A process algebra for Interworkings

S. Mauw and M. A. Reniers
Faculty of Mathematics and Computing Science,
Eindhoven University of Technology,
P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands
CWI,
P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands
Email: sjouke@win.tue.nl, M.A.Reniers@tue.nl

Abstract

The Interworking language (IW) is a graphical formalism for displaying the communication behaviour of system components. In this chapter, we develop a formal semantics for the Interworking language. This semantics must support the analysis of (collections of) Interworking diagrams and allow to express the relation between diagrams. We will explain how techniques from process algebra can be successfully applied to this problem. Thereto, we introduce process operators for expressing the relationship between Interworking diagrams. We define a number of process algebras with increasing complexity. For each of these we prove completeness with respect to an operational semantics.
Keywords: process algebra, Interworkings, semantics, composition operators.
Note: To appear as a chapter in *Handbook of Process Algebra*, editors A. Ponse and S. Smolka, Elsevier Science Publishers B.V.

1 Introduction

1.1 History and motivation

The Interworking language (IW) is a graphical formalism for displaying the communication behaviour of system components. It was developed in order to support the informal diagrams used at Philips Kommunikations Industrie (Nürnberg) which were used for requirements specification and design. Before discussing the rationale behind the IW language, we first show a simple Interworking diagram[1] in Figure 1. The name of the Interworking is displayed in the upper left corner

[1]The Interworking diagrams in this chapter are drawn with the *MSC Macro package* which can be obtained at http://www.win.tue.nl/~sjouke/mscpackage.html.

of the diagram. This Interworking describes the interaction behaviour of three entities, which are called *s*, *medium* and *r*. Each entity is represented by a vertical line, which, when read from top to bottom, describes the successive interactions in which this entity takes part. A message exchange is represented by an arrow. The diagram shows that the three entities exchange four messages. First, *s* sends a *req* message to *medium*. Next, the same message is being sent from *medium* to *r*. Then, *r* sends a message *reply* back to *medium*, which sends the same message to *s*.

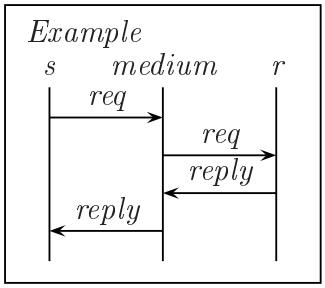

Figure 1: An example Interworking diagram

This example shows the basic use of Interworkings. It describes one scenario of interaction between communicating entities. In general, when using IW for requirements specification, a collection of Interworkings is needed, containing a description of the most interesting scenarios. Often there is one main scenario, complemented with a number of scenarios describing exceptional behaviour. Using Interworkings in this way, the scenarios express alternative behaviours.

There are, however, more reasons for having to deal with large collections of Interworkings for the description of a distributed system. First, the specified scenario can be too long to physically or logically fit in one diagram. Such a large scenario is then decomposed into a number of sub scenarios which are "sequentially" linked to each other.

A second reason is that the horizontal size of the system, or more precisely the number of distinct entities, may be too large to fit in a single diagram. This gives rise to a collection of sub scenarios which denote the behaviour of different parts of the system. Each part then describes the behaviour of just a number of (logically related) entities. Of course, there must be a means to express that entities from distinct parts exchange messages with each other. The scenarios of these parts are linked to each other in a parallel way.

Due to the above mentioned reasons, in practice a system description using Interworkings often consists of a large collection of diagrams. Practical experience showed that it was very hard to maintain such large collections by hand. First of all, manually drawing and updating diagrams is an expensive activity. Secondly, the relation between the diagrams in a collection is only implicit. Some diagrams describe alternatives, some describe successive behaviour, and some describe parallel behaviour. The third problem, assuming the relation between the diagrams to be known, is that if one diagram changes also several related diagrams must be updated. A consistent update of a large collection of Interworkings could not be achieved manually. A final problem was that there existed different interpretations of the meaning of even simple Interworkings.

These observations lead to the conclusion that when using Interworkings in the traditional and informal way it was not possible to take full advantage of the language. Therefore, the Interworking language needs a complete and explicit definition.

Not only the development of an explicit language is motivated in this way, but also the need for a formal semantics of Interworkings. This semantics must support the analysis of (collections of) Interworking diagrams and allow to express the relation between diagrams. Moreover, since tool support is needed, the semantics must allow for easy derivation of (prototype) tools.

In this chapter, we will explain how techniques from process algebra can be successfully applied to this problem. Thereto, we introduce process operators for expressing the relationship between Interworking diagrams. As explained above, there are three possible relations between Interworkings: alternative composition, sequential composition, and parallel composition. The most interesting is the *interworking sequencing* operator for composing Interworkings sequentially. Later in this chapter it will be explained why the standard process algebra operator for sequencing is not appropriate for Interworkings. The operator for parallel composition of Interworkings, is derived from the standard interleaving operator with synchronisation. For the alternative composition operator there are different choices. For a discussion on this choice we refer to Section 2.3.

1.2 Interworkings and similar languages

The Interworking language is not a unique and isolated language. It is very natural and intuitive to express the behaviour of a distributed system in such a graphical way. In fact, informal IW-like drawings are encountered very often in system design.

Therefore, the Interworking language is a member of a large class of similar graphical notations, most of which are only informally defined, such as Signal Sequence Charts, Use Cases, Information Flow Diagrams, Message Flow and Arrow Diagrams. In object oriented design, a similar notation, called Sequence Diagrams, is used. They play an important role in the description of Use Cases in UML [RJB99]. Interworkings are also related to Message Sequence Charts (MSC), see [IT93], which are standardised by the International Telecommunication Union (ITU). The main difference is that Interworkings describe synchronous communication, whereas Message Sequence Charts describe asynchronous communication. The semantics of MSC as described in [MR99, Ren99] is also very similar to the semantics of IW.

Traditionally, the main application area for IW and similar languages is the field of telecommunication systems. This is mainly due to the distributed nature of these systems. However, more and more applications outside the telecommunication world can be found, e.g. the description of work flows in business organisations [Aal99].

The main reason why IW-like diagrams are so popular is the fact that they can be understood easily. This is due to their intuitive and graphical appearance. The diagrams can be used in different stages of the design of a software system. The main application is during requirements engineering, where they are used to capture initial requirements about the interactions in a system. Furthermore, they play a role in documentation, simulation and testing.

The results of this chapter cannot completely be transferred to similar languages. This is mainly because IW describes synchronous communication, whereas most similar languages consider asynchronous communication. Nevertheless, the approach taken in this chapter is generic. It is at the basis of the semantics definition of Message Sequence Charts, as standardised by the ITU in Annex B to Recommendation Z.120 [IT95].

1.3 Purpose and structure of this chapter

This chapter serves several purposes. First, it shows the process algebraic approach in defining the semantics of a scenario language. This typically entails the use of a number of operators which describe the ways in which scenarios or fragments of scenarios are combined. The meaning of such a diagram is then described by a process algebraic expression, which can be analysed using standard techniques.

Secondly, this chapter shows the development of non-standard operators in process algebra, needed for some domain specific application. These newly introduced operators will probably have little application outside the realm of scenarios. On the other hand, the interworking sequencing operator already received attention in a more general context, and was named *weak sequential composition* (see [RW94]).

Thirdly, we show in detail which (proof) obligations occur when introducing new operators. We both give an operational and an algebraic definition, and prove their correspondence.

This chapter is subdivided as follows.

First, we will introduce the Interworking language and the operators for combining Interworkings (Section 2). Next, we formally define the operators involved. We will not simply give one process algebra containing all operators, but we will formalise the operators in a modular way. This yields a collection of process algebras, for which we obtain some additional proof obligations, such as conservativity. The first process algebra (defined in Section 3) only contains the operator for sequential composition. This operator suffices to give a formal semantics of Interworking diagrams. In Section 4 we define the theory of the basic process algebra operators ($+$ and $\cdot$) which we enrich with partial deadlocks. Next, in Section 5, these process algebras are combined. The following two sections deal with the introduction of the interworking merge operator. In Section 6 we first define a parameterised version of this operator, the E-interworking merge. The general interworking merge operator is defined in Section 7, which yields the final process algebra for Interworkings.

Every operator is both defined algebraically and by means of an operational semantics. The relation between these descriptions is given in several soundness and completeness theorems.

The treatment of Interworkings in the current chapter is mainly on a theoretic level. We will not introduce graphical and linear syntax of the language, and we will not present a mapping from Interworking diagrams to process algebra expressions (for a thorough treatment see [MvWW92]). Our main goal is to define the theory needed to formally understand Interworkings. Neither will we explain methodological aspects of the use of Interworkings or supporting tools. For a

description of a prototype tool set based on these semantical definitions, we refer to [MW93].

2 Interworkings

An Interworking specification consists of a collection of Interworking diagrams. The relation between these diagrams is defined by means of operators. An Interworking diagram specifies (part of) a single scenario and the operators can be used to compose simple scenarios into more complex scenarios. We consider operators for sequential composition, alternative composition and parallel composition of Interworkings.

In this section we will only give an informal explanation of syntax and semantics of Interworkings. Simple examples show the relevant properties, which are formalised in the sections to come.

We will not give a formal definition of the graphical syntax of Interworkings, since for our purposes an informal and intuitive mapping from Interworkings to the semantical domain suffices. There exists a textual representation of Interworkings too, but we will not discuss this. Consult [MvWW92] for more information on this topic.

2.1 Interworking diagrams

An example of an Interworking diagram is shown in Figure 2. Such a diagram consists of a number of vertical lines and horizontal arrows, surrounded by a frame. The name of the Interworking diagram (*Co-operation*) is in the upper left corner of the frame. The vertical lines denote the entities of which (part of) the behaviour is being described. Above the lines are the names of these entities. Here we have four entities, called a, b, c, and d.

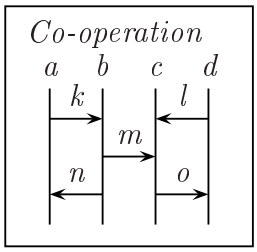

Figure 2: An example Interworking diagram

The arrows denote the exchange of messages between the entities. Interworkings describe synchronous communication, which means that an arrow represents one single event. The order in which the communications take place is also expressed in the diagram. On every entity axis, time runs from top to bottom and the events connected to an entity axis are causally ordered in this way. However, there is no global time axis and the only way to synchronise the behaviour of the entities is by means of a message exchange. So, message k causally precedes message m. And

because m precedes o, we have that k also precedes o. Messages k and l are not causally related; they may occur in any order. In our semantical treatment we assume an interleaved model of operation, which means that k and l cannot occur simultaneously.

The fact that the time lines of all entities are independent, implies that the vertical placement of two messages which are not causally related has no semantical meaning. Therefore, the Interworkings from Figure 3 have identical semantics.

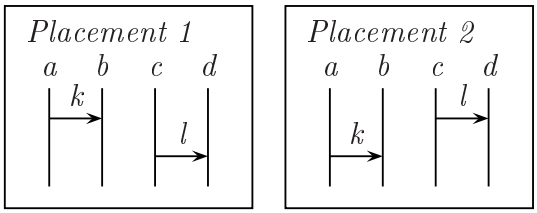

Figure 3: Two semantically equivalent Interworkings

A special case in our semantics is the *empty Interworking*. This is an Interworking which describes no behaviour at all and contains no entities. In the next sections the empty Interworking is denoted by ε.

2.2 Sequencing

Sequential composition is the easiest way to compose two Interworkings. Intuitively, sequential composition can be considered as the concatenation of two Interworkings, thereby connecting the corresponding entity axes. Figure 4 shows the sequential composition of two Interworkings. The circle denotes the sequencing operator.

