A Hierarchical and Modular Control Architecture for Sequential Behaviours

Christine Baier · Thomas Moor

Abstract This paper develops a hierarchical and modular control architecture for so called *sequential behaviours*, i.e. for plant dynamics and specifications that are represented as formal languages of infinite-length words. Our main result is the elaboration of structural properties that allow for abstraction based controller design and that are preserved under closed-loop composition. Thus, we propose to alternate controller design, closed-loop composition and abstraction in order to construct a hierarchical control system in a bottom-up fashion. When the overall plant is composed from a number of components, our approach naturally extends to the alternation of controller design, closed-loop composition, abstraction and component composition. Technically, our results are based on the notion of input-output systems known from *behavioural systems theory*, with a particular focus on *liveness properties* represented as sequential behaviours that are not necessarily *topologically closed*.

Keywords discrete-event systems · ω-languages · sequential behaviours · hierarchical control systems · modular control systems

Christine Baier · Thomas Moor
Lehrstuhl für Regelungstechnik
Friedrich-Alexander Universität Erlangen-Nürnberg
Cauerstr. 7, 91058 Erlangen, Germany
Tel.: +49-9131-8527130
Fax: +49-9131-8528715
E-mail: lrt@fau.de

Introduction

It is common engineering practice to address complex control problems by a hierarchical system design. In the context of supervisory control (Ramadge and Wonham, 1989), this principle has been formalized from a variety of perspectives, see e.g. (Wong and Wonham, 1996; Feng and Wonham, 2008; da Cunha et al, 2002; Leduc et al, 2005; Schmidt et al, 2008). In contrast to monolithic approaches, the benefits include a systematic derivation of adequate models for the individual levels of abstraction, as well as computational feasibility for large scale systems.

In this paper, we further develop a hierarchical control architecture that was originally introduced by Moor et al (2003) to address a class of hybrid systems, and subsequently discussed by Perk et al (2006) for discrete-event systems. For each individual level of the hierarchy, it is proposed to design a controller according to a language inclusion specification. When only *safety properties* are of concern, all relevant languages can be considered *prefix-closed*, and controller synthesis can be based on an abstraction of the levels below, i.e., on a superset language that accounts for any trajectory the lower levels may evolve on. In particular, the specification used for one level in the hierarchy can be utilized as an abstraction for the synthesis of the controller one level above. Furthermore, when the plant is composed from a number of components, one may first design individual low-level controllers that address local control objectives per plant component, and then compose the individual specifications to obtain an abstraction for the design of a high-level supervisor. Technically, as long as all relevant languages are considered prefix-closed, the validity of this approach is established by a fairly simple set-inclusion argument. Computational benefits are expected from alternating abstraction and controller synthesis, since the specification does not need to express how the control objective is achieved. This has been demonstrated by a transport system example in Perk et al (2008).

Regarding *liveness properties*, the situation is more involved than for safety properties. Depending on the particular approach (Wong and Wonham, 1996; Feng and Wonham, 2008; da Cunha et al, 2002; Leduc et al, 2005; Schmidt et al, 2008), additional conditions are imposed on the plant and/or the abstraction in order to retain liveness properties in a hierarchical design. Here, Perk et al (2006) refer to a variation of *input-output systems* proposed by Willems (1991) in order to establish the following structural liveness property of the overall closed-loop configuration: any finite string on which the overall system evolves (a) can be extended by one more event and (b) when extended to infinite length must include infinitely many high-level events. The former appears to be a natural choice for the hybrid systems discussed in Moor et al (2003), whereas the latter guarantees for persistent feedback through all levels of the hierarchy of controllers. However, in the context of discrete-event systems, more general liveness properties can be expressed by languages of infinite-length words, also known as *sequential behaviours* or *ω-languages*. Examples of liveness properties include eventual task completion or eventual feedback to a server of any individual one of a number of clients. The corresponding ω-languages share the technical property that they are not *topologically closed*; for a detailed classification of liveness properties in the context of temporal logics, see (Baier and Kwiatkowska, 2000; Manna and Pnueli, 1990). While the literature on supervisory control commonly represents relevant dynamics as $*$-languages, the core results are also available for ω-languages (Ramadge, 1989; Kumar et al, 1992; Thistle and Wonham, 1994), including abstraction-based controller design (Moor et al, 2011).

This paper is an extended revision of the conference contribution (Baier and Moor, 2012), where we introduce a hierarchical control architecture for not necessarily topologically closed ω-languages. In addition to the results in (Baier and Moor, 2012), we explicitly address modularity, i.e., plants that are composed from individual components, and we include an example to further illustrate our approach. Conceptually, we follow the discussion presented in (Moor et al, 2003; Perk et al, 2006). However, while Moor et al (2003) formally use ω-languages to represent behaviours, they effectively require plant and controller to be topologically closed, and, thus, exclude general liveness properties. Similarly, Perk et al (2006) model system components by prefix-closed $*$-languages, which exhibit a topologically closed limit, and, thus, can not represent general liveness properties. In this contribution, we refer to a notion of *non-anticipating input-output systems* to achieve a non-conflicting closed-loop configuration, but we impose no further constraints regarding liveness properties possessed by the plant or required by the specification.

The paper is organized as follows. Section 1 introduces notation, recalls well-known facts regarding formal languages, and gives a concise summary on the supervision of ω-languages. Section 2 discusses a closed-loop configuration with external signals and characterizes admissible controllers in terms of achievable closed-loop behaviours. Section 3 shows that relevant plant properties are retained under closed-loop composition, and, thereby, establishes a hierarchical control architecture. Section 4 elaborates a composition method for modular plants. An example is provided in Section 5 in order to illustrate the formal results in an application context. A number of technical lemmata have been moved to the Appendix.

1 Preliminaries and notation

1.1 $*$-languages

For an *alphabet* Σ, the *Kleene-closure* Σ^* is defined as a set of finite strings $s = \sigma_1\sigma_2\cdots\sigma_n$, $n \in \mathbb{N}$, $\sigma_i \in \Sigma$, including the *empty string* $\epsilon \in \Sigma^*$, $\epsilon \notin \Sigma$. The length of a string $s \in \Sigma^*$ is denoted $|s| \in \mathbb{N}$. A *$*$-language* (or short *language*) over Σ is a set $L \subseteq \Sigma^*$ of strings. A language $L \subseteq \Sigma^*$ is *bounded* if $\sup\{|s| \mid s \in L\} < \infty$, or, else, L is *unbounded*. An introduction to $*$-languages from a supervisory control perspective is included in e.g. (Cassandras and Lafortune, 2008).

If for two strings $s, r \in \Sigma^*$ there exists $t \in \Sigma^*$ such that $s = rt$, we say r is a *prefix* of s, and write $r \leq s$. If in addition $r \neq s$, we say r is a *strict prefix* of s and write $r < s$. The *prefix-closure* (or short *closure*) of $L \subseteq \Sigma^*$ is defined by $\operatorname{pre} L := \{r \mid \exists\, s \in L : r \leq s\} \subseteq \Sigma^*$. A language $L \subseteq \Sigma^*$ is called *closed*, if $L = \operatorname{pre} L$. Given $L,\ K \subseteq \Sigma^*$, we say K is *relatively closed w.r.t.* L if $K = (\operatorname{pre} K) \cap L$. If a language K is relatively closed w.r.t. a closed language, then K itself is closed. The closure operator distributes over arbitrary unions of languages. However, for the intersection of two languages $L,\ K \subseteq \Sigma^*$, we have $\operatorname{pre}(L \cap K) \subseteq (\operatorname{pre} L) \cap (\operatorname{pre} K)$, and, if equality holds, L and K are said to be *non-conflicting*. This is trivially the case for $K \subseteq L$.

The *natural projection* $\mathrm{p_o} : \Sigma^* \to \Sigma_\mathrm{o}^*$, $\Sigma_\mathrm{o} \subseteq \Sigma$, is defined iteratively: let $\mathrm{p_o}\epsilon := \epsilon$; and, for $s \in \Sigma^*$, $\sigma \in \Sigma$, if $\sigma \in \Sigma_\mathrm{o}$, let $\mathrm{p_o}(s\sigma) := (\mathrm{p_o}s)\,\sigma$, or else let $\mathrm{p_o}(s\sigma) = \mathrm{p_o}s$. The set-valued inverse $\mathrm{p_o^{-1}}$ is defined by $\mathrm{p_o^{-1}}(r) := \{s \in \Sigma^* \mid \mathrm{p_o}(s) = r\}$ for $r \in \Sigma_\mathrm{o}^*$. When applied to languages, the projection distributes over arbitrary unions, and the inverse projection distributes over arbitrary unions and arbitrary intersections. Furthermore, the closure operator commutes with projection and inverse projection.

Given $L,\ K \subseteq \Sigma^*$, and a set of uncontrollable events $\Sigma_\mathrm{uc} \subseteq \Sigma$, we say K is *controllable w.r.t.* $(\Sigma_\mathrm{uc},\ L)$, if $((\mathrm{pre}\,K)\Sigma_\mathrm{uc}) \cap (\mathrm{pre}\,L) \subseteq \mathrm{pre}\,K$; see (Ramadge and Wonham, 1987). Given $L,\ K \subseteq \Sigma^*$, and a set of observable events $\Sigma_\mathrm{o} \subseteq \Sigma$, we say K is *prefix-normal w.r.t.* $(\Sigma_\mathrm{o},\ L)$, if $\mathrm{pre}\,K = (\mathrm{p_o^{-1}}\mathrm{p_o}\,\mathrm{pre}\,K) \cap (\mathrm{pre}\,L)$; see (Lin and Wonham, 1988). A language $K \subseteq \Sigma^*$ is said to be *complete*, if for all $s \in \mathrm{pre}\,K$ there exists $\sigma \in \Sigma$ such that $s\sigma \in \mathrm{pre}\,K$; see e.g. (Kumar et al, 1992). Controllability, prefix-normality, completeness, prefix-closedness and relative closedness are retained under arbitrary union. In particular, given a language $E \subseteq \Sigma^*$, the supremal sublanguage that possesses any conjunction of the above properties exists uniquely.

A $*$-language $L \subseteq \Sigma^*$ is *regular*, if it is marked by a *finite automaton*; see e.g. (Hopcroft and Ullman, 1979). Operations on $*$-languages relevant to this paper retain regularity, with known algorithms to obtain the respective finite state realisation.

1.2 ω-languages

An *infinite string* over Σ is defined as a function $w : \mathbb{N} \to \Sigma$. By $\Sigma^\omega := \{w \mid w : \mathbb{N} \to \Sigma\}$ we denote the set of all infinite strings over Σ. An *ω-language* is a subset $\mathcal{L} \subseteq \Sigma^\omega$. For a general discussion of ω-languages, see e.g. (Mukund, 1996; Thomas, 1990). Throughout this paper, we follow the convention to denote ω-languages by calligraphic letters.

Given $w \in \Sigma^\omega$, the *prefix with length* $n \in \mathbb{N}$ is denoted $w^{(n)} \in \Sigma^*$ and we write $s < w$ for a prefix $s \in \Sigma^*$ of w. The *prefix* of an ω-language $\mathcal{L} \subseteq \Sigma^\omega$ is defined by $\mathrm{pre}\,\mathcal{L} := \{s \in \Sigma^* \mid \exists\, w \in \mathcal{L} : s < w\} \subseteq \Sigma^*$. The prefix of any ω-language is a complete prefix-closed $*$-language. The prefix operator distributes over arbitrary unions of ω-languages. However, for the intersection of two ω-languages $\mathcal{L}, \mathcal{K} \subseteq \Sigma^\omega$, we have $\mathrm{pre}(\mathcal{L} \cap \mathcal{K}) \subseteq (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{K})$, and, if equality holds, the languages are said to be *non-conflicting*. The languages $\mathcal{L},\ \mathcal{K} \subseteq \Sigma^\omega$ are *locally non-conflicting* if $(\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{K})$ is complete. If two languages are non-conflicting, they are also locally non-conflicting. Note, that for $\mathcal{K} \subseteq \mathcal{L}$ both conditions are trivially satisfied.

A *monotone sequence of strings*, denoted by $(s_n) \subseteq \Sigma^*$, is a sequence $(s_n)_{n\in\mathbb{N}}$, $s_n \in \Sigma^*$, $s_n \leq s_{n+1}$ for all $n \in \mathbb{N}$. The sequence (s_n) is *bounded* if $\sup\{|s_n| \mid n \in \mathbb{N}\} < \infty$, or, else, (s_n) is *unbounded*. The point-wise *limit* of a monotone sequence (s_n) is denoted by $\lim(s_n) \in \Sigma^* \cup \Sigma^\omega$. For a language $L \subseteq \Sigma^*$, the *limit* is defined by $\lim L := \{\lim(s_n) \mid (s_n) \subseteq L\} \cap \Sigma^\omega$. Note that $\mathrm{pre}\lim L = L$, if and only if L is complete and prefix-closed; see (Kumar et al, 1992). Hence, $\mathrm{pre}\lim\mathrm{pre}\,\mathcal{L} = \mathrm{pre}\,\mathcal{L}$. For the intersection of two $*$-languages $L,\ K \subseteq \Sigma^*$, with $K = \mathrm{pre}\,K$, it is $\lim(L \cap K) = (\lim L) \cap (\lim K)$; see (Baier and Moor, 2012), Lemma 1.

The *topological closure* (or short *closure*) of an ω-language $\mathcal{L} \subseteq \Sigma^\omega$ is defined by $\mathrm{clo}\,\mathcal{L} := \lim \mathrm{pre}\,\mathcal{L}$. An ω-language is said to be *topologically closed* (or short *closed*) if $\mathrm{clo}\,\mathcal{L} = \mathcal{L}$. The limit of a prefix-closed $*$-language is topologically closed. Given two ω-languages $\mathcal{L},\ \mathcal{K} \subseteq \Sigma^\omega$, we say, that $\mathcal{K}$ is *relatively closed w.r.t.* $\mathcal{L}$, if $\mathcal{K} = (\mathrm{clo}\,\mathcal{K}) \cap \mathcal{L}$. The closure operator distributes over finite unions of ω-languages, see e.g. Ramadge (1989).

For the *natural projection of infinite strings*, let $w \in \Sigma^\omega$, denote $(s_n) \subseteq \Sigma^*$ an unbounded monotone sequence of prefixes of w, and define $\mathrm{p}_\mathrm{o}^\omega w := \lim(\mathrm{p}_\mathrm{o} s_n) \in \Sigma_\mathrm{o}^* \cup \Sigma_\mathrm{o}^\omega$; see also (Kumar et al, 1992). The set-valued inverse is given by $\mathrm{p}_\mathrm{o}^{-\omega}(v) := \{\, w \in \Sigma^\omega \mid \mathrm{p}_\mathrm{o}^\omega(w) = v \,\}$ for $v \in \Sigma_\mathrm{o}^* \cup \Sigma_\mathrm{o}^\omega$. When applying the projection to languages, we obtain $\mathrm{p}_\mathrm{o}^\omega \mathcal{L} = \{\, \mathrm{p}_\mathrm{o}^\omega w \mid w \in \mathcal{L} \,\} \subseteq \Sigma_\mathrm{o}^* \cup \Sigma_\mathrm{o}^\omega$ for $\mathcal{L} \subseteq \Sigma^\omega$, and $\mathrm{p}_\mathrm{o}^{-\omega} \mathcal{L}_\mathrm{o} = \{\, w \in \Sigma^\omega \mid \mathrm{p}_\mathrm{o}^\omega w \in \mathcal{L}_\mathrm{o} \,\}$ for $\mathcal{L}_\mathrm{o} \subseteq \Sigma_\mathrm{o}^* \cup \Sigma_\mathrm{o}^\omega$. Here, the projection distributes over arbitrary unions, the inverse projection over arbitrary unions and arbitrary intersections. Both commute with the prefix operator. Note also that $\mathrm{p}_\mathrm{o}^{-\omega}\mathrm{p}_\mathrm{o}^\omega \mathcal{L} = \{\, w \in \Sigma^\omega \mid \exists\, w' \in \mathcal{L} : \mathrm{p}_\mathrm{o}^\omega w = \mathrm{p}_\mathrm{o}^\omega w' \,\}$. From (Baier and Moor, 2012), Lemma 2, we recall the following relationship between limit and projection: given Σ, $\Sigma_\mathrm{o} \subseteq \Sigma$, and $L = \mathrm{pre}\, L \subseteq \Sigma^*$, $L_\mathrm{o} \subseteq \Sigma_\mathrm{o}^*$, $\mathcal{L}_\mathrm{o} \subseteq \Sigma_\mathrm{o}^\omega$, then

(i) $(\mathrm{p}_\mathrm{o}^\omega \lim L) \cap \Sigma_\mathrm{o}^\omega = \lim \mathrm{p}_\mathrm{o} L$,
(ii) $\mathrm{p}_\mathrm{o}^{-\omega} \lim L_\mathrm{o} = (\lim \mathrm{p}_\mathrm{o}^{-1} L_\mathrm{o}) \cap (\mathrm{p}_\mathrm{o}^{-\omega} \Sigma_\mathrm{o}^\omega)$,
(iii) $\mathrm{clo}\,\mathrm{p}_\mathrm{o}^{-\omega} \mathcal{L}_\mathrm{o} = (\mathrm{p}_\mathrm{o}^{-\omega}\, \mathrm{clo}\, \mathcal{L}_\mathrm{o}) \cup (\mathrm{p}_\mathrm{o}^{-\omega}\, \mathrm{pre}\, \mathcal{L}_\mathrm{o})$.

Given $\mathcal{L},\ \mathcal{K} \subseteq \Sigma^\omega$, we say that $\mathcal{K}$ is *normal w.r.t.* $(\Sigma_\mathrm{o},\ \mathcal{L})$, if $\mathcal{K} = (\mathrm{p}_\mathrm{o}^{-\omega}\mathrm{p}_\mathrm{o}^\omega \mathcal{K}) \cap \mathcal{L}$. Normality is retained under arbitrary union. Provided that $\mathcal{K}$ is relatively closed w.r.t. $\mathcal{L}$, prefix-normality of $\mathrm{pre}\,\mathcal{K}$ w.r.t. $(\Sigma_\mathrm{o},\ \mathrm{pre}\,\mathcal{L})$ implies normality of $\mathcal{K}$ w.r.t. $(\Sigma_\mathrm{o}, \mathcal{L})$; see (Baier and Moor, 2012), Lemma 3.

An ω-language $\mathcal{L} \subseteq \Sigma^\omega$ is *regular*, if it is accepted by a *Büchi automaton*, i.e., a finite automaton that accepts all those infinite-length executions that infinitely often pass marked states; see e.g. (Thomas, 1990). Moreover, if an ω-language can be represented as the limit of a regular $*$-language, it is accepted by a *deterministic* Büchi automaton, and, hence, it is regular. The main results developed in this paper do neither rely on regularity nor on a particular form of finite state realisation. However, when referring to computational procedures we pragmatically focus attention on deterministic Büchi automata or, equivalently, on limits of regular $*$-languages; e.g., to address the natural projection, algorithms can be obtained via the above relations (i) and (ii); see also (Mukund, 1996).

1.3 ω-controllability

We recall the definition[1] of *ω-controllability* from (Thistle and Wonham, 1994) and discuss how it relates to earlier work (Ramadge, 1989).

