian National University, 2002. Submitted for publication.

[7] T. Moor, J.M. Davoren, and J. Raisch. Modular supervisory control of a class of hybrid systems in a behavioural framework. In *Proceedings of the European Control Conference 2001*, pages 870–875, Porto, Portugal, 2001.

[8] T. Moor and J. Raisch. Supervisory control of hybrid systems within a behavioural framework. *Systems and Control Letters*, 38:157–166, 1999.

[9] T. Moor, J. Raisch, and J.M. Davoren. Computational advantages of a two-level hybrid control architecture. In *IEEE Proc. of the 40th International Conference on Decision and Control*, pages 358–362, 2001.

[10] T. Moor, J. Raisch, and S.D. O'Young. Discrete supervisory control of hybrid systems based on l-complete approximations. *Journal of Discrete Event Dynamic Systems*, 12:83–107, 2002.

[11] J. Raisch and S.D. O'Young. Discrete approximation and supervisory control of continuous systems. *IEEE Transactions on Automatic Control, Special issue on hybrid systems*, 43:569–573, 1998.

[12] P.J. Ramadge and W.M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77:81–98, 1989.

[13] P.J. Ramadge and W.M. Wonham. Modular control of discrete event systems. *Maths. of Control, Signals & Systems*, 1:1:13–30, 1989.

[14] K. Rudie and W.M. Wonham. Think globally, act locally: decentralized supervisory control. *IEEE Transactions on Automatic Control*, 37:11:1692–1708, 1992.

[15] J. A. Stiver, P. J. Antsaklis, and M. D. Lemmon. Interface and controller design for hybrid systems. In P.J. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, editors, *Hybrid Systems II*, LNCS 999, pages 462–492. Springer-Verlag, 1995.

[16] J.C. Willems. Models for dynamics. *Dynamics Reported*, 2:172–269, 1989.

[17] J.C. Willems. Paradigms and puzzles in the theory of dynamic systems. *IEEE Transactions on Automatic Control*, 36:258–294, 1991.

[18] W.M. Wonham. Notes on control of discrete event systems. Technical report, Department of Electrical & Computer Engineering, University of Toronto, 1999.

[19] T. Yoo and S. Lafortune. New results on decentralized supervisory control of discrete-event systems. In *IEEE Proc. of the 39th International Conference on Decision and Control*, pages 1–6, 2000.