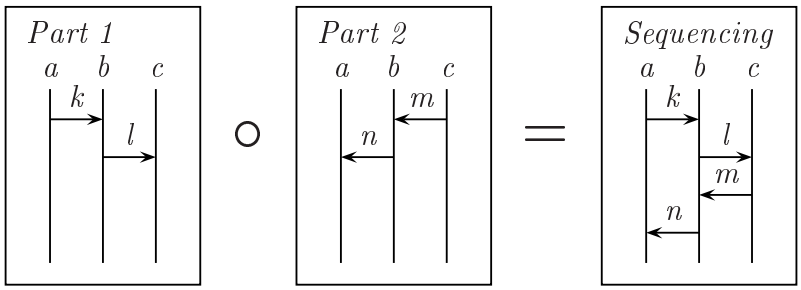

Figure 4: Sequential composition of two Interworkings

One must take into account that there is no (implicit) synchronisation between the entities at the point where the two Interworkings are concatenated. For this reason, the operator for sequential composition of Interworkings is called the *weak* sequential composition operator (or *interworking sequencing*). Although we will also introduce an operator for strong sequential composition

of Interworkings in our semantical treatment, this operator is not part of the Interworkings language. Figure 5 shows that the weak sequential composition of two unrelated messages gives an Interworking where these two messages still are unordered.

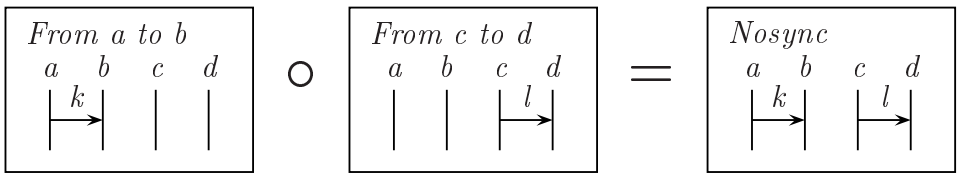

Figure 5: No synchronisation through sequential composition

In the previous examples, the two composed Interworkings contained the same set of entities. This is not a requirement for sequential composition. The Interworking resulting from a sequential composition simply contains all entities from its constituents, as shown in Figure 6.

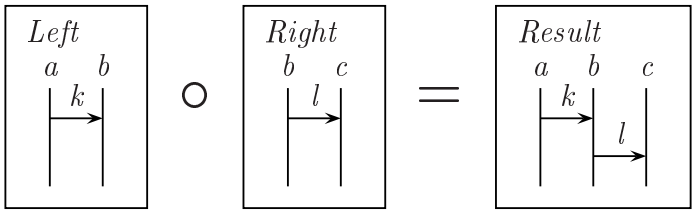

Figure 6: Sequential composition with different entity sets

Given the above interpretation of Interworking diagrams and sequential composition, the following observation is apparent. Every Interworking diagram is equivalent to the sequential composition of all its events. Look e.g. at Interworking *Co-operation* (Figure 2) which is the sequential composition of five simple Interworking diagrams, each containing one arrow. The order in which these Interworkings are composed should of course correspond to the causal ordering of the original Interworking. So, if K, L, M, N, and O are Interworking diagrams containing the messages k, l, m, n, and o, respectively, then $L \circ K \circ M \circ N \circ O$ would be an example of such an expression. An alternative for this expression is $K \circ L \circ M \circ O \circ N$.

2.3 Alternatives

In theoretical approaches to MSC-related languages different operators for alternative composition are used. These are the delayed choice operator ($\mp$, see [BM95]) and the non-deterministic choice operator ($+$, see [BW90]). In the standardised semantics of MSC [Ren99] the delayed choice operator is used. The essential difference between these two operators is that non-deterministic choice determines the moment of choice between the alternatives at the place where it occurs, whereas the delayed choice postpones the moment of choice to the place where the alternatives start to differ. The latter leads to a trace semantics (if non-deterministic choice is not present as

well). As a consequence also all other operators in which a choice is manifest (such as parallel composition) must be changed to adopt the delayed interpretation of choices [Ren99]. In our opinion the use of the delayed choice is only interesting if non-deterministic choice is present too. If the delayed choice is the only alternative composition operator of interest, then a better solution is to adopt a trace theoretical approach towards the semantics. In the process algebra approach of this handbook it seems more appropriate to study the non-deterministic choice operator.

Hence, the operator which expresses the fact that two Interworkings describe alternative scenarios is denoted by +. Figure 7 contains an example of the choice between two alternative Interworking diagrams. This expression describes the non-deterministic choice between the two given scenarios. Both scenarios start with message k, but the first continues with message l and the second with messages m and n.

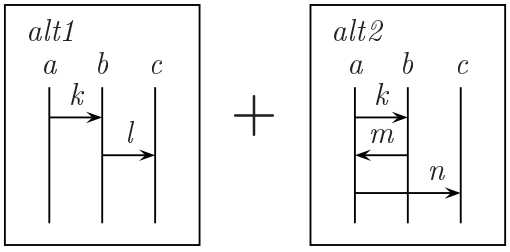

Figure 7: Alternative composition of two Interworking diagrams

Notice that the class of Interworking diagrams is not closed under application of the +-operator. The behaviour defined in Figure 7 cannot be expressed without application of the +.

2.4 Merge

Whereas the sequencing operator is used for vertical composition of Interworkings, the merge operator is used for horizontal composition.

In the case that the two operands have no entities in common, the merge of two Interworking diagrams is simply their juxtaposition, as illustrated in Figure 8.

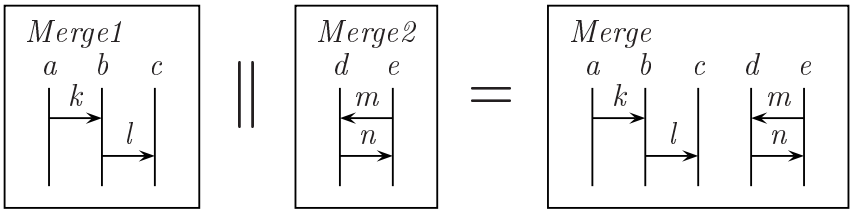

Figure 8: Merge of Interworking diagrams without shared entities

In the case that the two operands do share some entities, the situation is a bit more complicated.

Suppose, for example, that the two Interworking diagrams have two entities in common, as in Figure 9. Then the messages exchanged between the shared entities must be identical for both operands. The resulting Interworking contains only one occurrence of every shared entity. Also the messages exchanged between the shared entities, which must occur in the same order in both operands, appear only once in the resulting Interworking. In Figure 9 the two operands share the entities c and d with shared messages m and n.

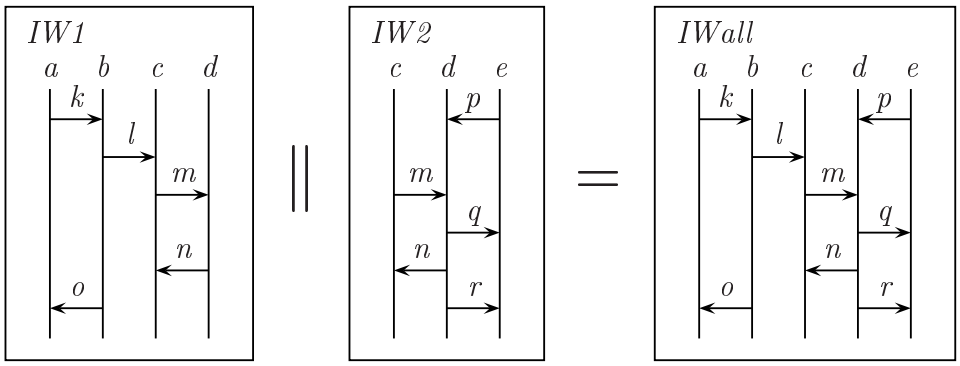

Figure 9: Merge of Interworking diagrams with two shared entities

In the case that the two operands do not describe identical behaviour with respect to the shared entities, as in Figure 10, a *deadlock* occurs. The resulting Interworking contains the parallel behaviour of the operands, up to the point where the behaviours on the shared entities start to diverge. At this point the deadlock occurs, denoted by two horizontal bars. Such a deadlock only covers entities which are blocked. This means that we do not have the *global deadlock* as used elsewhere, but a partial deadlock. We refer to this partial deadlock as deadlock. An entity shows no behaviour after it has entered a deadlock situation. All behaviour which is causally dependent on a communication which causes the deadlock, is also blocked. In Figure 10 this means that, since messages x and n do not match, a deadlock occurs on entities c and d. Moreover, since message r is causally dependent upon message n, the deadlock extends to entity e. In the following sections, such a deadlock will be denoted by δ_E, where E is the set of deadlocked entities. If a deadlock occurs as a consequence of merging two Interworkings, we say that the two operands are *merge-inconsistent*.

This explanation of the merge operator generalises easily to the case where the operands have more than two entities in common. However, the case where they share only one entity yields a different situation. It is clear that this shared entity should occur only once in the resulting Interworking, but what happens with the events that this entity takes part in? This situation occurs in Figure 11. There is no reason to introduce a causal ordering between the messages l and m, and therefore the result cannot be a single Interworking diagram. The result of the merge in Figure 11 contains two alternative Interworking diagrams, which together describe all possible orderings of l and m.

Care has to be taken to correctly handle entities which are included in an Interworking diagram but which do not take part in any communication, so-called *empty entities*. In the case that such an entity occurs in the set of shared entities, it cannot be discarded. Figure 12 shows an example.

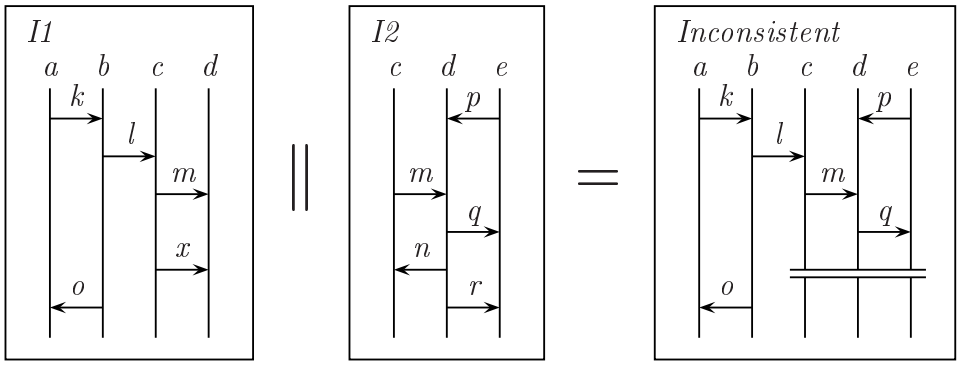

Figure 10: Merge of two inconsistent Interworking diagrams

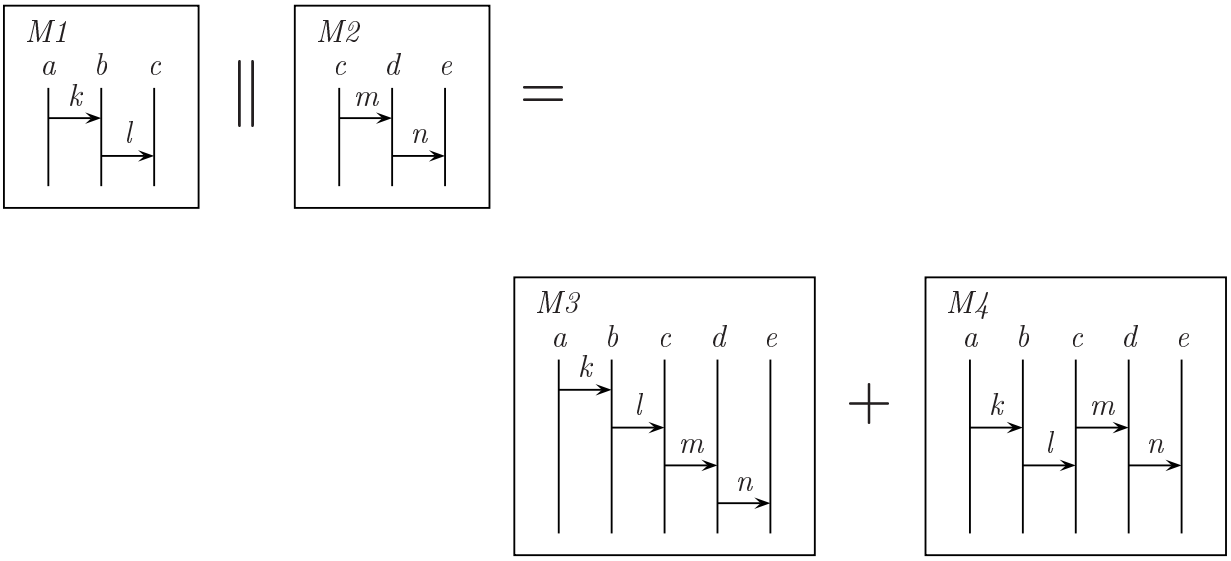

Figure 11: Consistent merge

Entity b occurs in both operands, but in the second operand there is no behaviour associated to b. Because in the first operand a message l is sent to b, a deadlock occurs.