[1] Literally, Definition 1 differs from the referenced literature, as we use slightly different notation to facilitate the subsequent extension to the situation of partial observation and to formally account for the case $\mathcal{H} \not\subseteq \mathcal{L}$ in the context of abstraction-based control. However, for $\mathcal{H} \subseteq \mathcal{L}$, our variation is technically equivalent to the corresponding definition in (Thistle and Wonham, 1994).

Definition 1 Let $\Sigma_{\mathrm{uc}} \subseteq \Sigma$ and consider the *plant* $\mathcal{L} \subseteq \Sigma^{\omega}$ and the *controller* $\mathcal{H} \subseteq \Sigma^{\omega}$. Then, $\mathcal{H}$ is said to be *ω-controllable* w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathcal{L})$, if for all $s \in (\operatorname{pre}\mathcal{L}) \cap (\operatorname{pre}\mathcal{H})$ there exists a $\mathcal{V}_s \subseteq \mathcal{L} \cap \mathcal{H}$ with $s \in \operatorname{pre}\mathcal{V}_s$, and

(i) $\operatorname{pre}\mathcal{V}_s$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \operatorname{pre}\mathcal{L})$, and
(ii) $\mathcal{V}_s$ is relatively topologically closed w.r.t. $\mathcal{L}$. □

Note that ω-controllability implies that $\mathcal{L}$ and $\mathcal{H}$ are non-conflicting and that $\operatorname{pre}\mathcal{H}$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \operatorname{pre}\mathcal{L})$; see Appendix, Lemma 2. In particular, if $\mathcal{H}$ is topologically closed, then the closed loop $\mathcal{K} = \mathcal{L} \cap \mathcal{H}$ is relatively topologically closed w.r.t. $\mathcal{L}$ and can be used to uniformly satisfy conditions (i) and (ii). This situation matches the notion of infinite-time controllability proposed by Ramadge (1989), where the exercised control is represented as a supervisor map $f\colon\ \Sigma^* \to \Gamma$, $\Gamma = \{\gamma \subseteq \Sigma \,|\, \Sigma_{\mathrm{uc}} \subseteq \gamma\}$, related to $\mathcal{K}$ by $f(s) = \{\sigma \,|\, s\sigma \in \operatorname{pre}\mathcal{K}\} \cup \Sigma_{\mathrm{uc}}$. If, on the other hand, the controller $\mathcal{H}$ is not topologically closed, then its effect cannot be represented by a single supervisor map f. For this situation, the quantification over all prefixes $s \in (\operatorname{pre}\mathcal{L}) \cap (\operatorname{pre}\mathcal{H})$ in the above definition effectively requires the persistent existence of supervisor maps f_s that can take over to run the plant for infinite time within $\mathcal{L} \cap \mathcal{H}$. This can be interpreted as a liveness property introduced by $\mathcal{H}$ and not present in $\mathcal{L}$; e.g., to eventually start a process (in contrast to doing so within a specified period of logic time). A conceptual benefit of this generalisation is that ω-controllability is retained under arbitrary union, whilst topological closedness is not. Thus, given a specification $\mathcal{E} \subseteq \mathcal{L}$, the supremal ω-controllable sublanguage $\mathcal{K}^{\Uparrow} \subseteq \mathcal{E}$ exists uniquely. It is only in the special case where $\mathcal{E}$ is relatively topologically closed w.r.t. $\mathcal{L}$, that $\mathcal{K}^{\Uparrow}$ turns out relatively topologically closed, too, and that $\mathcal{H} = \operatorname{clo}\mathcal{K}^{\Uparrow}$ can be used as a topologically closed controller with closed-loop behaviour $\mathcal{K}^{\Uparrow}$.

When the plant $\mathcal{L}$ and the specification $\mathcal{E} \subseteq \mathcal{L}$ are given as limits of regular $*$-languages, the computational procedure presented in (Thistle and Wonham, 1992), Section 8, can be applied to obtain a realisation of $\mathcal{K}^{\Uparrow}$. The procedure consists of five nested fixed-point iterations on the product state set of plant and specification, with runtime behaviour polynomial in the state count. Moreover, the procedure allows for the extraction of realisations for $\mathcal{V}_s$, $s \in \operatorname{pre}\mathcal{K}^{\Uparrow}$, to satisfy the above conditions (i) and (ii) and in addition to resolve eventuality properties in minimum logical time; e.g., when required to start a process eventually, it will be scheduled for the earliest time possible. Whenever $\mathcal{K}^{\Uparrow}$ is non-empty, one may use $\mathcal{H} = \operatorname{clo}\mathcal{V}_s$, $s = \epsilon$, as a controller that can be implemented as a single supervisor map $f\colon\ \Sigma^* \to \Gamma$ in order to enforce $\mathcal{E}$, however, in general not achieving the supremal closed-loop behaviour $\mathcal{K}^{\Uparrow}$. This can be interpreted as a practical solution to the synthesis for the case that $\mathcal{E}$ fails to be relatively topologically closed w.r.t. $\mathcal{L}$.

1.4 ω-admissibility

In order to address supervision under partial observation, we derive the following notion of *ω-admissibility* as a variation of ω-controllability. For the scope of this paper, all controllable events are assumed to be observable and, as with $*$-languages, we can utilize the concept of normality to guarantee that the supervisor maps f_s

are applicable under partial observation. A more general approach to the supervision of sequential behaviours under partial observation is presented in (Thistle and Lamouchi, 2009). The discussion therein includes the case of unobservable controllable events, however, the provided computational procedures require the plant to be topologically closed. Further related work from the area of program synthesis with incomplete information, based on alternating tree-automata, is given in (Kupferman and Vardi, 2000).

Definition 2 Let $\Sigma_{\mathrm{uc}} \subseteq \Sigma$, $\Sigma_{\mathrm{o}} \subseteq \Sigma$, $\Sigma - \Sigma_{\mathrm{uc}} \subseteq \Sigma_{\mathrm{o}}$, and consider the *plant* $\mathcal{L} \subseteq \Sigma^{\omega}$ and the *controller* $\mathcal{H} \subseteq \Sigma^{\omega}$. Then, $\mathcal{H}$ is said to be *ω-admissible* w.r.t. $(\Sigma_{\mathrm{uc}},\ \Sigma_{\mathrm{o}},\ \mathcal{L})$, if for all $s \in (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$ there exists a $\mathcal{V}_s \subseteq \mathcal{L} \cap \mathcal{H}$ with $s \in \mathrm{pre}\,\mathcal{V}_s$, and

(i) $\mathrm{pre}\,\mathcal{V}_s$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathrm{pre}\,\mathcal{L})$,

(ii) $\mathrm{pre}\,\mathcal{V}_s$ is prefix-normal w.r.t. $(\Sigma_{\mathrm{o}},\ \mathrm{pre}\,\mathcal{L})$, and

(iii) $\mathcal{V}_s$ is relatively topologically closed w.r.t. $\mathcal{L}$. □

Clearly, ω-admissibility implies ω-controllability. Note also that ω-admissibility implies that the local closed loop $(\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$ is prefix-normal w.r.t. $(\Sigma_{\mathrm{o}}, \mathrm{pre}\,\mathcal{L})$; see Appendix, Lemma 5. Furthermore, the above conditions (ii) and (iii) imply that $\mathcal{V}_s$ is normal w.r.t. $(\Sigma_{\mathrm{o}}, \mathcal{L})$. Now assume that $\mathcal{H}$ is topologically closed. Then, the closed loop $\mathcal{K} = \mathcal{L} \cap \mathcal{H}$ uniformly satisfies conditions (i)–(iii). Hence, referring to the situation of full observation, we obtain that the closed-loop behaviour $\mathcal{K}$ can be achieved by the supervisor map $f\colon\ \Sigma^* \to \Gamma$ defined $f(s) := \{\sigma \,|\, s\sigma \in \mathrm{pre}\,\mathcal{K}\} \cup \Sigma_{\mathrm{uc}}$. By the additional normality condition (ii), we have $\sigma \in f(s)$ if and only if $\mathrm{p_o}(s\sigma) \in \mathrm{p_o}\,\mathrm{pre}\,\mathcal{K}$ for any $\sigma \in \Sigma_{\mathrm{c}}$ with $s\sigma \in \mathrm{pre}\,\mathcal{L}$. Thus, the closed-loop behaviour $\mathcal{K}$ is also obtained under feedback $f'(s) := \{\sigma \,|\, \mathrm{p_o}(s\sigma) \in \mathrm{p_o}\,\mathrm{pre}\,\mathcal{K}\} \cup \Sigma_{\mathrm{uc}}$, which is implementable under partial observation. If, on the other hand, $\mathcal{H}$ is not topologically closed, the quantification over all prefixes $s \in (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$ ensures the persistent existence of supervisor maps f'_s that can take over to run the plant for infinite time within $\mathcal{L} \cap \mathcal{H}$. Here, we refer to condition (ii) and observe that $s \in \mathrm{pre}\,\mathcal{V}_s$ implies $s' \in \mathrm{pre}\,\mathcal{V}_s$ for any $s' \in (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$ with $\mathrm{p_o}s = \mathrm{p_o}s'$. Thus, if $\mathcal{H}$ is ω-admissible, the languages $\mathcal{V}_s$ can be chosen to additionally satisfy $\mathcal{V}_s = \mathcal{V}_{s'}$ for all s', $\mathrm{p_o}s = \mathrm{p_o}s'$. In particular, the application of a supervisor map f'_s can be based on the observation $\mathrm{p_o}s$. As with ω-controllability, this is interpreted as a liveness property introduced by $\mathcal{H}$ and not present in $\mathcal{L}$. Again, ω-admissibility is retained under arbitrary union, and, hence, given a specification $\mathcal{E} \subseteq \mathcal{L}$, the supremal ω-admissible sublanguage $\mathcal{K}^{\Uparrow} \subseteq \mathcal{E}$ exists uniquely; see Appendix, Lemma 3. Moreover, the supremal ω-admissible sublanguage $\mathcal{K}^{\Uparrow}$ turns out to be relatively topologically closed w.r.t. $\mathcal{L}$, provided that $\mathcal{E}$ is relatively topologically closed w.r.t. $\mathcal{L}$; see Appendix, Lemma 4.

We outline two procedures for the computation of an ω-admissible controller $\mathcal{H}$ for a given plant $\mathcal{L}$ and a given specification $\mathcal{E} \subseteq \mathcal{L}$, all realized as deterministic Büchi automata. First, assume that $\mathcal{E}$ is relatively topologically closed w.r.t. $\mathcal{L}$. For this case, a characterisation of $\mathcal{K}^{\Uparrow}$ in terms of limits of $*$-languages is given in the Appendix, Lemma 6. In particular, the algorithms provided by Moor et al (2012) can be applied to obtain a Büchi automata realisation of $\mathcal{K}^{\Uparrow}$, and we use $\mathcal{H} = \mathrm{clo}\,\mathcal{K}^{\Uparrow}$ as an ω-admissible controller with closed-loop behaviour $\mathcal{K}^{\Uparrow}$. Note that, due to natural projection operations and the construction of observer automata, the computational complexity is exponential in the number of plant and specification states. For our second computational procedure, we drop the assumption of a relatively closed specification. To our best knowledge, this situation has not been addressed in the literature. We, therefore, propose to first use the method presented in (Thistle and Wonham, 1992), Section 8, to obtain a representation of the supremal ω-controllable sublanguage, and in particular a relatively topologically closed behaviour $\mathcal{V}_\epsilon$ that satisfies conditions (i) and (iii). In a second stage, we use $\mathcal{V}_\epsilon$ as a relatively topologically closed specification for a subsequent synthesis via Lemma 6 and (Moor et al, 2012). The complexity is again exponential in the number of plant and specification states, and the result is guaranteed to be ω-admissible, however, not necessarily supremal. In ongoing work, we study variations of the algorithms by Thistle and Wonham (1992) in order to eliminate the intermediate step and to obtain a realisation of $\mathcal{K}^{\Uparrow}$.

1.5 Table of symbols

For easy reference, we provide a table of symbols used throughout this paper.

Symbol	Description	Page
Σ	overall alphabet, Sections 2 and 3	9 (Def. 3)
Σ_{c}	high-level control alphabet	9 (Def. 3)
$\Sigma_{\mathrm{cp}} = \Sigma_{\mathrm{c}} \dot\cup \Sigma_{\mathrm{p}}$	controller alphabet	9 (Def. 3)
$\Sigma_{\mathrm{pe}} = \Sigma_{\mathrm{p}} \dot\cup \Sigma_{\mathrm{e}}$	plant alphabet	9 (Def. 3)
Σ_{e}	low-level plant alphabet	9 (Def. 3)
Σ	overall alphabet, Section 4	18
$\Sigma_{\mathrm{el}} = \Sigma_{\mathrm{e}} \dot\cup \Sigma_{\mathrm{l}}$	environment alphabet	18 (Sec. 4.2)
$\mathcal{L}$	IO-plant over alphabet Σ_{pe}	8
$\mathcal{E}$	IO-specification over alphabet Σ	9
$\mathcal{H}$	IO-controller over alphabet Σ_{cp}	9
$\mathcal{I}$	IO-environment over alphabet Σ_{el}	16
$\mathcal{K}$	closed-loop behaviour over alphabet Σ	9
$\mathcal{L}_\Sigma$	full plant behaviour over alphabet Σ	8
$\mathcal{H}_\Sigma$	full controller behaviour over alphabet Σ	9
$\mathcal{I}_\Sigma$	full environment behaviour over alphabet Σ	16
$\mathcal{L}_{\mathrm{alt}}$	auxiliary language for event ordering	14
$\mathcal{L}_{\Vert}$	composition of IO-plants $\mathcal{L}_n$, $n = 1, ..., m$	15
$\mathcal{L}_{\mathrm{err}}$	error behaviour for composed IO-plants $\mathcal{L}_{\Vert}$	15
$\mathcal{L}_{\mathrm{IO}}$	IO-shuffle for IO-plants $\mathcal{L}_n$, $n = 1, ..., m$	15
pre	prefix closure operator for $*$-languages / prefix operator for ω-languages	3/4
lim	limit operator for $*$-languages	4
clo	topological closure operator ω-languages	4
$\mathrm{p}_{\_}$	projection from Σ^* to $\Sigma_{\_}^*$	4
$\mathrm{p}_{\_}^{-1}$	set-valued inverse projection from $\Sigma_{\_}^*$ to Σ^*	4
$\mathrm{p}_{\_}^{\omega}$	projection from Σ^ω to $\Sigma_{\_}^\omega \cup \Sigma_{\_}^*$	5
$\mathrm{p}_{\_}^{-\omega}$	set-valued inverse projection from $\Sigma_{\_}^\omega \cup \Sigma_{\_}^*$ to Σ^ω	5

2 Closed-loop with external signals

The closed-loop configuration under consideration consists of a controller component, a plant component, and three ports for system interconnection; see Figure 1, on the left. The motivation of explicitly addressing external interaction is to specify the relationship between internal and external behaviour as a formal requirement for the controller design. Each of the three ports is realised by synchronization of alternating

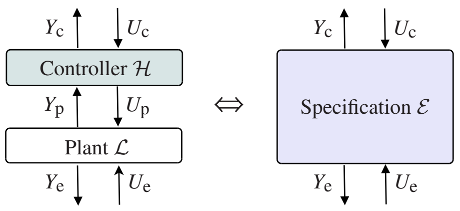


Fig. 1 Closed-loop configuration

input-events and output-events, from alphabets denoted U_{-} and Y_{-}, respectively. As in (Perk et al, 2006), this particular form of system interconnection refers to the notion of input-output systems by Willems (1991) and is a crucial prerequisite for our results on abstraction-based controller design in Section 3. Internally, the plant and the controller synchronize alternating symbols from $\Sigma_p = U_p \dot{\cup} Y_p$, and thus form a closed-loop configuration, similar to the common setting of sampled-data continuous control systems. Furthermore, the controller interacts with a high-level operator, while the plant is synchronized with a low-level environment. We take the perspective that the operator seeks to affect the environment according to high-level commands from U_c. The controller is meant to implement each high-level command on the plant by applying suitable events from U_p, while monitoring the plant responses ranging in Y_p. Eventually, the controller shall provide a high-level feedback event from Y_c to the operator, in order to receive the subsequent high-level command. A specification referring to the overall alphabet is meant to relate high-level events from $\Sigma_c = U_c \dot{\cup} Y_c$ with low-level events from $\Sigma_e = U_e \dot{\cup} Y_e$, and, thereby, formally define the consequences of high-level commands; see also Figure 1, to the right.

For the further discussion, we summarize the relevant parameters as a *control problem* and subsequently introduce conditions and requirements to characterize acceptable *solutions*.

Definition 3 A ***control problem*** consists of

$\Sigma := \Sigma_p \dot{\cup} \Sigma_e \dot{\cup} \Sigma_c$, the overall alphabet,
$\Sigma_c := U_c \dot{\cup} Y_c$, the *high-level control events*,
$\Sigma_p := U_p \dot{\cup} Y_p$, the *internal events*,
$\Sigma_e := U_e \dot{\cup} Y_e$, the *low-level plant events*,
$\mathcal{L} \subseteq (\Sigma_p \dot{\cup} \Sigma_e)^{\omega}$, the *plant behaviour*, and

$\mathcal{E} \subseteq \Sigma^{\omega}$, the *specification*.

Throughout this paper, the individual alphabets are obvious from the context and we concisely refer to the control problem by $(\Sigma, \mathcal{L}, \mathcal{E})$. Furthermore, we denote

$\Sigma_{\mathrm{pe}} := \Sigma_{\mathrm{p}} \dot\cup \Sigma_{\mathrm{e}}$, the *plant alphabet*,
$\Sigma_{\mathrm{cp}} := \Sigma_{\mathrm{c}} \dot\cup \Sigma_{\mathrm{p}}$, the *controller alphabet*,
$\Sigma_{\mathrm{ce}} := \Sigma_{\mathrm{c}} \dot\cup \Sigma_{\mathrm{e}}$, the *external alphabet*,
$\Sigma_{\mathrm{uc}} := U_{\mathrm{c}} \dot\cup Y_{\mathrm{p}} \dot\cup \Sigma_{\mathrm{e}}$, the *uncontrollable events*, and
$\Sigma_{\mathrm{o}} := \Sigma_{\mathrm{c}} \dot\cup \Sigma_{\mathrm{p}}$, the *observable events*. □

Projections from strings or infinite strings over Σ, to any of the above subsets of Σ, are denoted $\mathrm{p}_{\_}$ and $\mathrm{p}^{\omega}_{\_}$, respectively, with a subscript to indicate the respective range; e.g., p_{pe} for the projection from Σ^{*} to Σ^{*}_{pe}.

2.1 Plant properties

For the intended interpretation of inputs and outputs, the plant behaviour $\mathcal{L} \subseteq \Sigma^{\omega}_{\mathrm{pe}}$ must exhibit alternating input and output events, and, accept any input event from the controller and from the environment. For the acceptance of input events, we refer to the notion of a locally free input; see also Perk et al (2006).

Definition 4 For a language $L \subseteq \Sigma^{*}$, the alphabet $U \subseteq \Sigma$ is a *locally free input* if

$$(\forall\, s \in \Sigma^{*},\, \mu,\, \mu' \in U)\; [\, s\mu \in \operatorname{pre} L \Rightarrow s\mu' \in \operatorname{pre} L\,]\,. \qquad \square$$

Formally, we require the plant behaviour to possess properties P1 and P2 and refer to $\mathcal{L}$ as an ***IO-plant***:

P1 $\mathcal{L} \subseteq ((Y_{\mathrm{p}}U_{\mathrm{p}})^{*}(Y_{\mathrm{e}}U_{\mathrm{e}})^{*})^{\omega} \subseteq \Sigma^{\omega}_{\mathrm{pe}}$.
P2 $\operatorname{pre}\mathcal{L}$ possesses locally free inputs U_{p} and U_{e}.