The situation would be quite different if we would omit entity b from the second operand. Then the two operands would be merge-consistent. This is shown in Figure 13.

3 Semantics of interworkings

In this section we will present a simple process algebra that can be used for reasoning about the equality of Interworking diagrams. Based on the textual syntax of Interworking diagrams a process term is generated as follows. With every message in the Interworking diagram an atomic

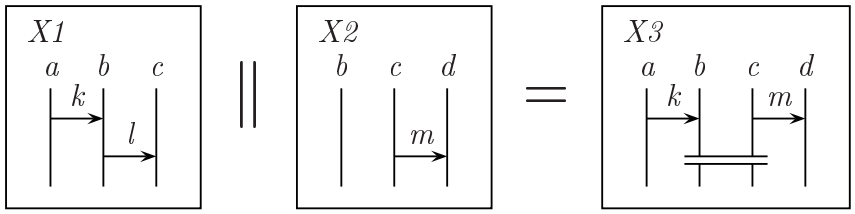

Figure 12: Inconsistent Interworking diagrams with empty entity

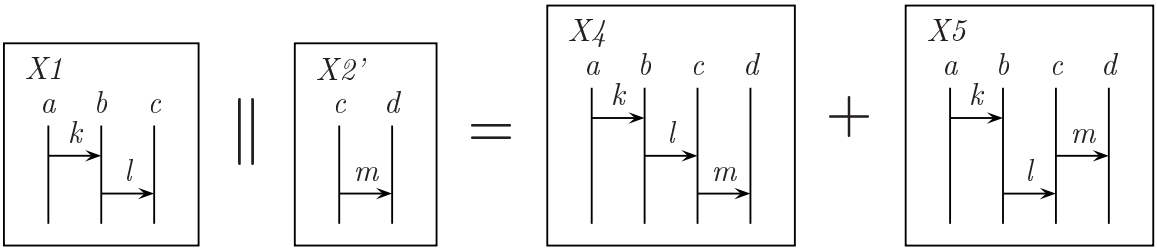

Figure 13: Empty entity removed

action is associated. A deadlock that covers the entities from a set E is denoted by δ_E. The atomic actions are combined by means of interworking sequencing. The process algebra is called $IWD(\circ)$.

We assume the existence of sets *EID* and *MID* of names of entities and messages, respectively. Actually, these can be considered as parameters of the process algebra. A message is characterised by the name of the sender, the name of the receiver, and the message name. These messages form the set of atomic actions.

Definition 1 (Atomic actions) The set A of *atomic actions* is given by

$$A = \{c(i, j, m) \mid i, j \in EID, m \in MID\}.$$

Definition 2 (Signature of *IWD*$(\circ)$) The *signature* Σ_{IWD} of the process algebra $IWD(\circ)$ consists of the atomic actions $a \in A$, the deadlock constants δ_E ($E \subseteq EID$), the empty process ε, and the binary operation interworking sequencing $\circ_{\mathrm{iw}}$.

The set of all (open) terms over the signature Σ_{IWD} is denoted as $\mathcal{O}(\Sigma_{IWD})$. The set of all closed terms over the signature Σ_{IWD} is denoted as $\mathcal{C}(\Sigma_{IWD})$. We will use similar notations for other signatures.

We provide the process algebra with an operational semantics by associating a term deduction system to it. We will first summarise the terminology related to term deduction systems. For a formal definition of term deduction systems and related notions we refer to [BV95]. A term deduction system is a structure (Σ, D) where Σ is a signature and D a set of deduction rules. The set of deduction rules is parameterised by a set of relation symbols and a set of predicate symbols. If P is such a predicate symbol, R such a relation symbol, and $s, t \in \mathcal{O}(\Sigma)$, then the

expressions Ps and sRt are called formulas. A deduction rule is of the form $\frac{H}{C}$ where H is a set of formulas, called *hypotheses*, and C is a formula, called the *conclusion*.

In the term deduction systems used in this chapter we use relations $\_ \stackrel{\_}{\rightarrow} \_ \subseteq \mathcal{O}(\Sigma) \times A \times \mathcal{O}(\Sigma)$ and the predicate $\_\downarrow \subseteq \mathcal{O}(\Sigma)$. The formula $x \stackrel{a}{\rightarrow} x'$ expresses that the process x can perform an action a and thereby evolves into the process x'. The formula $x\downarrow$ expresses that process x has an option to terminate immediately and successfully.

In the remainder of this chapter we use the following shorthands: $x \stackrel{a}{\rightarrow}$ represents the predicate that $x \stackrel{a}{\rightarrow} x'$ for some x', $x \stackrel{a}{\not\rightarrow} x'$ represents the proposition that $x \stackrel{a}{\rightarrow} x'$ is not derivable from the deduction system, $x \stackrel{a}{\not\rightarrow}$ represents $\neg(x \stackrel{a}{\rightarrow})$, and $x \not\rightarrow$ represents $x \stackrel{a}{\not\rightarrow}$ for all $a \in A$. Similarly we use $x \not\downarrow$ to represent $\neg(x\downarrow)$.

A proof of a formula ϕ is a well-founded upwardly branching tree of which the nodes are labeled by formulas such that the root is labeled by the formula ϕ and if χ is the label of a node and $\{\chi_i \mid i \in I\}$ is the set of labels belonging to the nodes directly above it, then

$$\frac{\{\chi_i \mid i \in I\}}{\chi}$$

is an instantiation of a deduction rule.

The term deduction system for the process algebra $IWD(\circ)$ consists of the signature Σ_{IWD} and the deduction rules given in Table 1.

Before we can give the operational description of the interworking sequencing operator we first define the *active entities* associated with a process term representing an Interworking diagram. The active entities of an Interworking diagram are those entities which are involved in a communication or in a deadlock.

Definition 3 (Active entities) For $i, j \in EID$, $m \in MID$, $E \subseteq EID$, and $x, y \in \mathcal{C}(\Sigma_{IWD})$ we define the mapping $AE : \mathcal{C}(\Sigma_{IWD}) \to I\!P(EID)$ inductively as follows:

$$\begin{array}{lcl} AE(c(i,j,m)) & = & \{i,j\}, \\ AE(\varepsilon) & = & \emptyset, \\ AE(\delta_E) & = & E, \\ AE(x \circ_{\mathrm{iw}} y) & = & AE(x) \cup AE(y). \end{array}$$

The operational semantics of the process algebra $IWD(\circ)$ is given by the deduction rules in Table 1 and the equations defining the active entities in Definition 3. These equations can easily be written as deduction rules. The empty process does not execute any actions, but it terminates successfully and immediately. The fact that it does not execute any action is visible by the impossibility of deriving that it can execute an action. The process a can execute the action a and in doing so evolves into the empty process ε. The process δ_E cannot execute any actions nor can it terminate successfully. The interworking sequencing of two processes terminates if and only if both processes can terminate. The process $x \circ_{\mathrm{iw}} y$ executes an action a if x can execute action a or if y can execute a and this action is not related to an active entity of x

$(AE(a) \cap AE(x) = \emptyset)$. This expresses the intuition that the first operand may always perform its actions, while the second operand may only perform actions which are not blocked because they are causally dependent on actions from the first operand.

$$\frac{}{\varepsilon \downarrow} \qquad \frac{}{a \xrightarrow{a} \varepsilon} \qquad \frac{x \downarrow \quad y \downarrow}{x \circ_{\mathrm{iw}} y \downarrow}$$

$$\frac{x \xrightarrow{a} x'}{x \circ_{\mathrm{iw}} y \xrightarrow{a} x' \circ_{\mathrm{iw}} y} \qquad \frac{AE(a) \cap AE(x) = \emptyset \quad y \xrightarrow{a} y'}{x \circ_{\mathrm{iw}} y \xrightarrow{a} x \circ_{\mathrm{iw}} y'}$$

Table 1: Deduction rules for interworking sequencing ($a \in A$)

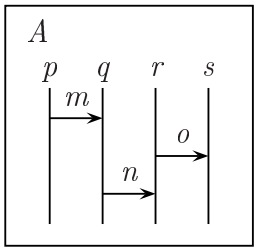

Figure 14: Example of an Interworking diagram

The Interworking from Figure 14 can semantically be represented by the process term

$$c(p,q,m) \circ_{\mathrm{iw}} (c(r,s,o) \circ_{\mathrm{iw}} c(q,r,n)).$$

Then the following is a derivation of the fact that first the communication of message o can take place:

$$\frac{AE(c(r,s,o)) \cap AE(c(p,q,m)) = \emptyset \qquad \dfrac{c(r,s,o) \xrightarrow{c(r,s,o)} \varepsilon}{c(r,s,o) \circ_{\mathrm{iw}} c(q,r,n) \xrightarrow{c(r,s,o)} \varepsilon \circ_{\mathrm{iw}} c(q,r,n)}}{c(p,q,m) \circ_{\mathrm{iw}} (c(r,s,o) \circ_{\mathrm{iw}} c(q,r,n)) \xrightarrow{c(r,s,o)} c(p,q,m) \circ_{\mathrm{iw}} (\varepsilon \circ_{\mathrm{iw}} c(q,r,n))}$$

Two processes x and y are considered equivalent if they can mimic each others behaviour in terms of the predicates and relations that are used in the term deduction system. In this case these are the execution of actions, the termination of a process, and the active entities of a process. As a consequence of introducing partial deadlock constants, we must be able to distinguish deadlocks over different sets of entities. This is the reason that we require that two processes are equivalent only if they have the same active entities. This type of equivalence is usually called *strong bisimilarity*, but we call it IWD-bisimilarity.

Definition 4 (IWD-bisimilarity) Let Σ be a signature. A symmetric relation $R \subseteq \mathcal{C}(\Sigma) \times \mathcal{C}(\Sigma)$ is called an *IWD-bisimulation* iff for all p, q such that pRq we have

1. $AE(p) = AE(q)$;
2. if $p\downarrow$, then $q\downarrow$;
3. if $p \xrightarrow{a} p'$ for some $a \in A$ and p', then there exists a q' such that $q \xrightarrow{a} q'$ and $p'Rq'$.

Two processes x and y are called *IWD-bisimilar*, notation $x \underline{\leftrightarrow}_{\text{iwd}} y$, iff there exists an IWD-bisimulation R such that xRy. The notation $R : x \underline{\leftrightarrow}_{\text{iwd}} y$ expresses that R is an IWD-bisimulation that relates x and y.

Theorem 1 (Equivalence) IWD-bisimilarity is an equivalence relation.

Proof. We must prove that IWD-bisimilarity is reflexive, symmetric, and transitive.

1. $\underline{\leftrightarrow}_{\text{iwd}}$ is reflexive. Let $R = \{(p, p) \mid p \in \mathcal{C}(\Sigma_{IWD})\}$. Clearly, R is an IWD-bisimulation.
2. $\underline{\leftrightarrow}_{\text{iwd}}$ is symmetric. Suppose that $p \underline{\leftrightarrow}_{\text{iwd}} q$. This means that there exists an IWD-bisimulation R such that pRq. Since any IWD-bisimulation is symmetrical we also have qRp. Hence $q \underline{\leftrightarrow}_{\text{iwd}} p$.
3. $\underline{\leftrightarrow}_{\text{iwd}}$ is transitive. Suppose $p \underline{\leftrightarrow}_{\text{iwd}} q$ and $q \underline{\leftrightarrow}_{\text{iwd}} r$. Thus there exist IWD-bisimulations R_1 and R_2 such that pR_1q and qR_2r. Let $R = (R_1 \circ R_2)^S$. For a relation ρ on X, the notation ρ^S denotes the symmetric closure of ρ. It is not hard to show that R is an IWD-bisimulation and pRr. Hence $p \underline{\leftrightarrow}_{\text{iwd}} r$.

⊠

Theorem 2 (Congruence) IWD-bisimilarity is a congruence for interworking sequencing.

Proof. The term deduction system for $IWD(\circ)$ is in path format. From [BV93], we then have that IWD-bisimilarity is a congruence for interworking sequencing. The path format is a syntactical restriction on the form of the deduction rules and can be easily checked. ⊠

In Table 2 we present the axioms of the process algebra $IWD(\circ)$.

Idem. $\circ_{\mathrm{iw}}$	$\varepsilon \circ_{\mathrm{iw}} x = x$	
Comm. $\circ_{\mathrm{iw}}$	$x \circ_{\mathrm{iw}} y = y \circ_{\mathrm{iw}} x$	if $AE(x) \cap AE(y) = \emptyset$
Ass. $\circ_{\mathrm{iw}}$	$(x \circ_{\mathrm{iw}} y) \circ_{\mathrm{iw}} z = x \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z)$	
$\circ_{\mathrm{iw}}1$	$\delta_E \circ_{\mathrm{iw}} a = \delta_{E \cup AE(a)}$	if $AE(a) \cap E \neq \emptyset$
$\circ_{\mathrm{iw}}3$	$\delta_E \circ_{\mathrm{iw}} \delta_F = \delta_{E \cup F}$	

Table 2: Axioms of $IWD(\circ)$ ($a \in A$, $E, F \in EID$)

The first three axioms express straightforward properties. The axioms $\circ_{\mathrm{iw}}1$ and $\circ_{\mathrm{iw}}3$ describe the propagation of partial deadlocks through the Interworking diagram. The first of these is illustrated in Figure 15 for $E = \{p, q\}$ and $a = c(q, r, m)$.

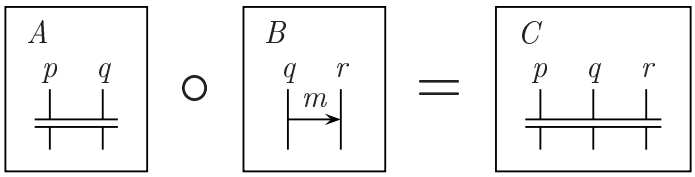

Figure 15: Propagation of partial deadlocks

For deriving equalities between process terms we can use all instantiations of the axioms and the usual laws of equational logic. These are reflexivity, symmetry, transitivity, and Leibniz's rule.

As a simple example, we present the derivation that the empty process is a right unit for interworking sequencing. The fact that it is a left unit is put forward as an axiom.

Lemma 1 (Properties) For $x \in \mathcal{O}(\Sigma_{IWD})$ we have $x \circ_{\mathrm{iw}} \varepsilon = x$.

Proof. As $AE(\varepsilon) = \emptyset$, we have $AE(x) \cap AE(\varepsilon) = \emptyset$. Then, using the axioms Comm. $\circ_{\mathrm{iw}}$ and Idem. $\circ_{\mathrm{iw}}$, we have $x \circ_{\mathrm{iw}} \varepsilon = \varepsilon \circ_{\mathrm{iw}} x = x$. ⊠

Thus far we have presented an operational semantics and a process algebra on the signature Σ_{IWD}. Ideally, there is a strong connection between these. In this case we will first show that every pair of derivably equal closed $IWD(\circ)$-terms is IWD-bisimilar. This relation between an equational theory and its model is usually referred to as *soundness* of the equational theory with respect to the operational semantics. It can also be stated from the point of view of the operational semantics: the set of closed $IWD(\circ)$-terms modulo IWD-bisimilarity is a *model* of the equational theory. Later we will also present a relation in the other direction: every pair of IWD-bisimilar closed $IWD(\circ)$-terms is also derivably equal. This result is referred to as *completeness*.

Theorem 3 (Soundness) *IWD*($\circ$) is a sound axiomatisation of IWD-bisimilarity on closed *IWD*($\circ$)-terms.

Proof. Due to the congruence of IWD-bisimilarity with respect to all operators from the signature of *IWD*($\circ$), it suffices to prove soundness of all closed instantiations of the axioms in isolation. We give an IWD-bisimulation for each of the axioms. These are the following

- for axiom Idem. $\circ_{\mathrm{iw}}$: $R = \{(\varepsilon \circ_{\mathrm{iw}} p, p) \mid p \in \mathcal{C}(\Sigma_{IWD})\}^S$;
- for axiom Comm. $\circ_{\mathrm{iw}}$: $R = \{(p \circ_{\mathrm{iw}} q, q \circ_{\mathrm{iw}} p) \mid p, q \in \mathcal{C}(\Sigma_{IWD}), AE(p) \cap AE(q) = \emptyset\}^S$;
- for axiom Ass. $\circ_{\mathrm{iw}}$: $R = \{(p \circ_{\mathrm{iw}} (q \circ_{\mathrm{iw}} r), (p \circ_{\mathrm{iw}} q) \circ_{\mathrm{iw}} r) \mid p, q, r \in \mathcal{C}(\Sigma_{IWD})\}^S$;
- for axiom $\circ_{\mathrm{iw}}1$: $R = \{(\delta_E \circ_{\mathrm{iw}} a, \delta_{E \cup AE(a)}) \mid AE(a) \cap E \neq \emptyset\}^S$;
- For axiom $\circ_{\mathrm{iw}}3$: $R = \{(\delta_E \circ_{\mathrm{iw}} \delta_F, \delta_{E \cup F}) \mid E, F \subseteq EID\}^S$.

⊠

The proof of completeness consists of a number of steps. First we define the notion of basic term and prove that every closed term is derivably equal to some basic term. The introduction of basic terms makes it easier to perform inductive reasoning on the structure of a closed term.

Definition 5 (Basic terms) The set of *basic terms* is the smallest set such that

1. ε is a basic term;
2. for $E \subseteq EID$, δ_E is a basic term;
3. for $a \in A$ and x a basic term, $a \circ_{\mathrm{iw}} x$ is a basic term.

The set of all basic terms over the signature Σ_{IWD} is denoted $\mathcal{B}(\Sigma_{IWD})$.

Theorem 4 (Elimination) For every closed term there is a basic term which is derivably equal.

Proof. By induction on the structure of closed term x.

1. $x \equiv \varepsilon$. This is a basic term.
2. $x \equiv \delta_E$ for some $E \subseteq EID$. This is a basic term.
3. $x \equiv a$ for some $a \in A$. Then, using Lemma 1, $a = a \circ_{\mathrm{iw}} \varepsilon$ which is a basic term.
4. $x \equiv x_1 \circ_{\mathrm{iw}} x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{IWD})$. By induction we have the existence of basic terms b_1 and b_2 such that $x_1 = b_1$ and $x_2 = b_2$. By induction on the structure of basic term b_1.

(a) $b_1 = \varepsilon$. Then $x = x \circ_{\mathrm{iw}} y = b_1 \circ_{\mathrm{iw}} b_2 = \varepsilon \circ_{\mathrm{iw}} b_2 = b_2$ which is a basic term.

(b) $b_1 = \delta_E$ for some $E \subseteq EID$. By induction on the structure of basic term b_2.

i. $b_2 = \varepsilon$. Then $x = x_1 \circ_{\mathrm{iw}} x_2 = \delta_E \circ_{\mathrm{iw}} \varepsilon = \delta_E$, which is a basic term.

ii. $b_2 = \delta_F$ for some $F \subseteq EID$. Then $x = x_1 \circ_{\mathrm{iw}} x_2 = b_1 \circ_{\mathrm{iw}} b_2 = \delta_E \circ_{\mathrm{iw}} \delta_F = \delta_{E \cup F}$, which is a basic term.

iii. $b_2 = a_2 \circ_{\mathrm{iw}} b_2'$ for some $a_2 \in A$ and $b_2' \in \mathcal{B}(\Sigma_{IWD})$. By induction we have the existence of a basic term c such that $\delta_E \circ_{\mathrm{iw}} b_2' = c$. Also by induction we have the existence of a basic term c' such that $\delta_{E \cup AE(a_2)} \circ_{\mathrm{iw}} b_2' = c'$. If $AE(a_2) \cap E \neq \emptyset$, then $x = x_1 \circ_{\mathrm{iw}} x_2 = b_1 \circ_{\mathrm{iw}} b_2 = \delta_E \circ_{\mathrm{iw}} (a_2 \circ_{\mathrm{iw}} b_2') = (\delta_E \circ_{\mathrm{iw}} a_2) \circ_{\mathrm{iw}} b_2' = \delta_{E \cup AE(a_2)} \circ_{\mathrm{iw}} b_2' = c'$, which is a basic term. If $AE(a) \cap E = \emptyset$, then $x = x_1 \circ_{\mathrm{iw}} x_2 = b_1 \circ_{\mathrm{iw}} b_2 = \delta_E \circ_{\mathrm{iw}} (a_2 \circ_{\mathrm{iw}} b_2') = (\delta_E \circ_{\mathrm{iw}} a_2) \circ_{\mathrm{iw}} b_2' = (a_2 \circ_{\mathrm{iw}} \delta_E) \circ_{\mathrm{iw}} b_2' = a_2 \circ_{\mathrm{iw}} (\delta_E \circ_{\mathrm{iw}} b_2') = a_2 \circ_{\mathrm{iw}} c$ which is a basic term.

(c) $b_1 = a_1 \circ_{\mathrm{iw}} b_1'$ for some $a_1 \in A$ and $b_1' \in \mathcal{B}(\Sigma_{IWD})$. By induction we have the existence of a basic term c such that $b_1' \circ_{\mathrm{iw}} b_2 = c$. Then $x = x_1 \circ_{\mathrm{iw}} x_2 = b_1 \circ_{\mathrm{iw}} b_2 = (a_1 \circ_{\mathrm{iw}} b_1') \circ_{\mathrm{iw}} b_2 = a_1 \circ_{\mathrm{iw}} (b_1' \circ_{\mathrm{iw}} b_2) = a_1 \circ_{\mathrm{iw}} c$, which is a basic term.

⊠

The next step towards the proof of completeness is the following lemma. It provides a link between axiomatic reasoning and reasoning in the (operational) model. The proof of this lemma requires the notion of *norm* of a closed term. It counts the number of actions and sequencing operators occurring in the term.

Definition 6 (Norm) For $E \subseteq EID$, $a \in A$, and $x, y \in \mathcal{C}(\Sigma_{IWD})$ we define the mapping $|\_| : \mathcal{C}(\Sigma_{IWD}) \to I\!N$ inductively as follows:

$$\begin{array}{lcl} |\varepsilon| & = & 0, \\ |\delta_E| & = & 0, \\ |a| & = & 1, \\ |x \circ_{\mathrm{iw}} y| & = & |x| + |y| + 1. \end{array}$$

Lemma 2 For all $x, x' \in \mathcal{C}(\Sigma_{IWD})$ and $a \in A$ we have

1. if $x \xrightarrow{a} x'$, then $|x'| < |x|$;
2. if $x \downarrow$, then $x = \varepsilon$;
3. if $x \xrightarrow{a} x'$, then $x = a \circ_{\mathrm{iw}} x'$;
4. if $x \not\downarrow$, $x \nrightarrow$, then $x = \delta_{AE(x)}$.

Proof.

1. By induction on the structure of closed term x. Suppose $x \xrightarrow{a} x'$.

(a) $x \equiv \varepsilon$. This case cannot occur.

(b) $x \equiv \delta_E$ for some $E \subseteq EID$. This case cannot occur.