For the subsequent discussion, it turns out convenient to raise $\mathcal{L} \subseteq \Sigma^{\omega}_{\mathrm{pe}}$ to the overall alphabet Σ, and to consider

$$\mathcal{L}_{\Sigma} := \,(\,\mathrm{p}^{-\omega}_{\mathrm{pe}}(\mathcal{L} \,\cup\, \operatorname{pre}\mathcal{L})) \cap (\operatorname{clo}\,(\,(Y_{\mathrm{p}}(Y_{\mathrm{c}}U_{\mathrm{c}})^{*}U_{\mathrm{p}})^{*}\,(Y_{\mathrm{e}}U_{\mathrm{e}})^{*}\,)^{\omega}\,)$$

as the *full plant behaviour*. The particular construction ensures that the inverse projection does not introduce artificial liveness properties while enforcing the intended event order. Moreover, if $\mathcal{L}$ is an IO-plant, then $\mathcal{L}_{\Sigma}$ possesses locally free inputs U_{c}, $U_{\mathrm{p}} \dot\cup Y_{\mathrm{c}}$ and U_{e} by construction.

2.2 Specification properties

The main purpose of the language inclusion specification $\mathcal{E} \subseteq \Sigma^{\omega}$ is to relate external to internal signals. However, for the hierarchical control architecture in Section 3, we also require that the external closed-loop behaviour again possesses the plant properties P1 and P2. In particular, the external closed-loop must persistently provide high-level feedback ranging in Y_c and it must accept any external input events from U_c and U_e. Technically, the ***IO-specification*** $\mathcal{E}$ must satisfy:

E1 $\mathcal{E} \subseteq (\,(\,(Y_\mathrm{p}U_\mathrm{p})^*\,(Y_\mathrm{e}U_\mathrm{e})^*\,)^*\,(Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^+U_\mathrm{p})\,)^\omega$.

E2 $\mathrm{pre}\,\mathcal{E}$ possesses locally free inputs U_c and U_e.

2.3 Solution to the control problem

Given a control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$ with an IO-plant $\mathcal{L}$ and an IO-specification $\mathcal{E}$, consider a candidate controller $\mathcal{H} \subseteq \Sigma_\mathrm{cp}^\omega$. For convenience, we write $\mathcal{H}_\Sigma := \mathrm{p}_\mathrm{cp}^{-\omega}\mathcal{H} \subseteq \Sigma^\omega$ for the controller behaviour w.r.t. the overall alphabet. For $\mathcal{H}$ to solve the control problem, it must enforce the specification and satisfy a controllability condition w.r.t. the plant behaviour. Formally, we impose the following conditions on $\mathcal{H}$ to form a ***solution*** to the control problem:

C1 $\mathcal{H}$ enforces the specification $\mathcal{E}$, i.e., $\mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma \subseteq \mathcal{E}$.

C2 $\mathcal{H}_\Sigma$ is ω-admissible[2] w.r.t. $(\Sigma_\mathrm{uc},\ \Sigma_\mathrm{o},\ \mathcal{L}_\Sigma)$.

If $\mathcal{H}$ is a solution, we obtain by C1 and E1 the *full closed-loop behaviour*

$$\mathcal{K} := \mathcal{L} \parallel \mathcal{H} := (\mathrm{p}_\mathrm{pe}^{-\omega}\mathcal{L})\ \cap\ (\mathrm{p}_\mathrm{cp}^{-\omega}\mathcal{H}) = \mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma,$$

where the last equality follows from the particular event ordering in $\mathcal{L}_\Sigma$ and $\mathcal{E}$; i.e., we have $\mathcal{L}_\Sigma\ \cap\ \mathcal{H}_\Sigma\ \subseteq \mathcal{E} \subseteq (\,(\,(Y_\mathrm{p}U_\mathrm{p})^*\,(Y_\mathrm{e}U_\mathrm{e})^*\,)^*\,(Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^+U_\mathrm{p})\,)^\omega$, and, thus, $(\mathrm{p}_\mathrm{pe}^{-\omega}\,\mathrm{pre}\,\mathcal{L})\ \cap\ \mathcal{L}_\Sigma\ \cap\ \mathcal{H}_\Sigma = \emptyset$. Furthermore, recall that C2 implies that $\mathcal{L}_\Sigma$ and $\mathcal{H}_\Sigma$ are non-conflicting. This implies $(\mathrm{pre}\,\mathrm{p}_\mathrm{pe}^{-\omega}\mathcal{L})\ \cap\ (\mathrm{pre}\,\mathrm{p}_\mathrm{cp}^{-\omega}\mathcal{H})\ = \mathrm{pre}(\,(\mathrm{p}_\mathrm{pe}^{-\omega}\mathcal{L})\ \cap\ (\mathrm{p}_\mathrm{cp}^{-\omega}\mathcal{H})\,)$, and we note that the plant $\mathcal{L}$ and the controller $\mathcal{H}$ form a non-conflicting closed-loop configuration.

2.4 Closed-loop properties

The following propositions relate solutions of a control problem to properties of the full closed-loop behaviour.

Proposition 1 If $\mathcal{H}$ is a solution to the control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$, where $\mathcal{L}$ is an IO-plant, then the full closed-loop behaviour $\mathcal{K} = \mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma$ satisfies K1–K5:

K1 $\mathcal{K}$ enforces the specification $\mathcal{E}$, i.e., $\mathcal{K} \subseteq \mathcal{E}$,

K2 $\mathcal{K}$ is ω-admissible w.r.t. $(\Sigma_\mathrm{uc},\ \Sigma_\mathrm{o},\ \mathcal{L}_\Sigma)$,

K3 $\mathcal{K}$ is normal w.r.t. $(\Sigma_\mathrm{o},\ \mathcal{L}_\Sigma)$,

K4 $\mathrm{pre}\,\mathcal{K}$ is prefix-normal w.r.t. $(\Sigma_\mathrm{o}, \mathrm{pre}\,\mathcal{L}_\Sigma)$,

K5 $\mathrm{pre}\,\mathcal{K}$ possesses locally free inputs U_c and U_e.

[2] In our conference contribution (Baier and Moor, 2012), we use the weaker requirement of ω-controllability. However, as discussed in the preliminaries, ω-admissible is more adequate for supervision of ω-languages under partial observation.

Proof K1 and K2 are immediate consequences of C1 and C2. For K3 observe that

$$\begin{aligned}\mathcal{K} \subseteq &(\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathrm{p}_{\mathrm{cp}}^{\omega}\mathcal{K}) \cap \mathcal{L}_\Sigma = (\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathrm{p}_{\mathrm{cp}}^{\omega}(\mathcal{H}_\Sigma \cap \mathcal{L}_\Sigma)) \cap \mathcal{L}_\Sigma \subseteq \\ &(\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathrm{p}_{\mathrm{cp}}^{\omega}\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathcal{H}) \cap (\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathrm{p}_{\mathrm{cp}}^{\omega}\mathcal{L}_\Sigma) \cap \mathcal{L}_\Sigma = \mathcal{H}_\Sigma \cap \mathcal{L}_\Sigma = \mathcal{K}.\end{aligned}$$

K4 is obtained by

$$\begin{aligned}\operatorname{pre}\mathcal{K} &\subseteq (\mathrm{p}_{\mathrm{cp}}^{-1}\mathrm{p}_{\mathrm{cp}}\operatorname{pre}\mathcal{K}) \cap (\operatorname{pre}\mathcal{L}_\Sigma) = \\ &(\mathrm{p}_{\mathrm{cp}}^{-1}\mathrm{p}_{\mathrm{cp}}\operatorname{pre}(\mathcal{H}_\Sigma \cap \mathcal{L}_\Sigma)) \cap (\operatorname{pre}\mathcal{L}_\Sigma) \subseteq \\ (\mathrm{p}_{\mathrm{cp}}^{-1}\mathrm{p}_{\mathrm{cp}}\operatorname{pre}\mathcal{H}_\Sigma) &\cap (\mathrm{p}_{\mathrm{cp}}^{-1}\mathrm{p}_{\mathrm{cp}}\operatorname{pre}\mathcal{L}_\Sigma) \cap (\operatorname{pre}\mathcal{L}_\Sigma) = \\ &(\operatorname{pre}\mathcal{H}_\Sigma) \cap (\operatorname{pre}\mathcal{L}_\Sigma) = \operatorname{pre}(\mathcal{H}_\Sigma \cap \mathcal{L}_\Sigma) = \operatorname{pre}\mathcal{K}.\end{aligned}$$

For the penultimate equality, recall that C2 implies that $\mathcal{L}_\Sigma$ and $\mathcal{H}_\Sigma$ are non-conflicting. Regarding K5, we pick $s,\ r \in \operatorname{pre}\mathcal{K}$, $\mu,\ \mu' \in U_{\mathrm{e}}$, and $\nu,\ \nu' \in U_{\mathrm{c}}$, such that $s\mu \in \operatorname{pre}\mathcal{K}$ and $r\nu \in \operatorname{pre}\mathcal{K}$. Observe that $s\mu,\ r\nu \in \operatorname{pre}\mathcal{K} \subseteq \operatorname{pre}\mathcal{L}_\Sigma$. According to P2 it follows that $s\mu' \in \operatorname{pre}\mathcal{L}_\Sigma$. Furthermore, the locally free input U_{c} of $\operatorname{pre}\mathcal{L}_\Sigma$ implies that $s\nu' \in \operatorname{pre}\mathcal{L}_\Sigma$. From ω-admissibility of $\mathcal{H}_\Sigma$ w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathcal{L}_\Sigma)$ and $s,\ r \in \operatorname{pre}\mathcal{H}_\Sigma$ follows that $s\mu',\ r\nu' \in \operatorname{pre}\mathcal{H}_\Sigma$. Recall again that $\mathcal{L}_\Sigma$ and $\mathcal{H}_\Sigma$ are non-conflicting, to obtain $s\mu',\ r\nu' \in (\operatorname{pre}\mathcal{L}_\Sigma) \cap (\operatorname{pre}\mathcal{H}_\Sigma) = \operatorname{pre}\mathcal{K}$. □

Vice versa, any ω-language that satisfies properties K1–K3 can be shown to be a solution to the control problem.

Proposition 2 Given a control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$, consider a closed-loop candidate $\mathcal{K} \subseteq \mathcal{L}_\Sigma$. If $\mathcal{K}$ satisfies K1–K3, then the controller $\mathcal{H} = \mathrm{p}_{\mathrm{cp}}^{\omega}\mathcal{K}$ solves the control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$.

Proof Note that K2, by Lemma 5 from the Appendix, implies that $(\operatorname{pre}\mathcal{L}_\Sigma) \cap (\operatorname{pre}\mathcal{K})$ is prefix-normal w.r.t. $(\Sigma_{\mathrm{o}}, \operatorname{pre}\mathcal{L}_\Sigma)$, and, together with $\mathcal{K} \subseteq \mathcal{L}_\Sigma$, we obtain K4. According to K1 and K3, we have that $\mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma = \mathcal{L}_\Sigma \cap (\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathrm{p}_{\mathrm{cp}}^{\omega}\mathcal{K}) = \mathcal{K} \subseteq \mathcal{E}$, hence, $\mathcal{H}$ satisfies C1. To establish C2, pick an arbitrary $s \in (\operatorname{pre}\mathcal{L}_\Sigma) \cap (\operatorname{pre}\mathrm{p}_{\mathrm{cp}}^{-\omega}\mathrm{p}_{\mathrm{cp}}^{\omega}\mathcal{K})$. Here, K4 implies $s \in \operatorname{pre}\mathcal{K}$. According to K2, we can choose $\mathcal{V}_s \subseteq \mathcal{L}_\Sigma \cap \mathcal{K} \subseteq \mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma$ such that $s \in \operatorname{pre}\mathcal{V}_s$, $\operatorname{pre}\mathcal{V}_s$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \operatorname{pre}\mathcal{L}_\Sigma)$, prefix-normal w.r.t. $(\Sigma_{\mathrm{o}}, \operatorname{pre}\mathcal{L}_\Sigma)$, and relatively closed w.r.t. $\mathcal{L}_\Sigma$. Hence, $\mathcal{H}_\Sigma$ is ω-admissible and satisfies C2. □

Conclusion. In this section, we proposed a control problem for a closed-loop with external signals. By the above Propositions 1 and 2, and in compliance with common approaches to supervisory control, a solution can be obtained from the supremal closed-loop behaviour $\mathcal{K}^{\Uparrow}$ that satisfies K1–K3, i.e., the supremal ω-admissible and normal sublanguage of a specification $\mathcal{E}$. When all relevant languages are given as limits of regular $*$-languages (or, equivalently, are realised by deterministic Büchi automata), and if $\mathcal{E}$ is relatively closed w.r.t. $\mathcal{L}_\Sigma$, $\mathcal{K}^{\Uparrow}$ can be obtained via Lemma 6 and the results from Moor et al (2012). More general settings are subject of ongoing research; see also the discussion on the Preliminaries and the example in Section 5.

3 Hierarchical controller design

Consider a control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$, a solution $\mathcal{H}$ and the full closed-loop behaviour $\mathcal{K} = \mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma$. The *external closed-loop behaviour* $\mathcal{L}^{\mathrm{hi}} := \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K}$ can again be interpreted as a plant. Thus, given a specification $\mathcal{E}^{\mathrm{hi}}$, we obtain another control problem $(\Sigma^{\mathrm{hi}},\ \mathcal{L}^{\mathrm{hi}},\ \mathcal{E}^{\mathrm{hi}})$. Provided we find a solution $\mathcal{H}^{\mathrm{hi}}$, we end up with a hierarchical control architecture; see Figure 2, to the right.

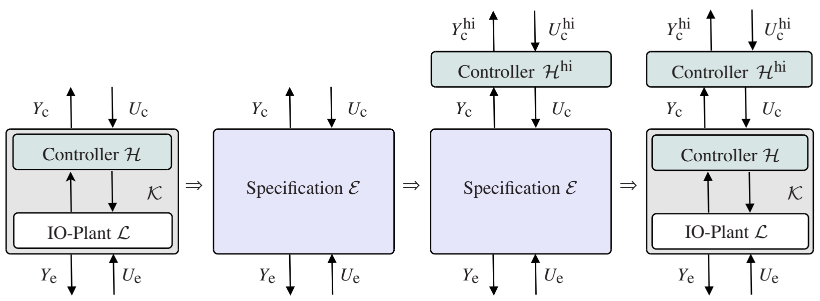


Fig. 2 Abstraction-based hierarchical controller design

Rather than to solve $(\Sigma^{\mathrm{hi}},\ \mathcal{L}^{\mathrm{hi}},\ \mathcal{E}^{\mathrm{hi}})$ directly, we propose to use the specification $\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{E}$ as an abstraction of the plant behaviour $\mathcal{L}^{\mathrm{hi}}$; see again Figure 2. Since $\mathcal{K} \subseteq \mathcal{E}$ implies $\mathcal{L}^{\mathrm{hi}} = \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K} \subseteq \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{E}$, solutions $\mathcal{H}^{\mathrm{hi}}$ of $(\Sigma^{\mathrm{hi}},\ \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{E},\ \mathcal{E}^{\mathrm{hi}})$ are readily observed to also satisfy C1 for the actual control problem $(\Sigma^{\mathrm{hi}},\ \mathcal{L}^{\mathrm{hi}},\ \mathcal{E}^{\mathrm{hi}})$. In contrast to the actual closed-loop $\mathcal{K}$, the specification $\mathcal{E}$ does not express how the control objective is achieved and, hence, is expected to be considerably less complex. The proposed approach raises two questions:

- Are the plant properties P1 and P2 of $\mathcal{L}$ retained under closed-loop composition and, thus, also satisfied by $\mathcal{L}^{\mathrm{hi}}$?
- Can we guarantee that the solutions of $(\Sigma^{\mathrm{hi}},\ \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{E},\ \mathcal{E}^{\mathrm{hi}})$ also solve the actual problem $(\Sigma^{\mathrm{hi}},\ \mathcal{L}^{\mathrm{hi}},\ \mathcal{E}^{\mathrm{hi}})$, i.e., possess not only C1 but also C2?

As it turns out, we need to impose an additional condition on the plant $\mathcal{L}$ in order to provide affirmative answers to both questions.

3.1 Non-anticipating IO-plant

In Moor et al (2011), it has been demonstrated that locally free inputs, as imposed by P2, do in general not imply a non-conflicting closed-loop for an abstraction-based controller design. While P2 requires the plant to accept any input locally, we need an additional structural plant property that requires that the liveness properties possessed by the plant may at no instance of time restrict the input in its infinite future. Hence, the plant shall always be in the position to choose its outputs such that it satisfies its own liveness properties; see Moor et al (2011) for a detailed discussion of this property, including examples. More specifically, the cited paper develops a variation of Willems' notion of non-anticipating input-output systems as a sufficient structural plant property for a non-conflicting closed-loop. Based on these considerations, we impose the additional requirement P3 on $\mathcal{L}$ and refer to the plant as a ***non-anticipating IO-plant***.

P3 $\mathcal{L}$ is ω-controllable w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{e},\ \mathrm{clo}\,\mathcal{L})$.

Technically, property P3 is a controllability condition, where the inputs events are considered the uncontrollable events, and can be verified, e.g., by the algorithm provided in (Thistle and Wonham, 1992), Section 8. The non-anticipating property propagates from $\mathcal{L}$ to the full plant behaviour $\mathcal{L}_\Sigma$; see Appendix, Lemma 7. By the following proposition, it is also preserved in the full closed-loop and we obtain an additional closed-loop property.

Proposition 3 If $\mathcal{H}$ is a solution to the control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$, and if $\mathcal{L}$ is a non-anticipating IO-plant, then

K6 $\mathcal{K}$ is ω-controllable w.r.t. $(U_\mathrm{c} \dot\cup U_\mathrm{e},\ \mathrm{clo}\,\mathcal{K})$.