(c) $x \equiv b$ for some $b \in A$. Then necessarily $b \equiv a$ and $x' \equiv \varepsilon$. Observe that

$$|x'| = |\varepsilon| = 0 < 1 = |b| = |x|.$$

(d) $x \equiv x_1 \circ_{\mathrm{iw}} x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{IWD})$. We can distinguish two cases for $x_1 \circ_{\mathrm{iw}} x_2 \xrightarrow{a} x'$.

i. $x_1 \xrightarrow{a} x_1'$ for some $x_1' \in \mathcal{C}(\Sigma_{IWD})$ such that $x' \equiv x_1' \circ_{\mathrm{iw}} x_2$. By induction we have that $|x_1'| < |x_1|$. Thus we obtain

$$|x'| = |x_1' \circ_{\mathrm{iw}} x_2| = |x_1'| + |x_2| + 1 < |x_1| + |x_2| + 1 = |x_1 \circ_{\mathrm{iw}} x_2| = |x|.$$

ii. $AE(a) \cap AE(x_1) = \emptyset$ and $x_2 \xrightarrow{a} x_2'$ for some $x_2' \in \mathcal{C}(\Sigma_{IWD})$ such that $x' \equiv x_1 \circ_{\mathrm{iw}} x_2'$. By induction we have that $|x_2'| < |x_2|$. Thus we obtain

$$|x'| = |x_1 \circ_{\mathrm{iw}} x_2'| = |x_1| + |x_2'| + 1 < |x_1| + |x_2| + 1 = |x_1 \circ_{\mathrm{iw}} x_2| = |x|.$$

2. By induction on the structure of closed term x. Suppose $x\downarrow$.

(a) $x \equiv \varepsilon$. Trivial.

(b) $x \equiv \delta_E$ for some $E \subseteq EID$. This case cannot occur.

(c) $x \equiv a$ for some $a \in A$. This case cannot occur.

(d) $x \equiv x_1 \circ_{\mathrm{iw}} x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{IWD})$. As $x\downarrow$, we have $x_1\downarrow$ and $x_2\downarrow$. By induction we then have $x_1 = \varepsilon$ and $x_2 = \varepsilon$. Then $x = x_1 \circ_{\mathrm{iw}} x_2 = \varepsilon \circ_{\mathrm{iw}} \varepsilon = \varepsilon$.

3. By induction on the structure of closed term x. Suppose $x \xrightarrow{a} x'$.

(a) $x \equiv \varepsilon$. This case cannot occur.

(b) $x \equiv \delta_E$ for some $E \subseteq EID$. This case cannot occur.

(c) $x \equiv b$ for some $b \in A$. Then necessarily $b \equiv a$ and $x' \equiv \varepsilon$. Then,

$$x = b = a = a \circ_{\mathrm{iw}} \varepsilon = a \circ_{\mathrm{iw}} x'.$$

(d) $x \equiv x_1 \circ_{\mathrm{iw}} x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{IWD})$. For $x_1 \circ_{\mathrm{iw}} x_2 \xrightarrow{a} x'$ two cases can be distinguished:

i. $x_1 \xrightarrow{a} x_1'$ for some $x_1' \in \mathcal{C}(\Sigma_{IWD})$ such that $x' \equiv x_1' \circ_{\mathrm{iw}} x_2$. By induction we then have $x_1 = a \circ_{\mathrm{iw}} x_1'$. Then,

$$x = x_1 \circ_{\mathrm{iw}} x_2 = (a \circ_{\mathrm{iw}} x_1') \circ_{\mathrm{iw}} x_2 = a \circ_{\mathrm{iw}} (x_1' \circ_{\mathrm{iw}} x_2) = a \circ_{\mathrm{iw}} x'.$$

ii. $x_2 \xrightarrow{a} x_2'$ and $AE(a) \cap AE(x_1) = \emptyset$ for some $x_2' \in \mathcal{C}(\Sigma_{IWD})$ such that $x' \equiv x_1 \circ_{\mathrm{iw}} x_2'$. By induction we have $x_2 = a \circ_{\mathrm{iw}} x_2'$. Then,

$$\begin{aligned} x = x_1 \circ_{\mathrm{iw}} x_2 &= x_1 \circ_{\mathrm{iw}} (a \circ_{\mathrm{iw}} x_2') = (x_1 \circ_{\mathrm{iw}} a) \circ_{\mathrm{iw}} x_2' \\ &= (a \circ_{\mathrm{iw}} x_1) \circ_{\mathrm{iw}} x_2' = a \circ_{\mathrm{iw}} (x_1 \circ_{\mathrm{iw}} x_2') = a \circ_{\mathrm{iw}} x'. \end{aligned}$$

4. By induction on $|x|$ and case analysis on the structure of x. Suppose $x \not\downarrow$ and $x \nrightarrow$.

   (a) $x \equiv \varepsilon$. This case cannot occur.

   (b) $x \equiv \delta_E$ for some $E \subseteq EID$. Trivial as $AE(x) = AE(\delta_E) = E$ and $x = \delta_E = \delta_{AE(x)}$.

   (c) $x \equiv b$ for some $b \in A$. This case cannot occur as $b \xrightarrow{b}$ contradicts the assumption that $x \nrightarrow$.

   (d) $x \equiv x_1 \circ_{\mathrm{iw}} x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{IWD})$. If $x_1 \downarrow$ then we find $x_1 = \varepsilon$. As $x \not\downarrow$, we also find $x_2 \not\downarrow$. As $x_1 \circ_{\mathrm{iw}} x_2 \nrightarrow$, we find $x_1 \nrightarrow$, and $x_2 \xrightarrow{a} \vee\, AE(a) \cap AE(x_1) \neq \emptyset$ for all $a \in A$. As $x_1 = \varepsilon$, we find $AE(a) \cap AE(x_1) = AE(a) \cap AE(\varepsilon) = \varepsilon$. Therefore, we must have $x_2 \xrightarrow{a}$. By induction (note that $|x_2| < |x|$) we thus have $x_2 = \delta_{AE(x_2)}$. Then

$$x = x_1 \circ_{\mathrm{iw}} x_2 = \varepsilon \circ_{\mathrm{iw}} \delta_{AE(x_2)} = \delta_{AE(x_2)} = \delta_{AE(x_1) \cup AE(x_2)} = \delta_{AE(x)}.$$

   If $x_1 \not\downarrow$, then we have by induction $x_1 = \delta_{AE(x_1)}$ as we also have $x_1 \xrightarrow{a}$. First, suppose that $x_2 \downarrow$. Then $x_2 = \varepsilon$ and we obtain $x = x_1 \circ_{\mathrm{iw}} x_2 = \delta_{AE(x_1)} \circ_{\mathrm{iw}} \varepsilon = \delta_{AE(x_1)} = \delta_{AE(x_1) \cup AE(x_2)} = \delta_{AE(x)}$. Second, suppose $x_2 \not\downarrow$. Again we can distinguish two cases:

      i. $x_2 \xrightarrow{a}$ for all $a \in A$. As $|x_2| < |x|$, we can apply the induction hypothesis and obtain $x_2 = \delta_{AE(x_2)}$. Thus,

$$x = x_1 \circ_{\mathrm{iw}} x_2 = \delta_{AE(x_1)} \circ_{\mathrm{iw}} \delta_{AE(x_2)} = \delta_{AE(x_1) \cup AE(x_2)} = \delta_{AE(x)}.$$

      ii. $x_2 \xrightarrow{a} x_2'$ for some $a \in A$. Then we have $x_2 = a \circ_{\mathrm{iw}} x_2'$. As $x_1 \circ_{\mathrm{iw}} x_2 \xrightarrow{a}$, we must have $AE(a) \cap AE(x_1) \neq \emptyset$. Then,

$$\begin{aligned} x = x_1 \circ_{\mathrm{iw}} x_2 &= \delta_{AE(x_1)} \circ_{\mathrm{iw}} (a \circ_{\mathrm{iw}} x_2') = (\delta_{AE(x_1)} \circ_{\mathrm{iw}} a) \circ_{\mathrm{iw}} x_2' \\ &= \delta_{AE(x_1) \cup AE(a)} \circ_{\mathrm{iw}} x_2'. \end{aligned}$$

      Note that $|x_2'| < |x_2|$. Observe that

$$\begin{aligned} |\delta_{AE(x_1) \cup AE(a)} \circ_{\mathrm{iw}} x_2'| &= |x_2'| + 1 < |x_2| + 1 \leq |x_1| + |x_2| + 1 \\ &= |x_1 \circ_{\mathrm{iw}} x_2|. \end{aligned}$$

      Hence we can apply the induction hypothesis to obtain

$$\begin{aligned} \delta_{AE(x_1) \cup AE(a)} \circ_{\mathrm{iw}} x_2' &= \delta_{AE(x_1) \cup AE(a) \cup AE(x_2')} = \delta_{AE(x_1) \cup AE(x_2)} \\ &= \delta_{AE(x)}. \end{aligned}$$

⊠

Theorem 5 (Completeness) *IWD*(∘) is a complete axiomatisation of IWD-bisimilarity on closed *IWD*(∘)-terms.

Proof. Suppose that $x \underline{\leftrightarrow}_{\mathrm{iwd}} y$. Then we must prove that $x = y$. By the elimination theorem we have the existence of a basic term x' such that $x = x'$. As the axioms are sound, we also have $x \underline{\leftrightarrow}_{\mathrm{iwd}} x'$. Hence it suffices to prove $x' = y$. We do this by induction on the structure of basic term x'.

1. $x' = \varepsilon$. Then $x' \downarrow$. Since $x' \underline{\leftrightarrow}_{\mathrm{iwd}} y$, we also have $y \downarrow$. By Lemma 2.2 we then have $y = \varepsilon$. Hence $x' = \varepsilon = y$.

2. $x' = \delta_E$ for some $E \subseteq EID$. Then $x' \not\downarrow$ and $x' \stackrel{a}{\not\rightarrow}$ for all $a \in A$. As $x' \underline{\leftrightarrow}_{\mathrm{iwd}} y$, also $y \not\downarrow$ and $y \stackrel{a}{\not\rightarrow}$. We also have $AE(y) = AE(x') = AE(\delta_E) = E$. By Lemma 2.4 we have $y = \delta_E$. So $x' = \delta_E = y$.

3. $x' = a \circ_{\mathrm{iw}} x''$ for some $a \in A$ and $x'' \in \mathcal{B}(\Sigma_{IWD})$. Then $x' \stackrel{a}{\rightarrow} \varepsilon \circ_{\mathrm{iw}} x''$. Since $x' \underline{\leftrightarrow}_{\mathrm{iwd}} y$ we also have $y \stackrel{a}{\rightarrow} y'$ for some y' such that $\varepsilon \circ_{\mathrm{iw}} x'' \underline{\leftrightarrow}_{\mathrm{iwd}} y'$. Then, using transitivity of IWD-bisimilarity and the soundness of Idem. $\circ_{\mathrm{iw}}$, also $x'' \underline{\leftrightarrow}_{\mathrm{iwd}} y'$. By induction we then have $x'' = y'$. By Lemma 2.3 we have $y = a \circ_{\mathrm{iw}} y'$. Then $x' = a \circ_{\mathrm{iw}} x'' = a \circ_{\mathrm{iw}} y' = y$.

⊠

4 Sequential and alternative composition

In the previous section we have defined a sound and complete axiomatisation of Interworking diagrams. For this purpose we needed to introduce the interworking sequencing operator only. If we want to extend this theory with other operators, we first have to introduce the Basic Process Algebra operators $+$ and $\cdot$. This section is devoted to the development of the process algebra $BPA(\delta_E)$ without interworking sequencing. In the next section, the interworking sequencing is added to this algebra.

The $+$ is called *alternative composition* and $\cdot$ is called *sequential composition*. The process $x + y$ can execute either process x or process y, but not both. The process $x \cdot y$ starts executing process x, and upon termination thereof starts the execution of process y. Operationally these operators are described by the deduction rules given in Table 3. In this table we assume that $a \in A$ and $E \subseteq EID$. The theory presented in this section is very similar to standard Basic Process Algebra with deadlock and empty process $BPA_{\delta\varepsilon}$ (see e.g. [BV95]).

Definition 7 (Active entities) For $i, j \in EID$, $m \in MID$, $E \subseteq EID$, $x, y \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$, and $\odot \in \{+, \cdot\}$, we define the mapping $AE : \mathcal{C}(\Sigma_{BPA(\delta_E)}) \to I\!P(EID)$ inductively as follows:

$$\begin{array}{lcl} AE(c(i,j,m)) & = & \{i,j\}, \\ AE(\varepsilon) & = & \emptyset, \\ AE(\delta_E) & = & E, \\ AE(x \odot y) & = & AE(x) \cup AE(y). \end{array}$$

The alternative composition of two terms can terminate if either one of these terms can terminate. It can perform every action that its operands can perform, but by doing so a choice is made. A sequential composition can terminate if both operands can terminate. It can perform all actions from its first operand and if the first operand can terminate, it can perform the actions from the second operand.

$$\frac{x\downarrow}{x+y\downarrow} \qquad \frac{y\downarrow}{x+y\downarrow} \qquad \frac{x\downarrow \quad y\downarrow}{x\cdot y\downarrow}$$

$$\frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x+y \xrightarrow{a} y'} \qquad \frac{x \xrightarrow{a} x'}{x\cdot y \xrightarrow{a} x'\cdot y} \qquad \frac{x\downarrow \quad y \xrightarrow{a} y'}{x\cdot y \xrightarrow{a} y'}$$

Table 3: Deduction rules for alternative and sequential composition ($a \in A$)

Again, we first need to prove that IWD-bisimilarity is a congruence for all operators in the process algebra.

Theorem 6 (Congruence) IWD-bisimilarity is a congruence for alternative composition and sequential composition.

Proof. The term deduction system for $BPA(\delta_E)$ is in path format. From [BV93], we then have that IWD-bisimilarity is a congruence for all operators. ⊠

These operators are axiomatised by the axioms from Table 4. In these axioms the variables x, y and z denote arbitrary process terms. In order to reduce the number of parentheses in processes we have the following priorities on operators: unary operators bind stronger that binary operators; $\cdot$ binds stronger than all other binary operators and $+$ binds weaker than all other binary operators.

A1	$x + y = y + x$	
A2	$(x + y) + z = x + (y + z)$	
A3	$x + x = x$	
A4	$(x + y) \cdot z = x \cdot z + y \cdot z$	
A5	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	
A6	$x + \delta_E = x$	if $E \subseteq AE(x)$
A7	$\delta_E \cdot x = \delta_{E \cup AE(x)}$	
A8	$\varepsilon \cdot x = x$	
A9	$x \cdot \varepsilon = x$	

Table 4: Axioms of alternative and sequential composition ($E \subseteq EID$)

Axioms A1-A5 express straightforward properties, such as commutativity, associativity, and idempotency of alternative composition, distributivity of alternative composition over sequential composition, and associativity of sequential composition. Axioms A6 and A7 characterise the deadlock constant. A6 states that if an entity has the choice between performing an action and deadlocking, it will never deadlock. Axiom A7 expresses that after a deadlock no more actions can occur. The scope of the deadlock is thereby extended to include all entities on which blocked actions occur. Axioms A8 and A9 express the standard behaviour of the empty process.

The proof of soundness is straightforward.

Theorem 7 (Soundness) $BPA(\delta_E)$ is a sound axiomatisation of IWD-bisimilarity on closed $BPA(\delta_E)$-terms.

Proof. In this and other soundness proofs we use I to denote the diagonal relation. If "$s = t$ if b" represents either one of A1, A2, A3, A4, A6, A7, or A8, then the relation $R = \{(s,t),(t,s) \mid b\} \cup I$ is an IWD-bisimulation for that axiom. For the axioms A5 and A9 the IWD-bisimulations are given by $R = \{((p \cdot y) \cdot z, p \cdot (y \cdot z)) \mid p \in \mathcal{C}(\Sigma_{BPA(\delta_E)})\}^S \cup I$ and $R = \{(p \cdot \varepsilon, p) \mid p \in \mathcal{C}(\Sigma_{BPA(\delta_E)})\}^S$ respectively. ⊠

The proof of completeness consists of a number of steps. First we define basic terms and prove the elimination property. Next, we formulate a lemma which relates semantical properties to equational properties, and, finally, we prove completeness.

Definition 8 (Basic terms) The set of basic terms is the smallest set that satisfies:

1. ε is a basic term;
2. for $E \subseteq EID$, δ_E is a basic term;

3. for $a \in A$ and x a basic term, $a \cdot x$ is a basic term;
4. if x and y are basic terms, then $x + y$ is a basic term.

The set of all basic terms of the process algebra $BPA(\delta_E)$ is denoted by $\mathcal{B}(\Sigma_{BPA(\delta_E)})$.

The following lemma expresses that we can always combine multiple deadlock alternatives into one deadlock alternative.

Lemma 3 For $E, F \subseteq EID$ we have $\delta_E + \delta_F = \delta_{E \cup F}$.

Proof. Consider the following derivation: $\delta_E + \delta_F = (\delta_E + \delta_F) + \delta_{E \cup F} = \delta_{E \cup F} + (\delta_E + \delta_F) = (\delta_{E \cup F} + \delta_E) + \delta_F = \delta_{E \cup F} + \delta_F = \delta_{E \cup F}$. ⊠

As alternative composition is idempotent, commutative and associative, and $\delta_\emptyset$ is a zero element for it, we can define a generalised alternative composition operator. For finite index set I, the notation $\sum_{i \in I} x_i$ represents the alternative composition of the process terms x_i. If $I = \emptyset$, then $\sum_{i \in I} x_i = \delta_\emptyset$. If $I = \{i_1, \cdots, i_n\}$ for $n \geq 1$, then

$$\sum_{i \in I} x_i = x_{i_1} + x_{i_2} + \cdots + x_{i_n}.$$

Then we can easily establish that every basic term is of the form

$$\sum_{i \in I} a_i \cdot x_i + \sum_{j \in J} \delta_{E_j} + \sum_{k \in K} \varepsilon$$

for some finite index sets I, J, K, $a_i \in A$, $E_j \subseteq EID$ and basic terms x_i of a similar form. For convenience in proofs to follow we combine the deadlock alternatives into one alternative by using Lemma 3:

$$\sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon,$$

where $E = \bigcup_{j \in J} E_j$. The summand $\sum_{k \in K} \varepsilon$ is only used to indicate presence ($K \neq \emptyset$) or absence ($K = \emptyset$) of a termination option.

Theorem 8 (Elimination) For every closed term there is a derivably equal basic term.

Proof. We prove this theorem by induction on the structure of closed term x.

1. $x \equiv \varepsilon$. Trivial as ε is a basic term.
2. $x \equiv \delta_E$ for some $E \subseteq EID$. Trivial as δ_E is a basic term.

3. $x \equiv a$ for some $a \in A$. Then $x = a = a \cdot \varepsilon$, which is a basic term.

4. $x \equiv x_1 \cdot x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$. By induction we have the existence of basic terms b_1 and b_2 such that $x_1 = b_1$ and $x_2 = b_2$. By induction on the structure of basic term b_1 we will prove that there exists a basic term c such that $b_1 \cdot b_2 = c$.

   (a) $b_1 \equiv \varepsilon$. Then $b_1 \cdot b_2 = \varepsilon \cdot b_2 = b_2$.

   (b) $b_1 \equiv \delta_E$ for some $E \subseteq EID$. Then $b_1 \cdot b_2 = \delta_E \cdot b_2 = \delta_{E \cup AE(b_2)}$.

   (c) $b_1 \equiv a \cdot b_1'$ for some $a \in A$ and $b_1' \in \mathcal{B}(\Sigma_{BPA(\delta_E)})$. By induction we have the existence of basic term c' such that $b_1' \cdot b_2 = c_1'$. Then $b_1 \cdot b_2 = (a \cdot b_1') \cdot b_2 = a \cdot (b_1' \cdot b_2) = a \cdot c_1'$.

   (d) $b_1 \equiv b_1' + b_1''$ for some $b_1', b_1'' \in \mathcal{B}(\Sigma_{BPA(\delta_E)})$. By induction we have the existence of basic terms c' and c'' such that $b_1' \cdot b_2 = c'$ and $b_1'' \cdot b_2 = c_2$. Then $b_1 \cdot b_2 = (b_1' + b_1'') \cdot b_2 = b_1' \cdot b_2 + b_1'' \cdot b_2 = c' + c''$.

   Observe that in each case we have the existence of basic term c such that $b_1 \cdot b_2 = c$. Hence $x = x_1 \cdot x_2 = b_1 \cdot b_2 = c$, which is a basic term.

5. $x \equiv x_1 + x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$. By induction we have the existence of basic terms b_1 and b_2 such that $x_1 = b_1$ and $x_2 = b_2$. Then $x = x_1 + x_2 = b_1 + b_2$, which is a basic term.

$\boxtimes$

Lemma 4 For all $x, x' \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$ and $a \in A$ we have

1. if $x \downarrow$, then $x = \varepsilon + x$;
2. if $x \xrightarrow{a} x'$, then $x = a \cdot x' + x$.

Proof.

1. We will prove this by induction on the structure of x. Suppose $x \downarrow$.

   (a) $x \equiv \varepsilon$. Then trivially $x = \varepsilon = \varepsilon + \varepsilon = \varepsilon + x$.

   (b) $x \equiv \delta_E$ for some $E \subseteq EID$. Then we have a contradiction as $\delta_E \not\downarrow$.

   (c) $x \equiv a$ for some $a \in A$. Then we also have a contradiction as $a \not\downarrow$.

   (d) $x \equiv x_1 + x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$. Then we have $x_1 \downarrow$ or $x_2 \downarrow$. By induction we then have $x_1 = \varepsilon + x_1$ or $x_2 = \varepsilon + x_2$. In both cases we find $x = x_1 + x_2 = \varepsilon + x_1 + x_2 = \varepsilon + x$.

   (e) $x \equiv x_1 \cdot x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$. Then we have $x_1 \downarrow$ and $x_2 \downarrow$. By induction we then have $x_1 = \varepsilon + x_1$ and $x_2 = \varepsilon + x_2$. Therefore, $x = x_1 \cdot x_2 = (\varepsilon + x_1) \cdot x_2 = \varepsilon \cdot x_2 + x_1 \cdot x_2 = x_2 + x_1 \cdot x_2 = \varepsilon + x_2 + x_1 \cdot x_2 = \varepsilon + \varepsilon \cdot x_2 + x_1 \cdot x_2 = \varepsilon + (\varepsilon + x_1) \cdot x_2 = \varepsilon + x_1 \cdot x_2 = \varepsilon + x$.

2. We will prove this by induction on the structure of x. Suppose $x \xrightarrow{a} x'$.

   (a) $x \equiv \varepsilon$. Then we have a contradiction as $\varepsilon \not\xrightarrow{a}$.

   (b) $x \equiv \delta_E$ for some $E \subseteq EID$. Then we also have a contradiction as $\delta_E \not\xrightarrow{a}$.

   (c) $x \equiv b$ for some $b \in A$. Then necessarily $b \equiv a$ and $x' \equiv \varepsilon$. Hence $x = b = b + b = a + b = a \cdot \varepsilon + b = a \cdot x' + x$.

   (d) $x \equiv x_1 + x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$. Then we have $x_1 \xrightarrow{a} x'$ or $x_2 \xrightarrow{a} x'$. By induction we then have $x_1 = a \cdot x' + x_1$ or $x_2 = a \cdot x' + x_2$. Then $x = x_1 + x_2 = a \cdot x' + x_1 + x_2 = a \cdot x' + x$.

   (e) If $x \equiv x_1 \cdot x_2$ for some $x_1, x_2 \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$. We can distinguish two cases.

   First, $x_1 \xrightarrow{a} x_1'$ for some $x_1' \in \mathcal{C}(\Sigma_{BPA(\delta_E)})$ such that $x' \equiv x_1' \cdot x_2$. By induction we then have $x_1 = a \cdot x_1' + x_1$. Therefore, $x = x_1 \cdot x_2 = (a \cdot x_1' + x_1) \cdot x_2 = (a \cdot x_1') \cdot x_2 + x_1 \cdot x_2 = a \cdot (x_1' \cdot x_2) + x = a \cdot x' + x$.

   Second, $x_1 \downarrow$ and $x_2 \xrightarrow{a} x'$. By induction we have $x_2 = a \cdot x' + x_2$. From the first part of this lemma we have $x_1 = \varepsilon + x_1$. Therefore, $x = x_1 \cdot x_2 = (\varepsilon + x_1) \cdot x_2 = \varepsilon \cdot x_2 + x_1 \cdot x_2 = x_2 + x_1 \cdot x_2 = a \cdot x' + x_2 + x_1 \cdot x_2 = a \cdot x' + \varepsilon \cdot x_2 + x_1 \cdot x_2 = a \cdot x' + (\varepsilon + x_1) \cdot x_2 = a \cdot x' + x_1 \cdot x_2 = a \cdot x' + x$.