Proof We prove the claim by construction of a suitable $\mathcal{V}_s \subseteq \mathcal{K}$ for an arbitrarily chosen $s \in \mathrm{pre}\,\mathcal{K}$. Note that, by Lemma 7 from the Appendix, $\mathcal{L}_\Sigma$ is a non-anticipating IO-plant. Thus, we can choose $\tilde{\mathcal{V}}_s \subseteq \mathcal{L}_\Sigma$, such that $s \in \mathrm{pre}\,\tilde{\mathcal{V}}_s$, $\mathrm{pre}\,\tilde{\mathcal{V}}_s$ is controllable w.r.t. $(\Sigma_\mathrm{c} \dot\cup U_\mathrm{p} \dot\cup U_\mathrm{e},\ \mathrm{pre}\,\mathcal{L}_\Sigma)$, and $\tilde{\mathcal{V}}_s$ is relatively closed w.r.t. $\mathrm{clo}\,\mathcal{L}_\Sigma$. In particular, $\tilde{\mathcal{V}}_s$ is closed. By Proposition 1, $\mathcal{K}$ satisfies K1–K5. Referring to K2, we choose $\mathcal{W}_s \subseteq \mathcal{K}$ with $s \in \mathrm{pre}\,\mathcal{W}_s$, and $\mathrm{pre}\,\mathcal{W}_s$ is controllable w.r.t. $(\Sigma_\mathrm{uc},\ \mathrm{pre}\,\mathcal{L}_\Sigma)$, and $\mathcal{W}_s$ is relatively closed w.r.t. $\mathcal{L}_\Sigma$. To establish ω-controllability of $\mathcal{K}$ w.r.t. $\mathrm{clo}\,\mathcal{K}$, consider the candidate $\mathcal{V}_s := \tilde{\mathcal{V}}_s \cap \mathcal{W}_s$. Clearly, $\mathcal{V}_s \subseteq \mathcal{K}$. Furthermore, $\mathcal{V}_s = \tilde{\mathcal{V}}_s \cap \mathcal{W}_s = \tilde{\mathcal{V}}_s \cap (\mathrm{clo}\,\mathcal{W}_s) \cap \mathcal{L}_\Sigma = \tilde{\mathcal{V}}_s \cap (\mathrm{clo}\,\mathcal{W}_s) = (\mathrm{clo}\,\tilde{\mathcal{V}}_s) \cap (\mathrm{clo}\,\mathcal{W}_s) \supseteq \mathrm{clo}\,\mathcal{V}_s$, i.e., $\mathcal{V}_s$ is closed and, thus, relatively closed w.r.t. any superset. To show controllability of $\mathrm{pre}\,\mathcal{V}_s$ w.r.t. $\mathrm{pre}\,\mathcal{K}$, we pick $r \in \mathrm{pre}(\tilde{\mathcal{V}}_s \cap \mathcal{W}_s) \subseteq (\mathrm{pre}\,\tilde{\mathcal{V}}_s) \cap (\mathrm{pre}\,\mathcal{W}_s)$ and $\sigma \in U_\mathrm{c} \dot\cup U_\mathrm{e}$ such that $r\sigma \in \mathrm{pre}\,\mathcal{K} \subseteq \mathrm{pre}\,\mathcal{L}_\Sigma$. By controllability of $\mathrm{pre}\,\tilde{\mathcal{V}}_s$ and $\mathrm{pre}\,\mathcal{W}_s$, it follows that $r\sigma \in (\mathrm{pre}\,\tilde{\mathcal{V}}_s) \cap (\mathrm{pre}\,\mathcal{W}_s)$. To establish $r\sigma \in \mathrm{pre}(\tilde{\mathcal{V}}_s \cap \mathcal{W}_s)$, observe that each event in Σ is uncontrollable for either $\mathrm{pre}\,\tilde{\mathcal{V}}_s$ or $\mathrm{pre}\,\mathcal{W}_s$. Thus, starting with $r_0 = r\sigma$, we can construct an unbounded sequence $(r_n) \subseteq (\mathrm{pre}\,\tilde{\mathcal{V}}_s) \cap (\mathrm{pre}\,\mathcal{W}_s)$ with limit $w := \lim(r_n) \in (\mathrm{clo}\,\tilde{\mathcal{V}}_s) \cap (\mathrm{clo}\,\mathcal{W}_s)$. Since $\tilde{\mathcal{V}}_s$ is closed, we have $w \in \tilde{\mathcal{V}}_s \subseteq \mathcal{L}_\Sigma$. By relative closedness of $\mathcal{W}_s$ w.r.t. $\mathcal{L}_\Sigma$, we obtain $w \in \mathcal{W}_s$. Hence, $r\sigma \in \mathrm{pre}(\tilde{\mathcal{V}}_s \cap \mathcal{W}_s)$. □

3.2 Propagation of plant properties

We are now in the position to show that the plant properties P1–P3 are retained under closed-loop composition, i.e., the external closed-loop behaviour is again a non-anticipating IO-plant.

Theorem 1 For a non-anticipating IO-plant $\mathcal{L}$ and an IO-specification $\mathcal{E}$, consider a solution $\mathcal{H}$ of the control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$. Then the external closed-loop $\mathrm{p}^\omega_\mathrm{ce}\mathcal{K}$, with $\mathcal{K} = \mathcal{L}_\Sigma \cap \mathcal{H}_\Sigma$, is a non-anticipating IO-plant, too.

Proof Regarding the event ordering P1, we refer to K1 and E1 to obtain $\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K} \subseteq ((Y_{\mathrm{c}}U_{\mathrm{c}})^{*}(Y_{\mathrm{e}}U_{\mathrm{e}})^{*})^{\omega}$. Regarding locally free inputs P2, recall from K5 that $\mathcal{K}$ has locally free inputs U_{c} and U_{e}, that are preserved under projection to Σ_{ce}. We are left to verify non-anticipation P3. Pick $s \in \mathrm{pre}\,\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K}$. Then, there exists $t \in \mathrm{pre}\,\mathcal{K}$ such that $\mathrm{p}_{\mathrm{ce}}t = s$. According to K6, we can choose $\mathcal{W}_t \subseteq \mathcal{K}$ such that $t \in \mathrm{pre}\,\mathcal{W}_t$, and $\mathrm{pre}\,\mathcal{W}_t$ is controllable w.r.t. $(U_{\mathrm{c}} \dot{\cup} U_{\mathrm{e}},\ \mathrm{pre}\,\mathcal{K})$, and $\mathcal{W}_t$ is closed. As a candidate to establish P3, let $\mathcal{V}_s := \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{W}_t$. Note that $\mathcal{V}_s \subseteq \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K}$. Further, $s = \mathrm{p}_{\mathrm{ce}}t \in \mathrm{p}_{\mathrm{ce}}\,\mathrm{pre}\,\mathcal{W}_t = \mathrm{pre}\,\mathcal{V}_s$. To verify controllability of $\mathrm{pre}\,\mathcal{V}_s$, consider an arbitrary $\hat{s} \in \mathrm{pre}\,\mathcal{V}_s$ and $\sigma \in U_{\mathrm{c}} \dot{\cup} U_{\mathrm{e}}$ such that $\hat{s}\sigma \in \mathrm{pre}\,\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K}$. Then, there exists $\hat{t} \in \mathrm{pre}\,\mathcal{K}$ such that $\mathrm{p}_{\mathrm{ce}}\hat{t} = \hat{s}$ and $\hat{t} \in \mathrm{pre}\,\mathcal{W}_t$. Furthermore, $\hat{t}\sigma \in \mathrm{pre}\,\mathcal{K}$, since $\hat{s}\sigma = \mathrm{p}_{\mathrm{ce}}(\hat{t}\sigma) \in \mathrm{p}_{\mathrm{ce}}\,\mathrm{pre}\,\mathcal{K}$. Controllability of $\mathrm{pre}\,\mathcal{W}_t$ implies that $\hat{t}\sigma \in \mathrm{pre}\,\mathcal{W}_t$ and $\mathrm{p}_{\mathrm{ce}}(\hat{t}\sigma) \in \mathrm{p}_{\mathrm{ce}}\,\mathrm{pre}\,\mathcal{W}_t = \mathrm{pre}\,\mathcal{V}_s$. Consequently, the candidate $\mathrm{pre}\,\mathcal{V}_s$ is controllable w.r.t. $(U_{\mathrm{c}} \dot{\cup} U_{\mathrm{e}},\ \mathrm{pre}\,\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{K})$. To verify closedness of $\mathcal{V}_s$, observe that $\mathcal{V}_s = \mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{W}_t = (\mathrm{p}^{\omega}_{\mathrm{ce}}\,\mathrm{clo}\,\mathcal{W}_t) \cap \Sigma^{\omega}_{\mathrm{ce}} = \mathrm{clo}\,\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{W}_t$. □

3.3 Abstraction-based controller design

We adapt the argument regarding abstraction-based controller design from (Moor et al, 2011) to the closed-loop configuration with external signals, see Figure 1, to obtain the following theorem.

Theorem 2 Given a control problem $(\Sigma,\ \mathcal{L},\ \mathcal{E})$ with a non-anticipating IO-plant $\mathcal{L}$, let $\mathcal{L}' \subseteq \Sigma^{\omega}$ denote a plant abstraction, i.e., $\mathcal{L} \subseteq \mathcal{L}'$. Then, any solution $\mathcal{H}$ of $(\Sigma,\ \mathcal{L}',\ \mathcal{E})$ also solves $(\Sigma,\ \mathcal{L},\ \mathcal{E})$.

The proof of Theorem 2 refers to (Baier and Moor, 2012), Lemma 14, which, for convenience, is repeated below, with a proof provided in the Appendix.

Lemma 1 Under the hypothesis of Theorem 2, consider any solution $\mathcal{H}$ of the control problem $(\Sigma,\ \mathcal{L}',\ \mathcal{E})$. If $\mathcal{V}' \subseteq \mathcal{L}'_{\Sigma} \cap \mathcal{H}_{\Sigma}$, and if $\mathrm{pre}\,\mathcal{V}'$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathcal{L}'_{\Sigma})$, and if $\mathcal{V}'$ is relatively closed w.r.t. $\mathcal{L}'_{\Sigma}$, then $\mathcal{L}_{\Sigma}$ and $\mathcal{V}'$ are non-conflicting. □

Hence, any controller candidate $\mathcal{V}'$, found by using the abstraction $\mathcal{L}'$ as plant, will result in a non-conflicting closed loop composed from the original plant $\mathcal{L}$ and the candidate $\mathcal{V}'$.

Proof of Theorem 2. Note that $\mathcal{H}$ trivially enforces the specification C1, since $\mathcal{L}_{\Sigma} \cap \mathcal{H}_{\Sigma} \subseteq \mathcal{L}'_{\Sigma} \cap \mathcal{H}_{\Sigma} \subseteq \mathcal{E}$. We are left to verify admissibility C2. Pick an arbitrary $s \in (\mathrm{pre}\,\mathcal{L}_{\Sigma}) \cap (\mathrm{pre}\,\mathcal{H}_{\Sigma})$. Since $\mathcal{H}$ is a solution to $(\Sigma,\ \mathcal{L}',\ \mathcal{E})$, we can choose $\mathcal{V}'_s \subseteq \mathcal{L}'_{\Sigma} \cap \mathcal{H}_{\Sigma}$ such that $s \in \mathrm{pre}\,\mathcal{V}'_s$, and $\mathrm{pre}\,\mathcal{V}'_s$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathrm{pre}\,\mathcal{L}'_{\Sigma})$, and $\mathrm{pre}\,\mathcal{V}'_s$ is prefix-normal w.r.t. $(\Sigma_{\mathrm{o}},\ \mathrm{pre}\,\mathcal{L}'_{\Sigma})$, and $\mathcal{V}'_s$ is relatively closed w.r.t. $\mathcal{L}'_{\Sigma}$. We choose the candidate $\mathcal{V}_s := \mathcal{V}'_s \cap \mathcal{L}_{\Sigma}$. Observe that $\mathcal{V}_s \subseteq \mathcal{L}'_{\Sigma} \cap \mathcal{H}_{\Sigma} \cap \mathcal{L}_{\Sigma} = \mathcal{H}_{\Sigma} \cap \mathcal{L}_{\Sigma}$ and $s \in (\mathrm{pre}\,\mathcal{L}_{\Sigma}) \cap (\mathrm{pre}\,\mathcal{H}_{\Sigma}) \cap (\mathrm{pre}\,\mathcal{V}'_s)$. By Lemma 1, we obtain $s \in \mathrm{pre}(\mathcal{L}_{\Sigma} \cap \mathcal{V}'_s) = \mathrm{pre}\,\mathcal{V}_s$. Regarding controllability, pick any $s\nu \in \mathrm{pre}\,\mathcal{L}_{\Sigma}$ with $s \in \mathrm{pre}\,\mathcal{V}_s$ and $\nu \in \Sigma_{\mathrm{uc}}$. By controllability of $\mathrm{pre}\,\mathcal{V}'_s$ w.r.t. $\mathrm{pre}\,\mathcal{L}'_{\Sigma}$ and $\mathcal{L}_{\Sigma} \subseteq \mathcal{L}'_{\Sigma}$, we deduce that $s\nu \in \mathrm{pre}\,\mathcal{V}'_s$. Again by Lemma 1, we obtain $s\nu \in \mathrm{pre}(\mathcal{L}_{\Sigma} \cap \mathcal{V}'_s)$. Hence, $\mathrm{pre}\,\mathcal{V}_s$ is controllable w.r.t. $\mathrm{pre}\,\mathcal{L}_{\Sigma}$. Regarding prefix-normality, we first observe that $(\mathrm{pre}\,\mathcal{L}_{\Sigma}) \cap (\mathrm{p}^{-1}_{\mathrm{cp}}\mathrm{p}_{\mathrm{cp}}\,\mathrm{pre}\,\mathcal{V}_s) \subseteq (\mathrm{pre}\,\mathcal{L}_{\Sigma}) \cap (\mathrm{p}^{-1}_{\mathrm{cp}}\mathrm{p}_{\mathrm{cp}}\,\mathrm{pre}\,\mathcal{V}'_s) = (\mathrm{pre}\,\mathcal{L}_{\Sigma}) \cap (\mathrm{pre}\,\mathcal{L}'_{\Sigma}) \cap (\mathrm{p}^{-1}_{\mathrm{cp}}\mathrm{p}_{\mathrm{cp}}\,\mathrm{pre}\,\mathcal{V}'_s) = (\mathrm{pre}\,\mathcal{L}_{\Sigma}) \cap (\mathrm{pre}\,\mathcal{V}'_s) = \mathrm{pre}(\mathcal{L}_{\Sigma} \cap$

$\mathcal{V}_s')=\operatorname{pre}\mathcal{V}_s$, where the last equality is by Lemma 1. Together with $\operatorname{pre}V_s\subseteq\operatorname{pre}\mathcal{L}_\Sigma$, this constitutes prefix-normality of $\operatorname{pre}\mathcal{V}_s$ w.r.t. $(\Sigma_o,\ \mathcal{L}_\Sigma)$. Regarding relative closedness, observe that $(\operatorname{clo}\mathcal{V}_s)\cap\mathcal{L}_\Sigma\subseteq(\operatorname{clo}\mathcal{V}_s')\cap\mathcal{L}_\Sigma'\cap\mathcal{L}_\Sigma=\mathcal{V}_s'\cap\mathcal{L}_\Sigma=\mathcal{V}_s$. This concludes the verification of admissibility C2. □

Conclusion. As our main result in Section 3, we proved that the plant properties P1–P3 of $\mathcal{L}$ are retained under closed-loop composition with $\mathcal{H}$ and, thus, also satisfied by $\mathcal{L}^{\mathrm{hi}}$. Moreover, the solutions of $(\Sigma^{\mathrm{hi}},\ p^{\omega}_{\mathrm{ce}}\mathcal{E},\ \mathcal{E}^{\mathrm{hi}})$ possess both properties C1 and C2 and, thus, also solve the actual problem $(\Sigma^{\mathrm{hi}},\ \mathcal{L}^{\mathrm{hi}},\ \mathcal{E}^{\mathrm{hi}})$. In summary, Theorem 1 and Theorem 2 formally justify the hierarchical controller design as proposed by Figure 2.

4 Composition of modular plants

The modular control systems under consideration consist of a number of independent non-anticipating IO-plants. In contrast to other approaches of modular supervision, we require that all alphabets are disjoint and represent dependencies between individual subsystems (e.g. shared resources) by so called *IO-environments*. In this section, we investigate the overall behaviour of the *modular IO-system*, see Figure 3, in order to verify that the IO-plant properties are retained under the proposed system composition. Thus, the resulting external behaviour is again an IO-plant, subject to subsequent controller design.

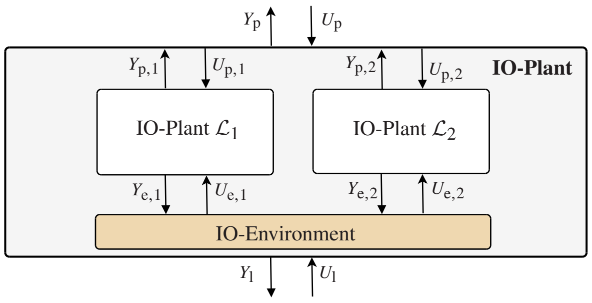


Fig. 3 Modular IO-System

4.1 Composition of non-anticipating IO-plants

We first let aside the IO-environment and consider the parallel composition of $m\in\mathbb{N}$ IO-plants $\mathcal{L}_n\subseteq\Sigma_n^\omega$, $\Sigma_n=U_{\mathrm{p},n}\,\dot\cup\,Y_{\mathrm{p},n}\,\dot\cup\,U_{\mathrm{e},n}\,\dot\cup\,Y_{\mathrm{e},n}$, $n=1,\ldots,m$, i.e.,

$$\mathcal{L}_1\parallel\cdots\parallel\mathcal{L}_m:=\{\,w\in\Sigma_{\mathrm{pe}}^\omega\mid \mathrm{p}_n^\omega w\in\mathcal{L}_n\ \text{for}\ n=1,\ldots,m\,\},$$

using the notational convention $\Sigma_{\mathrm{pe}}:=\Sigma_1\,\dot\cup\cdots\dot\cup\,\Sigma_m$ for the overall plant alphabet, and

$$U_{\mathrm{p}}:=\dot{\bigcup}_{n=1}^m U_{\mathrm{p},n},\ Y_{\mathrm{p}}:=\dot{\bigcup}_{n=1}^m Y_{\mathrm{p},n},\ U_{\mathrm{e}}:=\dot{\bigcup}_{n=1}^m U_{\mathrm{e},n},\ Y_{\mathrm{e}}:=\dot{\bigcup}_{n=1}^m Y_{\mathrm{e},n}.$$

for the respective input alphabets and output alphabets.

Since we assume all alphabets disjoint, the above parallel composition amounts to a shuffle product. However, in order to maintain the structural requirement of input-output alternation in the composed system, it is necessary to restrict the shuffle accordingly. To this end, we propose to intersect with

$$\mathcal{L}_{\mathrm{alt}} := (\Sigma_1\,\Sigma_1 + \cdots + \Sigma_m\,\Sigma_m)^\omega\,.$$

Thus, a controller that operates a composed system is meant to reply instantaneously to any output event by an input event directed to the respective plant component. In order to formally obtain a locally free input $U_\mathrm{p} \cup U_\mathrm{e}$, we take the union with an artificial error behaviour $\mathcal{L}_{\mathrm{err}}$ that accounts for miss-directed input events, i.e.,

$$\begin{aligned}\mathcal{L}_{\mathrm{err}} := &\bigcup_{\substack{k,n=1\\k\neq n}}^{m} ((\Sigma_{\mathrm{pe}}^* Y_{\mathrm{p},n}) \cap (\operatorname{pre}\mathcal{L}_{\|}))U_{\mathrm{p},k}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*)^\omega\\ &\bigcup_{\substack{k,n=1\\k\neq n}}^{m} ((\Sigma_{\mathrm{pe}}^* Y_{\mathrm{e},n}) \cap (\operatorname{pre}\mathcal{L}_{\|}))U_{\mathrm{e},k}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*)^\omega\,.\end{aligned}$$

and consider the **IO-shuffle**

$$\mathcal{L}_{\mathrm{IO}} := \mathcal{L}_1 \parallel_{\mathrm{IO}} \cdots \parallel_{\mathrm{IO}} \mathcal{L}_m := \mathcal{L}_{\|} \cup \mathcal{L}_{\mathrm{err}} \subseteq \Sigma_{\mathrm{pe}}^\omega\,,$$

with

$$\mathcal{L}_{\|} := (\mathcal{L}_1 \parallel \cdots \parallel \mathcal{L}_m) \cap \mathcal{L}_{\mathrm{alt}}\,,$$

as the behaviour of the composed IO-plants $\mathcal{L}_n$, $n = 1, \ldots, m$. Note that any relevant specification will implicitly prevent the controller from issuing miss-directed input events in order to avoid the error behaviour in the closed-loop configuration.