⊠

Theorem 9 (Completeness) $BPA(\delta_E)$ is a complete axiomatisation of IWD-bisimilarity on closed $BPA(\delta_E)$-terms.

Proof. Suppose that $x \underline{\leftrightarrow}_{\text{iwd}} y$. By the elimination theorem and the soundness of the axioms we can assume, without loss of generality, that x is a basic term. By congruence and the soundness of axiom A3 it suffices to prove that if $x + y \underline{\leftrightarrow}_{\text{iwd}} y$ then $x + y = y$. This can be seen as follows. From $x \underline{\leftrightarrow}_{\text{iwd}} y$ we obtain $x + y \underline{\leftrightarrow}_{\text{iwd}} y + y$ using congruence of $\underline{\leftrightarrow}_{\text{iwd}}$ with respect to $+$, and reflexivity of $\underline{\leftrightarrow}_{\text{iwd}}$. Using the soundness of axiom A3 we have $y + y \underline{\leftrightarrow}_{\text{iwd}} y$. By transitivity of $\underline{\leftrightarrow}_{\text{iwd}}$ we obtain $x + y \underline{\leftrightarrow}_{\text{iwd}} y$. Then $x + y = y$. Similarly we can obtain $y + x = x$. Therefore, $x = y + x = x + y = y$.

We prove this by induction on the structure of basic term x.

1. $x \equiv \varepsilon$. Then $x \downarrow$. So $x + y \downarrow$. Therefore, $y \downarrow$. Then, by Lemma 4.1, we have $y = \varepsilon + y$. So we find $x + y = \varepsilon + y = y$.

2. $x \equiv \delta_E$. Then $AE(x + y) = AE(\delta_E + y) = E \cup AE(y)$. As $x + y \underline{\leftrightarrow}_{\text{iwd}} y$, we must also have $AE(y) = AE(x + y) = E \cup AE(y)$. Thus we obtain $E \subseteq AE(y)$. Then $x + y = \delta_E + y = y + \delta_E = y$.

3. $x \equiv a \cdot x'$. Then $x \xrightarrow{a} \varepsilon \cdot x'$. So $x + y \xrightarrow{a} \varepsilon \cdot x'$. Therefore, $y \xrightarrow{a} y'$ for some y' such that $\varepsilon \cdot x' \underline{\leftrightarrow}_{\text{iwd}} y'$. By the soundness of axiom A8 we find $x' \underline{\leftrightarrow}_{\text{iwd}} y'$. By induction we then have $x' = y'$. By Lemma 4.2 we have $y = a \cdot y' + y$. Then $x + y = a \cdot x' + y = a \cdot y' + y = y$.

4. $x \equiv x_1+x_2$. Using $x_1+x_2+y \underline{\leftrightarrow}_{\mathrm{iwd}}\ y$ implies $x_1+y \underline{\leftrightarrow}_{\mathrm{iwd}}\ y$ and $x_2+y \underline{\leftrightarrow}_{\mathrm{iwd}}\ y$. By induction we then have $x_1+y = y$ and $x_2+y = y$. Then $x+y = (x_1+x_2)+y = x_1+(x_2+y) = x_1+y = y$.

⊠

5 The interworking sequencing

In Section 3 we have introduced the interworking sequencing and in Section 4 we have defined alternative and sequential composition operators. When combining these operators into one single theory, we need to express the relation between the interworking sequencing on the one hand and alternative and sequential composition on the other hand. By introducing the auxiliary operators $\mathbf{L}\!\circ_{\mathrm{iw}}$ and $\mathbf{R}\!\circ_{\mathrm{iw}}$, we can express the interworking sequencing in terms of the alternative and sequential composition operators. The process algebra obtained in this way is called $IWD(\circ, \cdot, +)$. It is a conservative extension of both the process algebra $IWD(\circ)$ from Section 3 and the process algebra $BPA(\delta_E)$ from Section 4. Furthermore, all axioms formulated for interworking sequencing in the theory $IWD(\circ)$ can now be derived for closed terms.

The intuition of the auxiliary operators is as follows. The process $x\,\mathbf{L}\!\circ_{\mathrm{iw}} y$ behaves like the process $x \circ_{\mathrm{iw}} y$ with the restriction that the first action to be executed must originate from process x. The process $x\,\mathbf{R}\!\circ_{\mathrm{iw}} y$ also behaves like the process $x \circ_{\mathrm{iw}} y$ but this time with the restriction that the first action to be executed must be from process y. In this case, the first action from y can only be executed if it is not blocked by any of the actions from x.

These definitions resemble the use of the left-merge operator in PA to define the merge operator. That we need two auxiliary operators instead of one is caused by the fact that interworking sequencing is not commutative.

Definition 9 (Active entities) For $i, j \in EID$, $m \in MID$, $E \subseteq EID$, $x, y \in \mathcal{C}(\Sigma_{IWD(\circ,\cdot,+)})$, and $\odot \in \{\circ_{\mathrm{iw}}, \mathbf{L}\!\circ_{\mathrm{iw}}, \mathbf{R}\!\circ_{\mathrm{iw}}, +, \cdot\}$, we define the mapping $AE : \mathcal{C}(\Sigma_{IWD(\circ,\cdot,+)}) \to \mathbb{P}(EID)$ inductively as follows:

$$
\begin{array}{lcl}
AE(c(i,j,m)) & = & \{i,j\}, \\
AE(\varepsilon) & = & \emptyset, \\
AE(\delta_E) & = & E, \\
AE(x \odot y) & = & AE(x) \cup AE(y).
\end{array}
$$

The operational semantics of the interworking sequencing is given already in Table 1. The operational semantics of the auxiliary operators is given in Table 5. The rules follow from the intuitive explanation above. The termination behaviour of the interworking sequencing is incorporated in both auxiliary operators. This is not necessary but facilitates the axiomatisation of these operators and the proof of the auxiliary proposition in the proof of Proposition 1.

$$\frac{x\downarrow \quad y\downarrow}{x\,\mathbf{L}\circ_{\mathrm{iw}} y\downarrow} \qquad \frac{x\downarrow \quad y\downarrow}{x\,\mathbf{R}\circ_{\mathrm{iw}} y\downarrow}$$

$$\frac{x \xrightarrow{a} x'}{x\,\mathbf{L}\circ_{\mathrm{iw}} y \xrightarrow{a} x' \circ_{\mathrm{iw}} y} \qquad \frac{AE(a)\cap AE(x)=\emptyset \quad y \xrightarrow{a} y'}{x\,\mathbf{R}\circ_{\mathrm{iw}} y \xrightarrow{a} x \circ_{\mathrm{iw}} y'}$$

Table 5: Deduction rules for auxiliary operators for interworking sequencing ($a \in A$)

Theorem 10 (Congruence) IWD-bisimilarity is a congruence for $\mathbf{L}\circ_{\mathrm{iw}}$ and $\mathbf{R}\circ_{\mathrm{iw}}$.

Proof. The term deduction system is in path format and hence IWD-bisimilarity is a congruence for all operators. ⊠

S	$x \circ_{\mathrm{iw}} y = x\,\mathbf{L}\circ_{\mathrm{iw}} y + x\,\mathbf{R}\circ_{\mathrm{iw}} y$	
L1	$\varepsilon\,\mathbf{L}\circ_{\mathrm{iw}}\varepsilon = \varepsilon$	
L2	$\varepsilon\,\mathbf{L}\circ_{\mathrm{iw}}\delta_E = \delta_E$	
L3	$\varepsilon\,\mathbf{L}\circ_{\mathrm{iw}} a\cdot x = \delta_{AE(a\cdot x)}$	
L4	$\varepsilon\,\mathbf{L}\circ_{\mathrm{iw}}(x+y) = \varepsilon\,\mathbf{L}\circ_{\mathrm{iw}} x + \varepsilon\,\mathbf{L}\circ_{\mathrm{iw}} y$	
L5	$\delta_E\,\mathbf{L}\circ_{\mathrm{iw}} x = \delta_{E\cup AE(x)}$	
L6	$a\cdot x\,\mathbf{L}\circ_{\mathrm{iw}} y = a\cdot(x \circ_{\mathrm{iw}} y)$	
L7	$(x+y)\,\mathbf{L}\circ_{\mathrm{iw}} z = x\,\mathbf{L}\circ_{\mathrm{iw}} z + y\,\mathbf{L}\circ_{\mathrm{iw}} z$	
R1-4	$x\,\mathbf{R}\circ_{\mathrm{iw}}\varepsilon = \varepsilon\,\mathbf{L}\circ_{\mathrm{iw}} x$	
R5	$x\,\mathbf{R}\circ_{\mathrm{iw}}\delta_E = \delta_{E\cup AE(x)}$	
R6a	$x\,\mathbf{R}\circ_{\mathrm{iw}} a\cdot y = a\cdot(x \circ_{\mathrm{iw}} y)$	if $AE(a)\cap AE(x)=\emptyset$
R6b	$x\,\mathbf{R}\circ_{\mathrm{iw}} a\cdot y = \delta_{AE(x)\cup AE(a\cdot y)}$	if $AE(a)\cap AE(x)\neq\emptyset$
R7	$x\,\mathbf{R}\circ_{\mathrm{iw}}(y+z) = x\,\mathbf{R}\circ_{\mathrm{iw}} y + x\,\mathbf{R}\circ_{\mathrm{iw}} z$	

Table 6: Axioms for interworking sequencing and auxiliary operators ($a \in A$, $E \subseteq EID$)

The axioms defining the interworking sequencing in terms of alternative and sequential composition are given in Table 6. The first axiom, S, states that the first action from $x \circ_{\mathrm{iw}} y$ can either come from x (via the term $x\,\mathbf{L}\circ_{\mathrm{iw}} y$) or from y (via the term $x\,\mathbf{R}\circ_{\mathrm{iw}} y$). Axioms L1-L7 define

the operator $\mathbf{L}\circ_{\mathrm{iw}}$ using the structure of basic terms. As stated before, $x\,\mathbf{L}\circ_{\mathrm{iw}}y$ behaves like the process $x \circ_{\mathrm{iw}} y$ with the restriction that the first action to be executed comes from x. This is expressed clearly in axiom L6. The relation between $\mathbf{L}\circ_{\mathrm{iw}}$ and $\circ_{\mathrm{iw}}$ also explains the distributive law L7 and the absorption law L5. Axioms L1-L4 define the termination behaviour of $\mathbf{L}\circ_{\mathrm{iw}}$: $x\,\mathbf{L}\circ_{\mathrm{iw}}y$ can only terminate if both operands can terminate. A deadlock occurs if the left operand is ε and the right operand cannot terminate (axioms L2, L3).

The definition of $\mathbf{R}\circ_{\mathrm{iw}}$ is similar. The intuition behind the operator $\mathbf{R}\circ_{\mathrm{iw}}$ is that the right operand may only execute actions which are not blocked by the left operand. Therefore, we make a distinction using the condition $AE(a) \cap AE(x) = \emptyset$ (see axioms R6a and R6b).

Theorem 11 (Soundness) The axioms given in Table 6 are sound with respect to IWD-bisimilarity on closed $IWD(\circ, \cdot, +)$-terms.

Proof. If "$s = t$ if b" represents either one of S, L1-L5, L7, R1-4, R5, R6b, or R7, then the relation $R = \{(s,t),(t,s) \mid b\} \cup I$ is an IWD-bisimulation for that axiom.