By the following proposition, the construction so far preserves the IO-plant properties P1–P3.

Proposition 4 Given $m \in \mathbb{N}$ non-anticipating IO-plants $\mathcal{L}_n \subseteq \Sigma_n^\omega$, $n = 1, \ldots, m$, then the IO-shuffle $\mathcal{L}_{\mathrm{IO}} := \mathcal{L}_1 \parallel_{\mathrm{IO}} \cdots \parallel_{\mathrm{IO}} \mathcal{L}_n$ is a non-anticipating IO-plant, too.

Proof Regarding the event ordering P1, and as a consequence of $\mathcal{L}_{\mathrm{alt}}$ and the alternating inputs and outputs from each component $\mathcal{L}_n$, $n = 1, \ldots, m$, observe that $\mathcal{L}_{\|} \subseteq [(Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*]^\omega$. Referring to the definition of $\mathcal{L}_{\mathrm{err}}$, this implies that $\mathcal{L}_{\mathrm{err}} \subseteq [(Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*]^\omega$, and, hence, $\mathcal{L}_{\mathrm{IO}} \subseteq [(Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*]^\omega$. Regarding locally free inputs P2, we focus attention on the input alphabet U_p, pick an arbitrary $s\mu \in \operatorname{pre}\mathcal{L}_{\mathrm{IO}}$, with $\mu \in U_\mathrm{p}$, and an alternative input symbol $\mu' \in U_\mathrm{p}$. By P1, we decompose $s = t\nu$ with $\nu \in Y_{\mathrm{p},n}$ for some n. If $t\nu \in \mathcal{L}_{\mathrm{err}}$, we obtain $t\nu\mu' \in \operatorname{pre}\mathcal{L}_{\mathrm{err}} \subseteq \operatorname{pre}\mathcal{L}_{\mathrm{IO}}$ by the definition of $\mathcal{L}_{\mathrm{err}}$, and, hence, $t\nu\mu' \in \mathcal{L}_{\mathrm{IO}}$. Else, we have $t\nu \notin \mathcal{L}_{\mathrm{err}}$ and obtain $t\nu \in \operatorname{pre}\mathcal{L}_{\|}$ by the definition of $\mathcal{L}_{\mathrm{IO}}$. Here, we distinguish two more cases. First, if $\mu' \in U_{\mathrm{p},n}$, the locally free input $U_{\mathrm{p},n}$ of $\operatorname{pre}\mathcal{L}_n$ implies that $\mathrm{p}_n(s)\nu\mu' \in \operatorname{pre}\mathcal{L}_n$ and, thus $t\nu\mu' \in \operatorname{pre}\mathcal{L}_{\|} \subseteq \operatorname{pre}\mathcal{L}_{\mathrm{IO}}$. In the second case we have $\mu' \in U_{\mathrm{p},k}$ for some $k \neq n$, and again obtain $t\nu\mu' \in \operatorname{pre}\mathcal{L}_{\mathrm{err}} \subseteq \operatorname{pre}\mathcal{L}_{\mathrm{IO}}$. This establishes that U_p is a locally free input of $\mathcal{L}_{\mathrm{IO}}$. The free input U_e is verified likewise and this concludes the proof of locally free inputs P2. Regarding non-anticipation P3, we verify ω-controllability of

$\mathcal{L}_\|$ w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{e}, \mathrm{clo}\,\mathcal{L}_\|)$. For clarity of representation, trivial case distinctions to extend the argument to account for the error behaviour $\mathcal{L}_\mathrm{err}$ have been omitted. Pick an arbitrary string $s \in \mathrm{pre}\,\mathcal{L}_\|$ and denote $r_n := \mathrm{p}_n s$ for $n = 1, \dots, m$. Choose $w \in \mathcal{L}_\|$, $s < w$, to observe $\mathrm{p}_n^\omega w \in \mathcal{L}_n$ and, hence, $r_n \in \mathrm{pre}\,\mathcal{L}_n$, for all $n = 1, \dots, m$. Since each $\mathcal{L}_n$ is non-anticipating, we can choose $\mathcal{V}_{r,n} \subseteq \mathcal{L}_n \subseteq \mathcal{L}_\mathrm{alt}$ with $r_n \in \mathrm{pre}\,\mathcal{V}_{r,n}$ and $\mathrm{pre}\,\mathcal{V}_{r,n}$ is controllable w.r.t. $(U_{\mathrm{p},n} \dot\cup U_{\mathrm{e},n}, \mathrm{pre}\,\mathcal{L}_n)$ and $\mathcal{V}_{r,n}$ is relatively closed w.r.t. $\mathrm{clo}\,\mathcal{L}_n$. In particular, $\mathcal{V}_{r,n}$ is closed. For the string s, we define the candidate

$$\mathcal{V}_s := \Big(\bigcap_{n=1}^{m} \mathrm{clo}\,\mathrm{p}_n^{-\omega} \mathcal{V}_{r,n} \Big) \cap \mathcal{L}_\mathrm{seq}$$

with $\mathcal{L}_\mathrm{seq} := \mathcal{L}_\mathrm{alt} \cap \Sigma^{|s|}(\epsilon + \Sigma)(\Sigma_1 \Sigma_1 \cdots \Sigma_m \Sigma_m)^\omega$. Note that, as a finite intersection of closed languages, $\mathcal{V}_s$ itself is closed. To show that $\mathcal{V}_s \subseteq \mathcal{L}_\|$, pick $w \in \mathcal{V}_s$ and n arbitrarily. By $w \in \mathrm{clo}\,\mathrm{p}_n^{-\omega}\mathcal{V}_{r,n}$, we have $\mathrm{pre}\,w \subseteq \mathrm{pre}\,\mathrm{p}_n^{-\omega}\mathcal{V}_{r,n}$ and, hence, $\mathrm{p}_n\,\mathrm{pre}\,w \subseteq \mathrm{pre}\,\mathcal{V}_{r,n}$. Referring to the definition of $\mathcal{L}_\mathrm{seq}$, $\mathrm{p}_n\,\mathrm{pre}\,w$ is unbounded and we obtain $\{\mathrm{p}_n^\omega w\} = \lim \mathrm{p}_n\,\mathrm{pre}\,w \subseteq \lim \mathrm{pre}\,\mathcal{V}_{r,n} = \mathcal{V}_{r,n}$, i.e., $\mathrm{p}_n^\omega w \in \mathcal{V}_{r,n}$. By the arbitrary choice of w and n, we conclude $\mathcal{V}_s \subseteq \mathcal{L}_\|$. Given an arbitrary $\hat{s} \in \mathrm{pre}\,\mathcal{L}_\mathrm{seq}$, we claim that $\hat{r}_n := \mathrm{p}_n \hat{s} \in \mathrm{pre}\,\mathcal{V}_{r,n}$ for all n implies $\hat{s} \in \mathrm{pre}\,\mathcal{V}_s$. From $\hat{r}_n \in \mathrm{pre}\,\mathcal{V}_{r,n}$, we choose $\hat{u}_n \in \Sigma_n^\omega$ such that $\hat{r}_n \hat{u}_n \in \mathcal{V}_{r,n}$. The alphabets Σ_1 to Σ_m are disjoint, and we can choose $\hat{u}$ in the shuffle $\mathrm{p}_1^{-\omega}\hat{u}_1 \cap \cdots \cap \mathrm{p}_m^{-\omega}\hat{u}_m$ such that $\mathrm{p}_n^\omega(\hat{s}\hat{u}) = \hat{r}_n\hat{u}_n \in \mathcal{V}_{r,n}$ and $\hat{s}\hat{u} \in \mathcal{L}_\mathrm{seq}$. Thus, we have indeed $\hat{s} \in \mathrm{pre}\,\mathcal{V}_s$. This concludes the proof of our claim, with $s \in \mathrm{pre}\,\mathcal{V}_s$ as an immediate consequence. Regarding controllability of $\mathrm{pre}\,\mathcal{V}_s$ w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{e}, \mathrm{pre}\,\mathcal{L}_\|)$, pick an arbitrary string $\hat{s} \in \mathrm{pre}\,\mathcal{V}_s$ and $\sigma \in \Sigma_\mathrm{uc}$, such that $\hat{s}\sigma \in \mathrm{pre}\,\mathcal{L}_\|$. Denote j the index of the corresponding component, i.e. $\sigma \in \Sigma_j$. Controllability of $\mathrm{pre}\,\mathcal{V}_{r,j}$ w.r.t. $\mathrm{pre}\,\mathcal{L}_j$ implies $(\mathrm{p}_j\hat{s})\sigma \in \mathrm{pre}\,\mathcal{V}_{r,j}$. Thus, we have $\mathrm{p}_n(\hat{s}\sigma) \in \mathrm{pre}\,\mathcal{V}_{r,n}$ for all n and $\hat{s}\sigma \in \mathrm{pre}\,\mathcal{L}_\mathrm{seq}$. This implies $\hat{s}\sigma \in \mathrm{pre}\,\mathcal{V}_s$. □

4.2 IO-environment

The IO-environment models dependencies between the individual plant components; see Figure 3. It is meant to facilitate the reuse of plant models when operated in different environments. Technically, we require the **IO-environment** $\mathcal{I} \subseteq \Sigma_\mathrm{el}^\omega$, defined over the alphabet $\Sigma_\mathrm{el} := \Sigma_\mathrm{e} \dot\cup \Sigma_\mathrm{l}$, with $\Sigma_\mathrm{l} := Y_\mathrm{l} \dot\cup U_\mathrm{l}$, to possess the below properties.

I1 $\mathcal{I} \subseteq ((Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega$.

I2 $\mathrm{pre}\,\mathcal{I}$ possesses locally free inputs U_l and Y_e.

I3 $\mathcal{I}$ is topologically closed.

The composition of the IO-plant $\mathcal{L}_\mathrm{IO}$ with an IO-environment $\mathcal{I}$ can be discussed in analogy to the construction of the full closed-loop behaviour presented in Section 2, where $\mathcal{L}_\mathrm{IO}$ plays the role of the plant and $\mathcal{I}$ corresponds to the controller. More specifically, we raise both languages to the overall alphabet $\Sigma := \Sigma_\mathrm{p} \dot\cup \Sigma_\mathrm{e} \dot\cup \Sigma_\mathrm{l}$, to consider the *full IO-shuffle behaviour* and the *full IO-environment behaviour*,

$$\begin{aligned}\mathcal{L}_\Sigma &:= \ (\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{L}_{\mathrm{IO}} \cup \mathrm{pre}\,\mathcal{L}_{\mathrm{IO}})) \ \cap \ (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\,,\\ \mathcal{I}_\Sigma &:= \ (\mathrm{p}_{\mathrm{el}}^{-\omega}(\mathcal{I} \cup \mathrm{pre}\,\mathcal{I})) \ \cap \ (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\,,\end{aligned}$$

respectively, and we obtain the intersection

$$\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma \subseteq \Sigma^\omega$$

to represent the **modular IO-system**, Figure 3. By the particular construction and by condition I3, $\mathcal{I}_\Sigma$ is observed to be topologically closed. Still following the discussion of the closed loop with external signals, proceeding with Section 3, we derive two propositions to observe that the non-anticipating property is retained under the proposed composition.

Proposition 5 For non-anticipating IO-plant components $\mathcal{L}_n \subseteq \Sigma_n^\omega$, $n = 1, \ldots, m$, the full IO-shuffle $\mathcal{L}_\Sigma \subseteq \Sigma^\omega$ is ω-controllable w.r.t. $(U_\mathrm{p} \,\dot\cup\, U_\mathrm{l}, \mathrm{clo}\,\mathcal{L}_\Sigma)$.

Proof The proof is almost literally identical to the proof of (Baier and Moor, 2012), Proposition 9. For convenience, an explicit proof is provided in the Appendix. □

Proposition 6 For non-anticipating IO-plant components $\mathcal{L}_n \subseteq \Sigma_n^\omega$, $n = 1, \ldots, m$ and an IO-environment $\mathcal{I} \subseteq \Sigma_{\mathrm{el}}^\omega$, consider the full behaviours $\mathcal{L}_\Sigma \subseteq \Sigma^\omega$ and $\mathcal{I}_\Sigma \subseteq \Sigma^\omega$, respectively. Then, $\mathcal{L}_\Sigma$ and $\mathcal{I}_\Sigma$ are non-conflicting. Moreover, the modular IO-system $\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ is ω-controllable w.r.t. $(U_\mathrm{p} \,\dot\cup\, U_\mathrm{l},\ \mathrm{clo}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma))$.

Proof The claim is verified by the same line of thought as in the proof of Proposition 3. An explicit proof is provided in the Appendix. □

4.3 Propagation of plant properties in the modular IO-system

We are now in the position, to state the main theorem of this section: the external behaviour of a modular IO-system again satisfies the IO-plant properties P1–P3.

Theorem 3 Given a modular IO-system, consisting of plant components $\mathcal{L}_n \subseteq \Sigma_n^\omega$, $n = 1, \ldots, m$, and an IO-environment $\mathcal{I} \subseteq \Sigma_{\mathrm{el}}^\omega$, denote the external behaviour $\mathcal{L}_{\mathrm{pl}} := \mathrm{p}_{\mathrm{pl}}^\omega\,(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$. If $\mathcal{L}_n \subseteq \Sigma_n^\omega$, $n = 1, \ldots, m$ are non-anticipating IO-plants, then so is $\mathcal{L}_{\mathrm{pl}}$.

Proof Regarding the event ordering P1, we observe that the modular IO-system satisfies $\mathcal{L}_{\mathrm{pl}} \subseteq \mathrm{p}_{\mathrm{pl}}^\omega((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega = ((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{l}U_\mathrm{l})^*)^\omega$. Regarding locally free inputs P2, we first show that $\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ possesses locally free inputs U_p and U_l. Pick $s, s' \in \mathrm{pre}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$ and $\mu, \mu' \in U_\mathrm{p}$, as well as $\nu, \nu' \in U_\mathrm{l}$ such that $s\mu, s'\nu \in \mathrm{pre}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$. By the locally free inputs U_p and U_l of $\mathrm{pre}\,\mathcal{L}_\Sigma$ and $\mathrm{pre}\,\mathcal{I}_\Sigma$ we obtain $s\mu', s'\nu' \in (\mathrm{pre}\,\mathcal{L}_\Sigma) \cap (\mathrm{pre}\,\mathcal{I}_\Sigma)$, and, referring to non-conflictingness from Proposition 6, $s\mu', s'\nu' \in \mathrm{pre}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$. Thus, $\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ indeed possesses locally free inputs U_p and U_l, which are retained under projection to Σ_{pl}. We are left to verify non-anticipation P3. Pick any $s \in \mathrm{pre}\,\mathcal{L}_{\mathrm{pl}} = \mathrm{pre}\,\mathrm{p}_{\mathrm{pl}}^\omega\,(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$. Then, there exists

$t \in \mathrm{pre}\,(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$ such that $\mathrm{p}_{\mathrm{pl}}t = s$. Recall from Proposition 6 that $\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ is ω-controllable w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{l},\ \mathrm{clo}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma))$. Thus, we can choose $\mathcal{W}_t \subseteq \mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ such that $t \in \mathrm{pre}\,\mathcal{W}_t$, and $\mathrm{pre}\,\mathcal{W}_t$ is controllable w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{l},\ \mathrm{pre}\ \mathrm{clo}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma))$, and $\mathcal{W}_t$ is closed. As a candidate to establish P3, let $\mathcal{V}_s := \mathrm{p}^\omega_{\mathrm{pl}}\mathcal{W}_t \subseteq \mathrm{p}^\omega_{\mathrm{pl}}\mathcal{L}_{\mathrm{pl}}$ and observe $s = \mathrm{p}_{\mathrm{pl}}t \in \mathrm{p}_{\mathrm{pl}}\,\mathrm{pre}\,\mathcal{W}_t = \mathrm{pre}\,\mathcal{V}_s$. Controllability and closedness of $\mathcal{V}_s$ follow as in the proof of Theorem 1, and we obtain that $\mathcal{L}_{\mathrm{pl}}$ is ω-controllable w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{l},\ \mathrm{clo}\,\mathcal{L}_{\mathrm{pl}})$. □

Conclusion. In this section, we proved that the IO-plant properties P1–P3 are retained under the proposed modular component composition. As our main result in this paper, Theorems 1–3 formally justify a hierarchical and modular controller design by alternating controller synthesis, closed-loop composition, abstraction and component composition, as illustrated by Figures 2 and 3 and demonstrated by an example in the next section.

5 Example

We apply the proposed approach to the control of a transport system. It consists of a number of conveyor belts arranged next to each other, with the specified behaviour to transport workpieces from the left to the right; see Figure 4. The example demon-

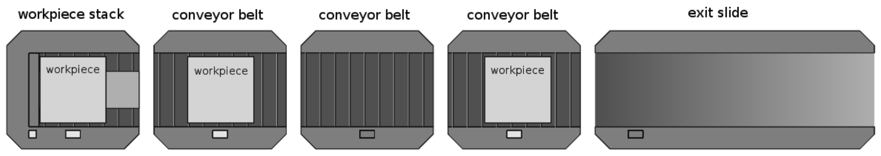


Fig. 4 Conveyor belt example

strates the use of separate models for components and their environments, the technical plant requirements P1–P3, and the use of specifications as abstractions. As indicated in the preliminaries, we use deterministic Büchi automata for the representation of the respective ω-languages. Recall that a Büchi automaton accepts those infinite executions that infinitely often pass marked states.

5.1 Low-level controller design

To begin with, we derive a model for an individual conveyor belt by referring to its physical configuration; see Figure 5. The *actuator events* `bon` and `boff` turn the belt motor on and off, respectively. The *sensor events* `wpar` and `wplv` indicate that a workpiece arrives at the sensor or leaves the sensor, respectively. Actuator and sensor events correspond to edges on the digital signals used to physically control the conveyor belt component. For possible interaction with components placed next to the conveyor belt, we define the events `enter` and `exit` for a workpiece to enter from the left or to exit to the right, respectively. Provided that the belt motor is on, `enter` is eventually acknowledged by the sensor event `wpar`. In order to also trace the `exit` event, we require that some component with a sensor is located on the right-hand side of the conveyor belt. We then associate workpiece arrival on the neighbouring component with `exit`.

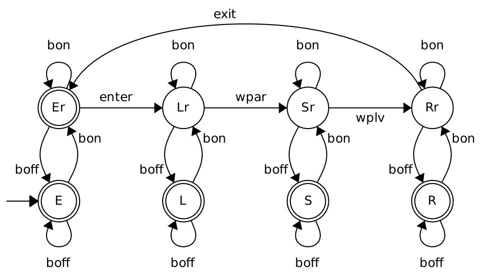
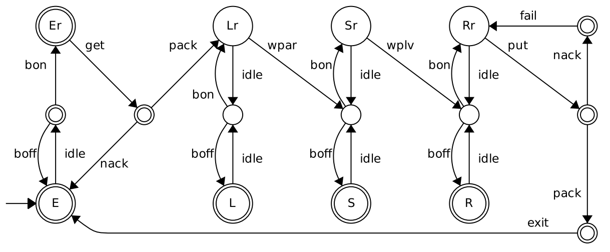


Fig. 5 Conveyor belt, physical model

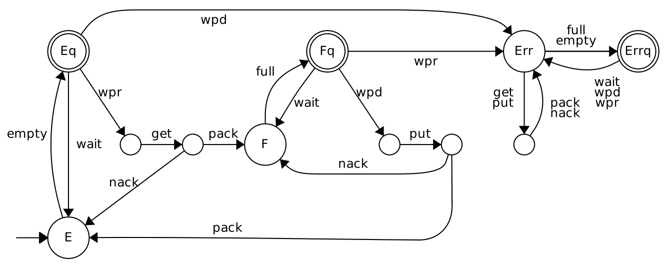
The physical model can be transformed to an IO-plant by replacing the occurrence of individual events by pairs, of each one input event and one output event; see Figure 6. For this purpose, the additional output event `idle` is introduced to formally provide feedback when no sensor event occurred. The events `enter` and `exit` are replaced by pairs of request and acknowledgement. Here, `enter` is modelled by `get` to indicate the attempt to get a workpiece from the left, which may succeed or fail, indicated by a subsequent `pack` or `nack`, respectively. Likewise, `exit` is modelled by `put`, followed by `pack` or `nack`. The event ordering P1 and locally free inputs P2 are readily verified on a per state basis. Regarding non-anticipation P3, we have computed the supremal ω-controllable sublanguage of the conveyor belt behaviour w.r.t. its topological closure and have verified that the result equals again the conveyor belt behaviour. Thus, P3 is indeed satisfied and our model is an IO-plant. Referring to the physical model, this was the expected outcome: the present eventuality properties do not restrict the applicable actuator events, neither locally nor on the infinite time axis.



$U_\mathrm{p} = \{\texttt{bon}, \texttt{boff}\}, \quad Y_\mathrm{p} = \{\texttt{idle}, \texttt{wpar}, \texttt{wplv}, \texttt{exit}, \texttt{fail}\}$
$U_\mathrm{e} = \{\texttt{pack}, \texttt{nack}\}, \quad Y_\mathrm{e} = \{\texttt{get}, \texttt{put}\}$

Fig. 6 Conveyor belt, IO-plant model

In order to state a control objective, we introduce the operator output events `full` and `empty` to indicate whether or not a workpiece is known to be present, and the operator input events `wpr` and `wpd` for the request and delivery of a workpiece, respectively. The intended semantics of the newly introduced events is defined by relating them to the environment events via a specification automaton; see Figure 7. Note that the specification automaton exclusively refers to the alphabet Σ_{ce} and that it is left to the synthesis procedure to figure out how to drive the plant by interleaving events from Σ_{p}. Technically, the depicted automaton realises the projection $\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{E}$ of the formal specification $\mathcal{E}$. In particular, there is no need to compute $\mathrm{p}^{\omega}_{\mathrm{ce}}\mathcal{E}$ from $\mathcal{E}$, avoiding a potentially exponential growth in the state count. The event ordering required by condition E1 is again verified on a per-state basis. The liveness requirement to eventually provide feedback to the operator is confirmed by inspecting all strongly connected components without an Y_{c} event and by verifying them not to include a marked state. Regarding locally free inputs E2, the error states `Err`, `Errq` have been introduced to obtain a locally free U_{c}. For a human operator, a more specific error behaviour would have been preferable. However, for the purpose of a subsequent design stage in a hierarchical control system, the proposed error behaviour will render the error states unreachable for any sensible high-level specification.



$U_{\mathrm{c}} = \{\texttt{wait}, \texttt{wpd}, \texttt{wpr}\}$, $Y_{\mathrm{c}} = \{\texttt{empty}, \texttt{full}\}$
$U_{\mathrm{e}} = \{\texttt{pack}, \texttt{nack}\}$, $Y_{\mathrm{e}} = \{\texttt{get}, \texttt{put}\}$

Fig. 7 Specification for one individual conveyor belt

Y_{c}:	`empty, full`	operator feedback to indicate presence of a workpiece
U_{c}:	`wait, wpr, wpr`	operator command to wait, or to receive/deliver one workpiece
Y_{e}:	`get, put`	attempt to get/put a workpiece from/to the environment
U_{e}:	`pack, nack`	acknowledgement of recent `get`/`put`
Y_{p}:	`idle, wpar, wplv`	plant sensors with dummy `idle` if nothing else is to report
U_{p}:	`bon, boff`	plant actuator to operate belt motor

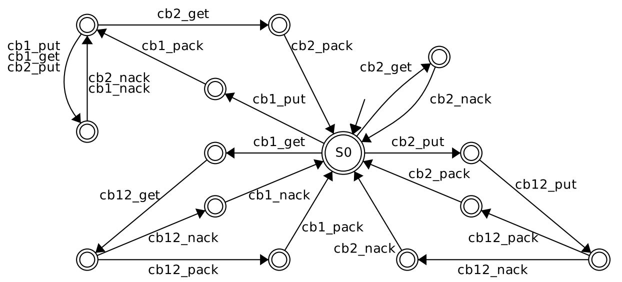
Table 1 List of events

As it turns out for this particular example, the specification fails to be relatively topologically closed w.r.t. the plant. Thus, we cannot compute a realisation of the supremal closed-loop behaviour. However, the approach outlined in Section 1.4 yields a non-empty closed loop that satisfies K1–K3, from which we obtain a controller that satisfies properties C1 and C2 and that amounts to 44 states. For the sake of simplicity, all conveyor belts from our transport system are assumed identical, and we can apply copies of the same controller to each belt. In order to formally end up with disjoint alphabets, we use the convention to prefix each event with an identifier of the respective component; e.g., `cb1_bon` to turn on the motor of the first conveyor belt `cb1`, counting from the left to the right.

5.2 Hierarchical and modular controller design

By the physical arrangement, the departure of a workpiece from one belt corresponds to the arrival of the workpiece at the next belt; e.g., `cb1_putr`–`cb1_pack` corresponds to `cb2_getl`–`cb2_pack`. This is accounted for by suitable environments, where, at this stage of our design, we compose groups of two conveyor belts each; see Figure 8 for an environment to synchronise the passing of workpieces between the two conveyor belts `cb1` and `cb2`. For the purpose of further compositions at subsequent stages of the hierarchical design, the proposed environment also introduces events to represent workpieces entering or leaving the group of the two units; these additional events are prefixed by `cb12_`. Adequate event-ordering I1 and locally free inputs I2 are verified by the same per-state inspection as P1 and P2. Topological closedness I3 is an immediate consequence of the marking.

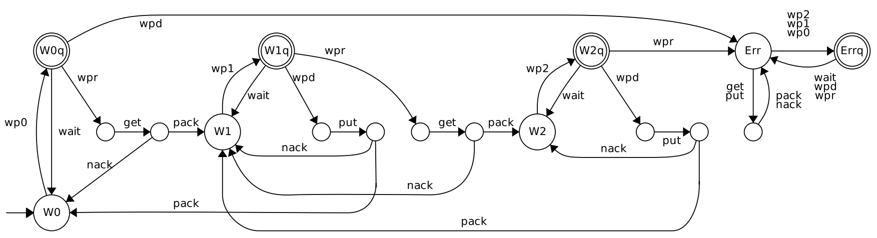
The actual behaviour of the two belts `cb1` and `cb2` under low-level control together with the environment is given by the composition discussed in Section 4. Referring to Theorem 1, the individual closed loops satisfy the IO-plant properties P1–P3, and, by Theorem 3, so does the composed system incl. the environment. In particular, the design of a high-level controller that coordinates the two conveyor belts w.r.t. their effect on the environment can be based on an abstraction. The latter can be constructed by replacing the low-level closed-loop behaviours with their respective specifications prior to the composition with the environment. The resulting automaton counts 172 states.



$U_\mathrm{l}=\{$`cb12_nack, cb12_pack`$\}$, $Y_\mathrm{l}=\{$`cb12_put, cb12_get`$\}$
$U_\mathrm{e}=\{$`cb1_pack,cb1_nack, cb2_pack, cb2_nack`$\}$,
$Y_\mathrm{e}=\{$`cb1_get, cb1_put, cb2_get, cb2_put`$\}$

Fig. 8 Environment for the two conveyor belts `cb1` and `cb2`

Informally, we want the high-level controller to coordinate a group of conveyor belts to behave as a single conveyor belt, except that the coordination should utilize the additional capacity as a buffer when required. Here, Figure 7 is re-interpreted as the behaviour of a transport unit with buffer capacity one, with obvious extensions to higher capacities. Technically, we introduce events {`wp0`, `wp1`, `wp` ...} as operator feedback and additional states to distinguish the number of present workpieces. For capacity two, the resulting specification is shown in Figure 9. To apply this automaton to the specific situation of the two left most conveyor belts `cb1` and `cb2`, the shown event labels are prefixed with `cb12_` and, thereby, match the environment model, Figure 8.



$U_c = \{\texttt{wait}, \texttt{wpd}, \texttt{wpr}\}$, $Y_c = \{\texttt{wp0}, \texttt{wp1}, \texttt{wp2}\}$
$U_e = \{\texttt{pack}, \texttt{nack}\}$, $Y_e = \{\texttt{get}, \texttt{put}\}$

Fig. 9 Specification for a transport unit with capacity two

We apply the approach outlined in Section 1.4 to obtain a high-level controller for the coordination of conveyor belts `cb1` and `cb2`. A state count of 1461 is observed. Thus, for the control of the two conveyor belts, we use two low-level controllers with 44 states each and one high-level controller with 1461 states. For a technical implementation, there is no need to apply a parallel composition to the three controllers. Thus, the overall state count $44 + 44 + 1461$ is considered an adequate indicator for the implementation complexity.

The control architecture extends to multiple levels when applied to multiple conveyor belts; see Figure 10. Assume that we have designed two controllers to operate one group of transport units each. To design the controller one level up the hierarchy, one refers to the two transport-unit specifications used for the design of each existing controller and to a third transport-unit specification for the overall behaviour. Each transport-unit specification has a state count linear to its capacity. The below Table 2 gives the state counts we observed for the resulting controllers for an overall capacity of up to eight workpieces. The table shows a polynomial growth of the state count for the controller w.r.t. the capacity. When restricting the left transport unit to capacity one, we experience a favourable linear growth of the state count. Hence, given a number of conveyor belts, one may start with two units and add one more per level of the hierarchy. With this strategy, we end up with as many controllers as conveyor belts for the hierarchical control architecture plus one low-level controller per conveyor belt; see left-hand side of Figure 10. The overall state count for this design strategy is quadratic in the number of conveyor belts; see Table 3. Alternatively, one may keep the tree balanced by forming groups with minimal difference in the number of components; see right-hand side of Figure 10.

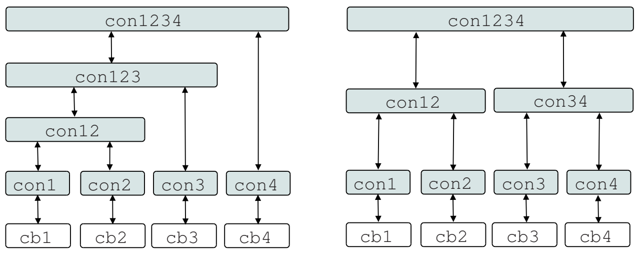


Fig. 10 Hierarchical control architectures for the conveyor belts `cb1`, `cb2`, `cb3` and `cb4`

overall capacity	2	3	4	4	5	5	6	6
capacity left/right	1/1	1/2	1/3	2/2	1/4	2/3	1/5	2/4
plant abs. state count	172	288	404	454	520	642	636	830
controller state count	1461	2479	3498	4041	4517	5727	5536	7417
overall capacity	6	7	7	7	8	8	8	8
capacity left/right	3/3	1/6	2/5	3/4	1/7	2/6	3/5	4/4
plant abs. state count	642	752	1018	1140	868	1206	1400	1450
controller state count	7954	6555	9104	10309	7574	10794	12667	13208

Table 2 Controller state count vs. capacity of two coordinated transport units

Our approach compares well with a monolithic controller design with an expected state count exponential in the number of conveyor belts. For a comparison, we used the plant model, Figure 5, and relabelled `enter` and `exit` events to obtain an overall model by parallel composition. As it turned out, we needed to include the stack feeder and the sink in the plant model for the synthesis of a nonblocking controller that enforces a collision avoidance specification. For a fair comparison, we count the stack feeder as one conveyor belt. In contrast to the hierarchical design, we did not need to specify the buffer capacity of the closed-loop system explicitly. The respective state counts for up to six conveyor belts are given in Table 3.

In order to validate the plant model and the formal specification, simulation experiments with four and eight conveyor belts were conducted. For the plant, the animated simulator FlexFact was used to simulate the continuous-time physical behaviour, including digital signals for the purpose of controller interconnection provided via a network interface. The hierarchy of controllers was interpreted by the discrete-event simulator from the software library libFAUDES. Here, events have been defined as edges on the digital signals accessible via the network interface of the plant simulation. The simulation experiment confirmed the expected closed-loop behaviour.

conveyor belts	1	2	3	4	5	6	7	8
one level per unit	44	1549	4072	7614	12175	17755	24354	31972
balanced hierarchy	44	1549	4072	7139	11348	16105	21520	27486
monolithic design	42	152	816	4290	23840	137110	–	–

Table 3 Controller state count vs. number of conveyor belts for different design strategies

Conclusion

In this paper, we discussed a closed-loop configuration with external signals, where plant and controller dynamics are represented as not necessarily topologically closed ω-languages. Based on Willems' notion of input-output systems, we identified the requirements P1–P3 for the plant behaviour, such that controller synthesis can be based on an abstraction while maintaining specified liveness and safety properties for the actual closed-loop. We have shown that the requirements P1–P3 are preserved under closed-loop composition, and, hence, that the closed-loop can again serve as a plant model. This leads to a hierarchical control architecture, in which we repeatedly design a controller, form the closed-loop and use the specification as an abstraction for the subsequent controller design.

To complement the "vertical" composition of controllers by a "horizontal" composition of plant components, we introduce a specialized shuffle-product that again preserves the requirements P1–P3. Here, dependencies between individual plant components are represented as environment models. Since both proposed forms of system composition retain the same three requirements, they can be freely combined to obtain a hierarchical and modular control architecture, where each controller is designed to addresses the coordination of a group of closed-loop systems on the level below.

Although not formally guaranteed, we expect computational benefits for relevant applications. In the alternation of controller design, system combination and abstraction, the former two stages tend to increase the number of states while that latter reduces the state count to represent the purpose of the design as opposed to the means of how it is achieved. The transport system example demonstrates, that the additional structure required by our control architecture introduces a certain penalty for the synthesis of the individual controllers, however, the overall complexity is observed to be polynomial in the number of plant components, and, thus, compares well against a monolithic design.

References

Baier C, Kwiatkowska M (2000) On topological hierarchies of temporal properties. Fundamenta Informaticae 41:259–294

Baier C, Moor T (2012) A hierarchical control architecture for sequential behaviours. Workshop on Discrete Event Systems 2012 pp 259–264

Cassandras CG, Lafortune S (2008) Introduction to Discrete Event Systems, 2nd edn. Springer

da Cunha AEC, Cury JER, Krogh BH (2002) An assume-guarantee reasoning for hierarchical coordination of discrete event systems. Workshop on Discrete Event Systems 2006 pp 75–80

Feng L, Wonham W (2008) Supervisory control architecture for discrete-event systems. IEEE Transactions on Automatic Control 53(6):1449–1461

Hopcroft JE, Ullman JD (1979) Introduction to Automata Theory, Languages and Computation. Addison-Wesley, Reading

Kumar R, Garg V, Marcus SI (1992) On supervisory control of sequential behaviors. IEEE Transactions on Automatic Control 37(12):1978 –1985

Kupferman O, Vardi MY (2000) Synthesis with incomplete informatio. In: Advances in Temporal Logic, Kluwer Academic Publishers, pp 109–127

Leduc RJ, Lawford M, Wonham WM (2005) Hierarchical interface-based supervisory control - part ii: Parallel case. IEEE Transactions on Automatic Control 50(9):1336–1348

Lin F, Wonham WM (1988) On observability of discrete-event systems. Information Sciences 44:173–198

Manna Z, Pnueli A (1990) A hierarchy of temporal properties. Proc 9th ACM Symposium on Principles of Distributed Computing pp 377–408

Moor T, Raisch J, Davoren JM (2003) Admissibility criteria for a hierarchical design of hybrid control systems. Proc IFAC Conference on the Analysis and Design of Hybrid Systems (ADHS'03') pp 389–394

Moor T, Schmidt K, Wittmann T (2011) Abstraction-based control for not necessarily closed behaviours. Proc 18th IFAC World Congress pp 6988–6993

Moor T, Baier C, Yoo TS, Lin F, Lafortune S (2012) On the computation of supremal sublanguages relevant to supervisory control. Workshop on Discrete Event Systems 2012 pp 175–180

Mukund M (1996) Finite-state automata on infinite inputs. Internal Report TCS-96-2, SPIC Mathematical Institute

Perk S, Moor T, Schmidt K (2006) Hierarchical discrete event systems with inputs and outputs. Workshop on Discrete Event Systems 2006 pp 427–432

Perk S, Moor T, Schmidt K (2008) Controller synthesis for an i/o-based hierarchical system architecture. Workshop on Discrete Event Systems 2008 pp 474–479

Ramadge PJ (1989) Some tractable supervisory control problems for discrete-event systems modeled by buchi automata. IEEE Transactions on Automatic Control 34(1):10–19

Ramadge PJ, Wonham WM (1987) Supervisory control of a class of discrete event processes. SIAM J Control Optim 25(1):206–230

Ramadge PJ, Wonham WM (1989) The control of discrete event systems. Proc IEEE 77(1):81–98

Schmidt K, Moor T, Perk S (2008) Nonblocking hierarchical control of decentralized discrete event systems. IEEE Transactions on Automatic Control 53(10):2252–2265

Thistle JG, Lamouchi HM (2009) Effective control synthesis for partially observed discrete-event systems. SIAM J Control and Optimization 48(3):1858–1887

Thistle JG, Wonham W (1992) Control of omega-automata, church's problem, and the emptiness problem for tree omega-automata. Computer Science Logic 626:367–381

Thistle JG, Wonham WM (1994) Supervision of infinite behavior of discrete-event systems. SIAM J Control and Optimization 32(4):1098–1113

Thomas W (1990) Automata on infinite objects. Handbook of theoretical computer science (vol B), The MIT Press, Cambridge, MA pp 133–191

Willems JC (1991) Paradigms and puzzles in the theory of dynamical systems. IEEE TAC 36(3):259–294

Wong KC, Wonham WM (1996) Hierarchical control of discrete-event systems. Discrete Event Dynamic Systems: Theory and Applications 6(3):241–273

Appendix

This section collects a number of technical lemmata to support the core arguments

Proof of Proposition 5 [See also the below Lemma 7]

Given non-anticipating IO-plant components, we have to show that the full IO-shuffle

$$\mathcal{L}_\Sigma = \;(\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{L}_{\mathrm{IO}} \cup \operatorname{pre}\mathcal{L}_{\mathrm{IO}})) \;\cap\; (\operatorname{clo}((Y_{\mathrm{p}}U_{\mathrm{p}})^*(Y_{\mathrm{e}}U_{\mathrm{e}})^*(Y_{\mathrm{e}}Y_{\mathrm{l}}U_{\mathrm{l}}U_{\mathrm{e}})^*)^\omega)$$

is ω-controllable w.r.t. $(\Sigma_{\mathrm{l}} \dot\cup U_{\mathrm{p}} \dot\cup U_{\mathrm{e}}, \operatorname{clo}\mathcal{L}_\Sigma)$. From the definition of $\mathcal{L}_\Sigma$, we note that $\operatorname{pre}\mathcal{L}_\Sigma \subseteq \operatorname{pre}\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{L}_{\mathrm{IO}} \cup \operatorname{pre}\mathcal{L}_{\mathrm{IO}}) = \mathrm{p}_{\mathrm{pe}}^{-1}\operatorname{pre}\mathcal{L}_{\mathrm{IO}}$. Pick an arbitrary string $s \in \operatorname{pre}\mathcal{L}_\Sigma$, let $r := \mathrm{p}_{\mathrm{pe}}s$, and observe that $r \in \operatorname{pre}\mathcal{L}_{\mathrm{IO}}$. Referring to Proposition 4, $\mathcal{L}_{\mathrm{IO}}$ is a non-anticipating IO-plant, i.e., $\mathcal{L}_{\mathrm{IO}}$ is ω-controllable w.r.t. $(U_{\mathrm{p}} \dot\cup U_{\mathrm{e}}, \operatorname{clo}\mathcal{L}_{\mathrm{IO}})$. Thus, we can choose $\mathcal{W}_r \subseteq \mathcal{L}_{\mathrm{IO}}$, such that $r \in \operatorname{pre}\mathcal{W}_r$, and $\operatorname{pre}\mathcal{W}_r$ is controllable w.r.t. $(U_{\mathrm{p}} \dot\cup U_{\mathrm{e}},\ \operatorname{pre}\mathcal{L}_{\mathrm{IO}})$, and $\mathcal{W}_r$ is relatively closed w.r.t. $\operatorname{clo}\mathcal{L}_{\mathrm{IO}}$. Recall that relative closedness w.r.t. a closed language implies closedness. In particular, $\mathcal{W}_r$ is closed. To establish the non-anticipating property for $\mathcal{L}_\Sigma$, consider the candidate

$$\mathcal{V}_s := (\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{W}_r \cup \operatorname{pre}\mathcal{W}_r)) \cap (\operatorname{clo}((Y_{\mathrm{p}}U_{\mathrm{p}})^*(Y_{\mathrm{e}}U_{\mathrm{e}})^*(Y_{\mathrm{e}}Y_{\mathrm{l}}U_{\mathrm{l}}U_{\mathrm{e}})^*)^\omega).$$

Clearly, $\mathcal{V}_s \subseteq \mathcal{L}_\Sigma$ and $\operatorname{pre}\mathcal{V}_s \subseteq \operatorname{pre}\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{W}_r \cup \operatorname{pre}\mathcal{W}_r) = \mathrm{p}_{\mathrm{pe}}^{-1}\operatorname{pre}\mathcal{W}_r$. Further, we have $s \in \operatorname{pre}\mathcal{V}_s$, since $\mathrm{p}_{\mathrm{pe}}s = r \in \operatorname{pre}\mathcal{W}_r$ and $s \in \operatorname{pre}((Y_{\mathrm{p}}U_{\mathrm{p}})^*(Y_{\mathrm{e}}U_{\mathrm{e}})^*(Y_{\mathrm{e}}Y_{\mathrm{l}}U_{\mathrm{l}}U_{\mathrm{e}})^*)^\omega$. To show controllability, pick an arbitrary string $\hat{s} \in \operatorname{pre}\mathcal{V}_s$ and $\sigma \in \Sigma_{\mathrm{l}} \dot\cup U_{\mathrm{p}} \dot\cup U_{\mathrm{e}}$ such that $\hat{s}\sigma \in \operatorname{pre}\mathcal{L}_\Sigma$. In particular, $\mathrm{p}_{\mathrm{pe}}\hat{s} \in \mathrm{p}_{\mathrm{pe}}\mathrm{p}_{\mathrm{pe}}^{-1}\operatorname{pre}\mathcal{W}_r = \operatorname{pre}\mathcal{W}_r$ and $\mathrm{p}_{\mathrm{pe}}(\hat{s}\sigma) \in \mathrm{p}_{\mathrm{pe}}\operatorname{pre}\mathcal{L}_\Sigma \subseteq \operatorname{pre}\mathcal{L}_{\mathrm{IO}}$. Controllability of $\operatorname{pre}\mathcal{W}_r$ w.r.t. $\operatorname{pre}\mathcal{L}_{\mathrm{IO}}$ implies that $\mathrm{p}_{\mathrm{pe}}(\hat{s}\sigma) \in \operatorname{pre}\mathcal{W}_r$. In addition, there exists $u \in \Sigma_{\mathrm{pe}}^\omega$, such that $\mathrm{p}_{\mathrm{pe}}(\hat{s}\sigma)u \in \mathcal{W}_r$. We choose $w \in \Sigma^\omega$ such that $\hat{s}\sigma w \in \operatorname{clo}((Y_{\mathrm{p}}U_{\mathrm{p}})^*(Y_{\mathrm{e}}U_{\mathrm{e}})^*(Y_{\mathrm{e}}Y_{\mathrm{l}}U_{\mathrm{l}}U_{\mathrm{e}})^*)^\omega$ and $\mathrm{p}_{\mathrm{pe}}^\omega(\hat{s}\sigma w) = \mathrm{p}_{\mathrm{pe}}(\hat{s}\sigma)u$. Note that $\hat{s}\sigma w \in \mathcal{V}_s$ and, hence, $\hat{s}\sigma \in \operatorname{pre}\mathcal{V}_s$. Finally, we have to establish relative closedness of $\mathcal{V}_s$ w.r.t. $\operatorname{clo}\mathcal{L}_\Sigma$. Since $\operatorname{clo}\mathcal{L}_\Sigma$ is a closed superset of $\mathcal{V}_s$, relative closedness of $\mathcal{V}_s$

and closedness of $\mathcal{V}_s$ are equivalent. Recall that the intersection of two topologically closed languages is again topologically closed, to obtain closedness of $\mathcal{V}_s$ by

$$\begin{aligned}\mathcal{V}_s &= (\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{W}_r \cup \mathrm{pre}\,\mathcal{W}_r)) \cap (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\\ &= (\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathrm{clo}\,\mathcal{W}_r \cup \mathrm{pre}\,\mathcal{W}_r)) \cap (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\\ &= (\mathrm{clo}\,\mathrm{p}_{\mathrm{pe}}^{-\omega}\mathcal{W}_r) \cap (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\,.\end{aligned}$$

□

Proof of Proposition 6

Given non-anticipating IO-plant components, consider the full IO-shuffle and the full environment

$$\begin{aligned}\mathcal{L}_\Sigma &= (\,\mathrm{p}_{\mathrm{pe}}^{-\omega}(\mathcal{L}_{\mathrm{IO}} \cup \mathrm{pre}\,\mathcal{L}_{\mathrm{IO}}))\ \cap\ (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\,,\\ \mathcal{I}_\Sigma &= (\,\mathrm{p}_{\mathrm{el}}^{-\omega}(\mathcal{I} \cup \mathrm{pre}\,\mathcal{I}))\ \cap\ (\mathrm{clo}((Y_\mathrm{p}U_\mathrm{p})^*(Y_\mathrm{e}U_\mathrm{e})^*(Y_\mathrm{e}Y_\mathrm{l}U_\mathrm{l}U_\mathrm{e})^*)^\omega)\,,\end{aligned}$$

respectively. We have to show that $\mathcal{L}_\Sigma$ and $\mathcal{I}_\Sigma$ are non-conflicting and that $\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ is ω-controllable w.r.t. $(U_\mathrm{p}\,\dot{\cup}\,U_\mathrm{l},\ \mathrm{clo}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma))$. We begin with ω-controllability and construct a suitable candidate $\mathcal{V}_s \subseteq \mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ for an arbitrarily chosen $s \in \mathrm{pre}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$. Referring to Proposition 5, there exists $\mathcal{W}_s \subseteq \mathcal{L}_\Sigma$, such that $s \in \mathrm{pre}\,\mathcal{W}_s$, $\mathrm{pre}\,\mathcal{W}_s$ is controllable w.r.t. $(\Sigma_\mathrm{l}\dot{\cup}U_\mathrm{p}\dot{\cup}U_\mathrm{e},\ \mathrm{pre}\,\mathcal{L}_\Sigma)$, and $\mathcal{W}_s$ is relatively closed w.r.t. $\mathrm{clo}\,\mathcal{L}_\Sigma$. In particular, $\mathcal{W}_s$ is closed. To establish ω-controllability of $\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ w.r.t. $\mathrm{clo}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$, consider the candidate $\mathcal{V}_s := \mathcal{W}_s \cap \mathcal{I}_\Sigma$. Clearly, $\mathcal{V}_s \subseteq \mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$. Furthermore, $\mathcal{V}_s = \mathcal{W}_s \cap \mathcal{I}_\Sigma = (\mathrm{clo}\,\mathcal{W}_s) \cap (\mathrm{clo}\,\mathcal{I}_\Sigma) \supseteq \mathrm{clo}\,\mathcal{V}_s$, i.e., $\mathcal{V}_s$ is closed and, thus, relatively closed w.r.t. any superset. To show controllability of $\mathrm{pre}\,\mathcal{V}_s$ w.r.t. $\mathrm{pre}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$, we pick $r \in \mathrm{pre}(\mathcal{W}_s \cap \mathcal{I}_\Sigma)$ and $\sigma \in U_\mathrm{l}\,\dot{\cup}\,U_\mathrm{p}$ such that $r\sigma \in \mathrm{pre}\ \mathrm{clo}(\,\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma\,) \subseteq (\mathrm{pre}\,\mathcal{L}_\Sigma)\ \cap\ (\mathrm{pre}\,\mathcal{I}_\Sigma)$. By controllability of $\mathrm{pre}\,\mathcal{W}_s$, it follows that $r\sigma \in (\mathrm{pre}\,\mathcal{W}_s)$. The locally free input U_l of $\mathcal{I}$ and the inverse projection in the definition of $\mathcal{I}_\Sigma$ imply $r\sigma \in (\mathrm{pre}\,\mathcal{I}_\Sigma)$ for either case $\sigma \in U_\mathrm{l}$ or $\sigma \in \dot{\cup}U_\mathrm{p}$, respectively. So far, we have $r\sigma \in \mathrm{pre}(\mathcal{W}_s)\ \cap\ (\mathrm{pre}\,\mathcal{I}_\Sigma)$. To establish $r\sigma \in \mathrm{pre}(\mathcal{W}_s \cap \mathcal{I}_\Sigma)$, observe that each event in Σ is either uncontrollable for $\mathrm{pre}\,\mathcal{W}_s$ or a locally free input for $\mathrm{pre}\,\mathcal{I}_\Sigma$. Thus, starting with $r_0 = r\sigma$, we can construct an unbounded sequence $(r_n) \subseteq (\mathrm{pre}\,\mathcal{W}_s) \cap (\mathrm{pre}\,\mathcal{I}_\Sigma)$ with limit $w := \lim(r_n) \in (\mathrm{clo}\,\mathcal{W}_s) \cap (\mathrm{clo}\,\mathcal{I}_\Sigma) = \mathcal{W}_s \cap \mathcal{I}_\Sigma$. Hence, $r\sigma \in \mathrm{pre}(\mathcal{W}_s \cap \mathcal{I}_\Sigma)$. This concludes the proof of ω-controllability. For non-conflictingness, pick an arbitrary $s \in (\mathrm{pre}\,\mathcal{L}_\Sigma)\ \cap\ (\mathrm{pre}\,\mathcal{I}_\Sigma)$. In our argument, we refer to the same candidate $\mathcal{W}_s$ as used in the first part of this proof. In particular, we have $s \in (\mathrm{pre}\,\mathcal{W}_s)\ \cap\ (\mathrm{pre}\,\mathcal{I}_\Sigma)$ and we can, as above, start with $s_0 = s$ to construct an unbounded sequence $(s_n) \subseteq (\mathrm{pre}\,\mathcal{W}_s) \cap (\mathrm{pre}\,\mathcal{I}_\Sigma)$ by successively appending events that are either uncontrollable for $\mathrm{pre}\,\mathcal{W}_s$ or a locally free input for $\mathrm{pre}\,\mathcal{I}_\Sigma$. Consequently, we obtain $w := \lim(s_n) \in (\mathrm{clo}\,\mathcal{W}_s) \cap (\mathrm{clo}\,\mathcal{I}_\Sigma) = \mathcal{W}_s \cap \mathcal{I}_\Sigma \subseteq \mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma$ and thus $s \in \mathrm{pre}(\mathcal{L}_\Sigma \cap \mathcal{I}_\Sigma)$. □

Proof of Lemma 1 [see also (Baier and Moor, 2012), Lemma 14]

This lemma supports the proof of Theorem 2, where for $\mathcal{L} \subseteq \mathcal{L}'$ a solution $\mathcal{H}$ of the control problem $(\Sigma,\ \mathcal{L}',\ \mathcal{E})$ is verified to also solve $(\Sigma,\ \mathcal{L},\ \mathcal{E})$. For the lemma, we have to show for $\mathcal{V}' \subseteq \mathcal{L}'_\Sigma \cap \mathcal{H}_\Sigma$ that, if $\mathrm{pre}\,\mathcal{V}'$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathcal{L}'_\Sigma)$, and if $\mathcal{V}'$ is relatively closed w.r.t. $\mathcal{L}'_\Sigma$, then $\mathcal{L}_\Sigma$ and $\mathcal{V}'$ are non-conflicting, i.e., that $\operatorname{pre}(\mathcal{L}_\Sigma \cap \operatorname{pre}\mathcal{V}') = (\operatorname{pre}\mathcal{L}_\Sigma) \cap (\operatorname{pre}\mathcal{V}')$. Pick an arbitrary string $s \in (\operatorname{pre}\mathcal{L}_\Sigma) \cap (\operatorname{pre}\mathcal{V}')$. Referring to Lemma 8, we represent $\mathcal{L}_\Sigma$ as $\mathcal{L}_\Sigma = \cup_{a\in A}\mathcal{L}_a$ with $\mathcal{L}_a$ satisfying conditions (i) and (ii). In particular, there exists $a \in A$ with $s \in \operatorname{pre}\mathcal{L}_a \subseteq \mathcal{L}_\Sigma \subseteq \mathcal{L}'_\Sigma$. To extend $s \in (\operatorname{pre}\mathcal{L}_a) \cap (\operatorname{pre}\mathcal{V}')$ by one event, pick σ such that $s\sigma \in \operatorname{pre}\mathcal{L}_a$. If $\sigma \in \Sigma_{\mathrm{uc}}$, then controllability of $\operatorname{pre}\mathcal{V}'$ implies $s\sigma \in \operatorname{pre}\mathcal{V}'$, and we end up with $s\sigma \in (\operatorname{pre}\mathcal{L}_a) \cap (\operatorname{pre}\mathcal{V}')$. If, on the other hand, $\sigma \in U_{\mathrm{p}} \dot\cup Y_{\mathrm{c}}$, we obtain by Lemma 8, condition (i), that $s(U_{\mathrm{p}} \dot\cup Y_{\mathrm{c}}) \subseteq \operatorname{pre}\mathcal{L}_a$. Referring to the event ordering in the definition of $\mathcal{L}_\Sigma$, we decompose $s = r\nu$ with $\nu \in U_{\mathrm{c}} \dot\cup Y_{\mathrm{p}}$. Again by the definition of $\mathcal{L}_\Sigma$, now using $r\nu \in \operatorname{pre}\mathcal{V}' \subseteq \operatorname{pre}\mathcal{L}'_\Sigma$, we obtain the existence of $\sigma \in U_{\mathrm{p}} \dot\cup Y_{\mathrm{c}}$ such that $s\sigma \in \operatorname{pre}\mathcal{V}'$ and, thus, conclude with $s\sigma \in (\operatorname{pre}\mathcal{L}_a) \cap (\operatorname{pre}\mathcal{V}')$. Repeatedly extending s, we construct a strictly monotone sequence $(s_n) \subseteq (\operatorname{pre}\mathcal{L}_a) \cap (\operatorname{pre}\mathcal{V}')$ with limit $w := \lim(s_n) \in (\operatorname{clo}\mathcal{L}_a) \cap (\operatorname{clo}\mathcal{V}')$ and $s = s_0 < w$. Since $\mathcal{L}_a$ is closed, we obtain $w \in \mathcal{L}_a$ to observe $w \in \mathcal{L}_a \cap (\operatorname{clo}\mathcal{V}') \subseteq \mathcal{L}_\Sigma \cap \mathcal{L}'_\Sigma \cap (\operatorname{clo}\mathcal{V}') = \mathcal{L}_\Sigma \cap \mathcal{V}'$. In particular, $s \in \operatorname{pre}(\mathcal{L}_\Sigma \cap \mathcal{V}')$. □

Lemma 2 Consider $\Sigma_{\mathrm{uc}} \subseteq \Sigma$ and $\mathcal{L}, \mathcal{H} \subseteq \Sigma^\omega$. If $\mathcal{H}$ is ω-controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathcal{L})$, then $\operatorname{pre}\mathcal{H}$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \operatorname{pre}\mathcal{L})$, and $\mathcal{L}$ and $\mathcal{H}$ are non-conflicting.

Proof Pick any $s \in \operatorname{pre}\mathcal{H}$ and $\sigma \in \Sigma_{\mathrm{uc}}$, such that $s\sigma \in \operatorname{pre} L$. In particular, this implies $s \in \operatorname{pre}\mathcal{L}$ and we can choose $\mathcal{V}_s \subseteq \mathcal{L}\cap\mathcal{H}$, $s \in \operatorname{pre}\mathcal{V}_s$, according to conditions (i) and (ii) from Definition 1. Here controllability (i) implies $s\sigma \in \operatorname{pre}\mathcal{V}_s \subseteq \operatorname{pre}(\mathcal{L} \cap \mathcal{H}) \subseteq \operatorname{pre}\mathcal{H}$. This concludes the proof of controllability for $\operatorname{pre}\mathcal{H}$. For non-conflictingness, pick an arbitrary $s \in (\operatorname{pre}\mathcal{H}) \cap (\operatorname{pre}\mathcal{L})$ and again choose $\mathcal{V}_s \subseteq \mathcal{L}\cap\mathcal{H}$, $s \in \operatorname{pre}\mathcal{V}_s$, according to Definition 1. Clearly, $s \in \operatorname{pre}\mathcal{V}_s \subseteq \operatorname{pre}(\mathcal{H} \cap \mathcal{L})$. □

Lemma 3 Let $\Sigma_{\mathrm{uc}} \subseteq \Sigma$, $\Sigma_{\mathrm{o}} \subseteq \Sigma$, $\Sigma - \Sigma_{\mathrm{uc}} \subseteq \Sigma_{\mathrm{o}}$, $\mathcal{L} \subseteq \Sigma^\omega$, and consider a family of languages $\mathcal{H}_a \subseteq \Sigma^\omega$, $a \in A$, each one ω-admissible w.r.t. $(\Sigma_{\mathrm{uc}},\ \Sigma_{\mathrm{o}},\ \mathcal{L})$. Then, the union $\mathcal{H} := \cup_{a\in A}\mathcal{H}_a$ is ω-admissible, too.

Proof Pick an arbitrary prefix $s \in (\operatorname{pre}\mathcal{L}) \cap (\operatorname{pre}\mathcal{H})$. By $\operatorname{pre}\mathcal{H} = \cup_{a\in A}\operatorname{pre}\mathcal{H}_a$, we can choose $a \in A$ such that $s \in \operatorname{pre}\mathcal{H}_a$. Since $\mathcal{H}_a$ is considered ω-admissible, we can also choose $\mathcal{V}_s \subseteq \mathcal{L} \cap \mathcal{H}_a$, $s \in \operatorname{pre}\mathcal{V}_s$ to satisfy conditions (i)–(iii) from Definition 2. Clearly, $\mathcal{V}_s \subseteq \mathcal{L} \cap \mathcal{H}$, and we have established ω-admissibility of $\mathcal{H}$. □

Lemma 4 Let $\Sigma_{\mathrm{uc}} \subseteq \Sigma$, $\Sigma_{\mathrm{o}} \subseteq \Sigma$, $\Sigma - \Sigma_{\mathrm{uc}} \subseteq \Sigma_{\mathrm{o}}$, and consider $\mathcal{L} \subseteq \Sigma^\omega$ and $\mathcal{E} \subseteq \mathcal{L}$. Denote $\mathcal{K}^\Uparrow \subseteq \mathcal{E}$ the supremal ω-admissible sublanguage of $\mathcal{E}$. If $\mathcal{E}$ is relatively topologically closed w.r.t. $\mathcal{L}$, then so is $\mathcal{K}^\Uparrow$.

Proof Let $\mathcal{K} := (\operatorname{clo}\mathcal{K}^\Uparrow) \cap \mathcal{L}$, and observe $\mathcal{K}^\Uparrow \subseteq \mathcal{K} \subseteq (\operatorname{clo}\mathcal{E}) \cap \mathcal{L} = \mathcal{E}$. Moreover, recall that $\operatorname{pre}\operatorname{clo}\mathcal{K}^\Uparrow = \operatorname{pre}\mathcal{K}^\Uparrow$, to obtain $\operatorname{pre}\mathcal{K} \subseteq (\operatorname{pre}\operatorname{clo}\mathcal{K}^\Uparrow) \cap (\operatorname{pre}\mathcal{L}) = (\operatorname{pre}\mathcal{K}^\Uparrow) \cap (\operatorname{pre}\mathcal{L}) = \operatorname{pre}\mathcal{K}^\Uparrow$, and, hence, $\operatorname{pre}\mathcal{K} = \operatorname{pre}\mathcal{K}^\Uparrow$. To show that $\mathcal{K}$ is ω-admissible w.r.t. $(\Sigma_{\mathrm{uc}},\ \Sigma_{\mathrm{o}},\ \mathcal{L})$, pick an arbitrary $s \in (\operatorname{pre}\mathcal{L}) \cap (\operatorname{pre}\mathcal{K}) = (\operatorname{pre}\mathcal{L}) \cap (\operatorname{pre}\mathcal{K}^\Uparrow)$. In particular, we can choose $\mathcal{V}_s \subseteq \mathcal{K}^\Uparrow$, $s \in \operatorname{pre}\mathcal{V}_s$, according to conditions (i)–(iii) in Definition 2, and conclude that $\mathcal{K}$ is ω-admissible w.r.t. $(\Sigma_{\mathrm{uc}},\ \Sigma_{\mathrm{o}},\ \mathcal{L})$. Supremality of $\mathcal{K}^\Uparrow$ implies $\mathcal{K} \subseteq \mathcal{K}^\Uparrow$. □

Lemma 5 Let $\Sigma_{\mathrm{uc}} \subseteq \Sigma$, $\Sigma_{\mathrm{o}} \subseteq \Sigma$, $\Sigma - \Sigma_{\mathrm{uc}} \subseteq \Sigma_{\mathrm{o}}$, and consider $\mathcal{L} \subseteq \Sigma^{\omega}$ and $\mathcal{H} \subseteq \Sigma^{\omega}$. If $\mathcal{H}$ is ω-admissible w.r.t. $(\Sigma_{\mathrm{uc}},\ \Sigma_{\mathrm{o}},\ \mathcal{L})$, then $(\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$ is prefix-normal w.r.t. $(\Sigma_{\mathrm{o}},\ \mathrm{pre}\,\mathcal{L})$.

Proof Pick an arbitrary string $s \in (\,\mathrm{p}_{\mathrm{o}}^{-1}\mathrm{p}_{\mathrm{o}}(\,(\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})\,)\,) \cap \mathrm{pre}\,\mathcal{L}$. Then there exists $s' \in (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$ such that $\mathrm{p}_{\mathrm{o}}s' = \mathrm{p}_{\mathrm{o}}s$, and we choose $\mathcal{V}_{s'} \subseteq \mathcal{L} \cap \mathcal{H}$, $s' \in \mathrm{pre}\,\mathcal{V}_{s'}$, according to conditions (i)–(iii), Definition 2. Here, prefix-normality (ii) implies $s \in \mathrm{pre}\,\mathcal{V}_{s'}$. Together with $\mathrm{pre}\,\mathcal{V}_{s'} \subseteq \mathrm{pre}(\mathcal{L}\cap\mathcal{H}) \subseteq (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$, we obtain $s \in (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{H})$. □

Lemma 6 Let $\Sigma_{\mathrm{uc}} \subseteq \Sigma$, $\Sigma_{\mathrm{o}} \subseteq \Sigma$, $\Sigma - \Sigma_{\mathrm{uc}} \subseteq \Sigma_{\mathrm{o}}$, and consider $\mathcal{L} \subseteq \Sigma^{\omega}$ and $\mathcal{E} \subseteq \mathcal{L}$. Assume that $\mathcal{E}$ is relatively topologically closed w.r.t. $\mathcal{L}$, and that both $\mathcal{L}$ and $\mathcal{E}$ are represented as limits of regular $*$-languages $\mathcal{L} = \lim L$, $\mathcal{E} = \lim E$, where L is complete and $E = (\mathrm{pre}\,\mathcal{E}) \cap L$. Let $K^{\uparrow} \subseteq \Sigma^{*}$ denote the supremal sublanguage of $K \subseteq E$ that satisfies the requirements (L1)–(L5) given in (Moor et al, 2012):

(L1) K is complete,

(L2) K is controllable w.r.t. $(L,\ \Sigma_{\mathrm{uc}})$,

(L3) K is normal w.r.t. $(L,\ \Sigma_{\mathrm{o}})$,

(L4) $K \subseteq E$, and

(L5) K is relatively prefix-closed w.r.t. L.

Then $\lim K^{\uparrow}$ is the supremal ω-admissible sublanguage $\mathcal{K}^{\Uparrow}$ of $\mathcal{E}$.

Proof The prerequisite that $\mathcal{E}$ is relatively topologically closed implies that $\mathcal{K}^{\Uparrow}$ is relatively topologically closed, too; see Lemma 4. In particular, $\mathcal{K}^{\Uparrow}$ uniformly satisfies conditions (i)-(iii) in Definition 2 for all $s \in (\mathrm{pre}\,\mathcal{L}) \cap (\mathrm{pre}\,\mathcal{K})$. Therefore, $\mathcal{K}^{\Uparrow}$ is identical with the supremal sublanguage $\mathcal{K}$ of $\mathcal{E}$ that satisfies

(i) $\mathrm{pre}\,\mathcal{K}$ is controllable w.r.t. $(\Sigma_{\mathrm{uc}},\ \mathrm{pre}\,\mathcal{L})$;

(ii) $\mathrm{pre}\,\mathcal{K}$ is prefix-normal w.r.t. $(\Sigma_{\mathrm{o}},\ \mathrm{pre}\,\mathcal{L})$; and

(iii) $\mathcal{K}$ is relatively topologically closed w.r.t. $\mathcal{L}$.

Now consider an arbitrary K that satisfies (L1)–(L5) and denote $\mathcal{K} = \lim K$. By (L1), we have $\mathrm{pre}\,\mathcal{K} = \mathrm{pre}\,K$, and, $\mathcal{K}$ satisfies (i) and (ii) by (L2) and (L3), respectively. By (L4), we have $\mathcal{K} \subseteq \mathcal{E}$. Finally, $(\mathrm{clo}\,\mathcal{K}) \cap \mathcal{L} = (\lim \mathrm{pre}\,K) \cap (\lim L) = \lim(\,(\mathrm{pre}\,K) \cap L\,) = \lim K = \mathcal{K}$ is obtained by (L5), and we note that $\mathcal{K}$ satisfies (iii). Thus, we have $\lim K^{\uparrow} \subseteq \mathcal{K}^{\Uparrow}$. Vice versa, consider an arbitrary $\mathcal{K} \subseteq \mathcal{E}$ conform to (i)–(iii) and let $K = (\mathrm{pre}\,\mathcal{K}) \cap L$. Obviously, this choice satisfies (L5). To verify (L1)–(L4), we first establish that $\mathrm{pre}\,K = \mathrm{pre}\,\mathcal{K}$. Clearly, $\mathrm{pre}\,K \subseteq \mathrm{pre}\,\mathcal{K}$. For the converse inclusion, pick any $s \in \mathrm{pre}\,\mathcal{K}$ and choose $w \in \Sigma^{\omega}$ such that $sw \in \mathcal{K} \subseteq \mathcal{E} \subseteq \mathcal{L}$. In particular, there exists $r < w$ with $sr \in L$. With $sr < sw$, we obtain $sr \in (\mathrm{pre}\,\mathcal{K}) \cap L = K$, and, hence, $s \in \mathrm{pre}\,K$. This concludes the proof of $\mathrm{pre}\,K = \mathrm{pre}\,\mathcal{K}$, and immediately implies (L1). Likewise, (L2) and (L3) follow directly from controllability (i) and normality (ii), respectively. Regarding (L4), observe $K = (\mathrm{pre}\,\mathcal{K}) \cap L \subseteq (\mathrm{pre}\,\mathcal{E}) \cap L = E$. Thus, we have $K^{\uparrow} \supseteq (\mathrm{pre}\,\mathcal{K}^{\Uparrow}) \cap L$, and, by taking limits, obtain $\lim K^{\uparrow} \supseteq \lim((\mathrm{pre}\,\mathcal{K}^{\Uparrow}) \cap L) = \mathrm{clo}\,\mathcal{K}^{\Uparrow} \cap \mathcal{L} = \mathcal{K}^{\Uparrow}$. This concludes the proof of $\lim K^{\uparrow} = \mathcal{K}^{\Uparrow}$. □

Lemma 7 [see also (Baier and Moor, 2012), Proposition 9] If $\mathcal{L}$ is a non-anticipating IO-plant, then $\mathcal{L}_\Sigma$ is ω-controllable w.r.t. $(\Sigma_\mathrm{c} \dot\cup U_\mathrm{p} \dot\cup U_\mathrm{e},\ \mathrm{clo}\,\mathcal{L}_\Sigma)$.

Proof Note that

$$\mathcal{L}_\Sigma = \,(\mathrm{p}_\mathrm{pe}^{-\omega}(\mathcal{L} \cup \mathrm{pre}\,\mathcal{L})) \cap (\mathrm{clo}\,(\,(Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* \,(Y_\mathrm{e}U_\mathrm{e})^*\,)^\omega\,)\,,$$

implies $\mathrm{pre}\,\mathcal{L}_\Sigma \subseteq \mathrm{pre}(\mathrm{p}_\mathrm{pe}^{-\omega}(\mathcal{L} \cup \mathrm{pre}\,\mathcal{L})) = \mathrm{p}_\mathrm{pe}^{-1}\,\mathrm{pre}\,\mathcal{L}$. Pick an arbitrary string $s \in \mathrm{pre}\,\mathcal{L}_\Sigma$, let $r := \mathrm{p}_\mathrm{pe}s$, and observe that $r \in \mathrm{pre}\,\mathcal{L}$. Since $\mathcal{L}$ is non-anticipating, we can choose $\mathcal{W}_r \subseteq \mathcal{L}$, such that $r \in \mathrm{pre}\,\mathcal{W}_r$, and $\mathrm{pre}\,\mathcal{W}_r$ is controllable w.r.t. $(U_\mathrm{p} \dot\cup U_\mathrm{e},\ \mathrm{pre}\,\mathcal{L})$, and $\mathcal{W}_r$ is relatively closed w.r.t. $\mathrm{clo}\,\mathcal{L}$. Recall that relative closedness w.r.t. a closed language implies closedness. In particular, $\mathcal{W}_r$ is closed. To establish the non-anticipating property of $\mathcal{L}_\Sigma$, consider the candidate

$$\mathcal{V}_s := (\mathrm{p}_\mathrm{pe}^{-\omega}(\mathcal{W}_r \cup \mathrm{pre}\,\mathcal{W}_r)) \;\cap\; (\mathrm{clo}((Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* (Y_\mathrm{e}U_\mathrm{e})^*)^\omega).$$

Clearly, $\mathcal{V}_s \subseteq \mathcal{L}_\Sigma$ and $\mathrm{pre}\,\mathcal{V}_s \subseteq \mathrm{pre}\,\mathrm{p}_\mathrm{pe}^{-\omega}(\mathcal{W}_r \cup \mathrm{pre}\,\mathcal{W}_r) = \mathrm{p}_\mathrm{pe}^{-1}\,\mathrm{pre}\,\mathcal{W}_r$. Further, we have that $s \in \mathrm{pre}\,\mathcal{V}_s$, since $\mathrm{p}_\mathrm{pe}s = r \in \mathrm{pre}\,\mathcal{W}_r$ and $s \in \mathrm{pre}((Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* (Y_\mathrm{e}U_\mathrm{e})^*)^\omega$. To show controllability, pick an arbitrary string $\hat{s} \in \mathrm{pre}\,\mathcal{V}_s$ and $\sigma \in \Sigma_\mathrm{c} \dot\cup U_\mathrm{p} \dot\cup U_\mathrm{e}$ such that $\hat{s}\sigma \in \mathrm{pre}\,\mathcal{L}_\Sigma$. In particular, $\mathrm{p}_\mathrm{pe}\hat{s} \in \mathrm{p}_\mathrm{pe}\mathrm{p}_\mathrm{pe}^{-1}\,\mathrm{pre}\,\mathcal{W}_r = \mathrm{pre}\,\mathcal{W}_r$ and $\mathrm{p}_\mathrm{pe}(\hat{s}\sigma) \in \mathrm{p}_\mathrm{pe}\,\mathrm{pre}\,\mathcal{L}_\Sigma \subseteq \mathrm{pre}\,\mathcal{L}$. Controllability of $\mathrm{pre}\,\mathcal{W}_r$ w.r.t. $\mathrm{pre}\,\mathcal{L}$ implies that $\mathrm{p}_\mathrm{pe}(\hat{s}\sigma) \in \mathrm{pre}\,\mathcal{W}_r$. In addition, there exists $u \in \Sigma_\mathrm{pe}^\omega$, such that $\mathrm{p}_\mathrm{pe}(\hat{s}\sigma)u \in \mathcal{W}_r$. We choose $w \in \Sigma^\omega$ such that $\hat{s}\sigma\, w \in \mathrm{clo}((Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* (Y_\mathrm{e}U_\mathrm{e})^*)^\omega$ and $\mathrm{p}_\mathrm{pe}^\omega(\hat{s}\sigma\, w) = \mathrm{p}_\mathrm{pe}(\hat{s}\sigma)u$. Note that $\hat{s}\sigma\, w \in \mathcal{V}_s$ and, hence, $\hat{s}\sigma \in \mathrm{pre}\,\mathcal{V}_s$. Finally, we have to establish relative closedness of $\mathcal{V}_s$ w.r.t. $\mathrm{clo}\,\mathcal{L}_\Sigma$. Since $\mathrm{clo}\,\mathcal{L}_\Sigma$ is a closed superset of $\mathcal{V}_s$, relative closedness of $\mathcal{V}_s$ and closedness of $\mathcal{V}_s$ are equivalent. The latter is obtained by

$$\begin{aligned}\mathcal{V}_s &= (\mathrm{p}_\mathrm{pe}^{-\omega}(\mathcal{W}_r \cup \mathrm{pre}\,\mathcal{W}_r)) \;\cap\; (\mathrm{clo}((Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* (Y_\mathrm{e}U_\mathrm{e})^*)^\omega)\\ &= (\mathrm{p}_\mathrm{pe}^{-\omega}\,\mathrm{clo}\,\mathcal{W}_r \cup \mathrm{p}_\mathrm{pe}^{-\omega}\,\mathrm{pre}\,\mathcal{W}_r) \cap (\mathrm{clo}((Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* (Y_\mathrm{e}U_\mathrm{e})^*)^\omega)\\ &= (\mathrm{clo}\,\mathrm{p}_\mathrm{pe}^{-\omega}\mathcal{W}_r) \;\cap\; (\mathrm{clo}((Y_\mathrm{p}(Y_\mathrm{c}U_\mathrm{c})^* U_\mathrm{p})^* (Y_\mathrm{e}U_\mathrm{e})^*)^\omega).\end{aligned}$$

The intersection of two topologically closed languages is topologically closed. □

Lemma 8 [see also (Baier and Moor, 2012), Lemma 13] For a non-anticipating IO-plant $\mathcal{L}$, the full behaviour can be represented as a union $\mathcal{L}_\Sigma = \cup_{a \in A}\mathcal{L}_a$, where for all $a \in A$

(i) $\mathcal{L}_a$ has locally free inputs U_c, $U_\mathrm{p} \dot\cup Y_\mathrm{c}$, and U_e.
(ii) $\mathcal{L}_a$ is closed.

Proof Technically, P3 together with (Baier and Moor, 2012), Proposition 9, implies that $\mathcal{L}_\Sigma$ itself is the supremal ω-controllable sublanguage of $\mathcal{L}_\Sigma$. Thus, by (Moor et al, 2011), Proposition 12, $\mathcal{L}_\Sigma$ can be represented as a union $\mathcal{L}_\Sigma = \cup_{a \in A}\mathcal{L}_a$, where, for each, $a \in A$, $\mathrm{pre}\,\mathcal{L}_a$ is controllable w.r.t. $(\Sigma_\mathrm{c} \dot\cup U_\mathrm{p} \dot\cup U_\mathrm{e},\ \mathrm{pre}\,\mathcal{L}_\Sigma)$ and $\mathcal{L}_a$ is closed. To establish (i), we pick $s \in \Sigma^*$, μ, $\mu' \in U_\mathrm{c}$, with $s\mu \in \mathrm{pre}\,\mathcal{L}_a$. The locally free input of $\mathrm{pre}\,\mathcal{L}_\Sigma$ implies $s\mu' \in \mathrm{pre}\,\mathcal{L}_\Sigma$, and controllability of $\mathrm{pre}\,\mathcal{L}_a$ w.r.t. $\mathrm{pre}\,\mathcal{L}_\Sigma$ implies, that $s\mu' \in \mathrm{pre}\,\mathcal{L}_a$. Locally free inputs $U_\mathrm{p} \dot\cup Y_\mathrm{c}$, and U_e are verified likewise. □