For the axioms L6 and R6a the IWD-bisimulations are given by

$$\begin{aligned} R \;=\; & \{(a \cdot x\,\mathbf{L}\circ_{\mathrm{iw}} y, a \cdot (x \circ_{\mathrm{iw}} y)), (\varepsilon \cdot x \circ_{\mathrm{iw}} y, \varepsilon \cdot (x \circ_{\mathrm{iw}} y)), (\varepsilon \cdot x \circ_{\mathrm{iw}} q, x \circ_{\mathrm{iw}} q) \\ & \mid q \in \mathcal{C}(\Sigma_{IWD(\circ,\cdot,+)})\}^S \cup I \end{aligned}$$

and

$$\begin{aligned} R \;=\; & \{(x\,\mathbf{R}\circ_{\mathrm{iw}} a \cdot y, a \cdot (x \circ_{\mathrm{iw}} y)), (x \circ_{\mathrm{iw}} \varepsilon \cdot y, \varepsilon \cdot (x \circ_{\mathrm{iw}} y)), (p \circ_{\mathrm{iw}} \varepsilon \cdot q, p \circ_{\mathrm{iw}} q) \\ & \mid p, q \in \mathcal{C}(\Sigma_{IWD(\circ,\cdot,+)}), AE(a) \cap AE(x) = \emptyset\}^S \cup I \end{aligned}$$

respectively. ⊠

We will consider basic terms as in Definition 8 of Section 4. To prove the elimination property we will need the following lemma.

Lemma 5 For arbitrary basic terms b_1 and b_2 we have the existence of basic terms c_1, c_2 and c_3 such that $b_1\,\mathbf{L}\circ_{\mathrm{iw}} b_2 = c_1$, $b_1\,\mathbf{R}\circ_{\mathrm{iw}} b_2 = c_2$ and $b_1 \circ_{\mathrm{iw}} b_2 = c_3$.

Proof. These statements are proven simultaneously with induction on the structure of basic terms b_1 and b_2. The details of the proofs are omitted. ⊠

Theorem 12 (Elimination) For every closed $IWD(\circ, \cdot, +)$-term x there is a derivably equal basic term s.

Proof. This theorem is proven by induction on the structure of closed $IWD(\circ, \cdot, +)$-term x. The only interesting cases are the following: $x \equiv x'\,\mathbf{L}\circ_{\mathrm{iw}} x''$, $x \equiv x'\,\mathbf{R}\circ_{\mathrm{iw}} x''$, and $x \equiv x' \circ_{\mathrm{iw}} x''$ for closed $IWD(\circ, \cdot, +)$-terms x' and x''. In all cases we find the existence of basic terms b_1 and b_2 such that $x_1 = b_1$ and $x_2 = b_2$. Using the previous lemma we find the desired result. ⊠

Next, we prove that the process algebra $IWD(\circ, \cdot, +)$ is a conservative extension of the process algebra $BPA(\delta_E)$. This means that every equality between closed terms from the signature of $BPA(\delta_E)$ is also derivable from the process algebra $IWD(\circ, \cdot, +)$, and also that in the process algebra $IWD(\circ, \cdot, +)$ only those equalities are derivable. The proof of this theorem uses the approach of [Ver95].

Theorem 13 (Conservativity) The process algebra $IWD(\circ, \cdot, +)$ is a conservative extension of the process algebra $BPA(\delta_E)$.

Proof. The conservativity follows from the following observations:

1. IWD-bisimilarity is definable in terms of predicate and relation symbols only,
2. $BPA(\delta_E)$ is a complete axiomatisation of IWD-bisimilarity on closed $BPA(\delta_E)$-terms,
3. $IWD(\circ, \cdot, +)$ is a sound axiomatisation of IWD-bisimilarity on closed $IWD(\circ, \cdot, +)$-terms (see Theorem 11),
4. The term deduction system for $BPA(\delta_E)$ is pure, well-founded and in path format, and
5. The term deduction system for $IWD(\circ, \cdot, +)$ is in path format.

⊠

Theorem 14 (Completeness) The process algebra $IWD(\circ, \cdot, +)$ is a complete axiomatisation of IWD-bisimilarity on closed $IWD(\circ, \cdot, +)$-terms.

Proof. By the General Completeness Theorem of [Ver94], the completeness of the process algebra $IWD(\circ, \cdot, +)$ follows immediately from the properties mentioned in the proof of Theorem 13 and the fact that $IWD(\circ, \cdot, +)$ has the elimination property for $BPA(\delta_E)$(see Theorem 8). ⊠

In Section 3 we have given a direct axiomatisation of interworking sequencing, while in this section we have expressed interworking sequencing in terms of alternative and sequential composition. We will prove that the axioms used in the direct axiomatisation are still valid in the current setting for closed terms.

As a consequence of the fact that IWD-bisimilarity is a congruence for all operators in the signature and the fact that for every closed term there exists a derivably equal basic term, we can prove equalities for closed terms with induction.

Proposition 1 (Commutativity of $\circ_{\mathbf{iw}}$) For arbitrary closed $IWD(\circ, \cdot, +)$-terms x and y such that $AE(x) \cap AE(y) = \emptyset$ we have

$$x \circ_{\mathrm{iw}} y = y \circ_{\mathrm{iw}} x.$$

Proof. Suppose that $AE(x) \cap AE(y) = \emptyset$. We prove the statements $x \mathbf{L}\circ_{\mathrm{iw}} y = y \mathbf{R}\circ_{\mathrm{iw}} x$ and $x \circ_{\mathrm{iw}} y = y \circ_{\mathrm{iw}} x$ simultaneously with induction on the structure of basic terms x and y. First we present the proof of $x \mathbf{L}\circ_{\mathrm{iw}} y = y \mathbf{R}\circ_{\mathrm{iw}} x$.

1. $x \equiv \varepsilon$. Trivial by axiom R1-4.
2. $x \equiv \delta_E$ for some $E \subseteq EID$. Then $x \mathbf{L}\circ_{\mathrm{iw}} y = \delta_E \mathbf{L}\circ_{\mathrm{iw}} y = \delta_{E \cup AE(y)} = y \mathbf{R}\circ_{\mathrm{iw}} \delta_E = y \mathbf{R}\circ_{\mathrm{iw}} x$.
3. $x \equiv a \cdot x'$ for some $a \in A$ and $x' \in \mathcal{B}(\Sigma_{IWD(\circ,\cdot,+)})$. As $AE(a \cdot x') \cap AE(y) = \emptyset$ implies $AE(x') \cap AE(y) = \emptyset$, we have by induction that $x' \circ_{\mathrm{iw}} y = y \circ_{\mathrm{iw}} x'$. Then
$$\begin{aligned} x \mathbf{L}\circ_{\mathrm{iw}} y &= a \cdot x' \mathbf{L}\circ_{\mathrm{iw}} y = a \cdot (x' \circ_{\mathrm{iw}} y) = a \cdot (y \circ_{\mathrm{iw}} x') \\ &= y \mathbf{R}\circ_{\mathrm{iw}} a \cdot x' = y \mathbf{R}\circ_{\mathrm{iw}} x. \end{aligned}$$
Note that we have also used that $AE(a \cdot x') \cap AE(y) = \emptyset$ implies $AE(a) \cap AE(y) = \emptyset$.
4. $x \equiv x' + x''$ for some $x', x'' \in \mathcal{B}(\Sigma_{IWD(\circ,\cdot,+)})$. As $AE(x' + x'') \cap AE(y) = \emptyset$ implies $AE(x') \cap AE(y) = \emptyset$ and $AE(x'') \cap AE(y) = \emptyset$, we have by induction $x' \mathbf{L}\circ_{\mathrm{iw}} y = y \mathbf{R}\circ_{\mathrm{iw}} x'$ and $x'' \mathbf{L}\circ_{\mathrm{iw}} y = y \mathbf{R}\circ_{\mathrm{iw}} x''$. Then
$$\begin{aligned} x \mathbf{L}\circ_{\mathrm{iw}} y &= (x' + x'') \mathbf{L}\circ_{\mathrm{iw}} y = x' \mathbf{L}\circ_{\mathrm{iw}} y + x'' \mathbf{L}\circ_{\mathrm{iw}} y = y \mathbf{R}\circ_{\mathrm{iw}} x' + y \mathbf{R}\circ_{\mathrm{iw}} x'' \\ &= y \mathbf{R}\circ_{\mathrm{iw}} (x' + x'') = y \mathbf{R}\circ_{\mathrm{iw}} x. \end{aligned}$$

Then we have $x \circ_{\mathrm{iw}} y = x \mathbf{L}\circ_{\mathrm{iw}} y + x \mathbf{R}\circ_{\mathrm{iw}} y = y \mathbf{R}\circ_{\mathrm{iw}} x + y \mathbf{L}\circ_{\mathrm{iw}} x = y \circ_{\mathrm{iw}} x$. ⊠

Proposition 2 (Unit element) For closed $IWD(\circ,\cdot,+)$-terms x we have

$$\varepsilon \circ_{\mathrm{iw}} x = x \circ_{\mathrm{iw}} \varepsilon = x.$$

Proof. First, we prove $\varepsilon \circ_{\mathrm{iw}} x = x$ with induction on the structure of basic term x.

1. $x \equiv \varepsilon$. Then $\varepsilon \circ_{\mathrm{iw}} x = \varepsilon \circ_{\mathrm{iw}} \varepsilon = \varepsilon \mathbf{L}\circ_{\mathrm{iw}} \varepsilon + \varepsilon \mathbf{R}\circ_{\mathrm{iw}} \varepsilon = \varepsilon + \varepsilon = \varepsilon = x$.
2. $x \equiv \delta_E$ for some $E \subseteq EID$. Then $\varepsilon \circ_{\mathrm{iw}} x = \varepsilon \circ_{\mathrm{iw}} \delta_E = \varepsilon \mathbf{L}\circ_{\mathrm{iw}} \delta_E + \varepsilon \mathbf{R}\circ_{\mathrm{iw}} \delta_E = \delta_E + \delta_{E \cup AE(\varepsilon)} = \delta_E = x$.
3. $x \equiv a \cdot x'$ for some $a \in A$ and $x' \in \mathcal{B}(\Sigma_{IWD(\circ,\cdot,+)})$. By induction we have $\varepsilon \circ_{\mathrm{iw}} x' = x'$. Then $\varepsilon \circ_{\mathrm{iw}} x = \varepsilon \circ_{\mathrm{iw}} a \cdot x' = \varepsilon \mathbf{L}\circ_{\mathrm{iw}} a \cdot x' + \varepsilon \mathbf{R}\circ_{\mathrm{iw}} a \cdot x' = \delta_{AE(a \cdot x')} + a \cdot (\varepsilon \circ_{\mathrm{iw}} x') = \delta_{AE(a \cdot x')} + a \cdot x' = a \cdot x'$.
4. $x \equiv x_1 + x_2$ for some $x_1, x_2 \in \mathcal{B}(\Sigma_{IWD(\circ,\cdot,+)})$. By induction we have $\varepsilon \circ_{\mathrm{iw}} x_1 = x_1$ and $\varepsilon \circ_{\mathrm{iw}} x_2 = x_2$. Then $\varepsilon \circ_{\mathrm{iw}} x = \varepsilon \circ_{\mathrm{iw}} (x_1 + x_2) = \varepsilon \mathbf{L}\circ_{\mathrm{iw}} (x_1 + x_2) + \varepsilon \mathbf{R}\circ_{\mathrm{iw}} (x_1 + x_2) = \varepsilon \mathbf{L}\circ_{\mathrm{iw}} x_1 + \varepsilon \mathbf{L}\circ_{\mathrm{iw}} x_2 + \varepsilon \mathbf{R}\circ_{\mathrm{iw}} x_1 + \varepsilon \mathbf{R}\circ_{\mathrm{iw}} x_2 = \varepsilon \circ_{\mathrm{iw}} x_1 + \varepsilon \circ_{\mathrm{iw}} x_2 = x_1 + x_2 = x$.

Then, using the commutativity of $\circ_{\mathrm{iw}}$ and the fact that $AE(x) \cap AE(\varepsilon) = \emptyset$ we easily find $x \circ_{\mathrm{iw}} \varepsilon = \varepsilon \circ_{\mathrm{iw}} x = x$. $\boxtimes$

Proposition 3 (Associativity of $\circ_{\mathrm{iw}}$) For closed $IWD(\circ, \cdot, +)$-terms x, y, and z, we have

$$(x \circ_{\mathrm{iw}} y) \circ_{\mathrm{iw}} z = x \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z).$$

Proof. Without loss of generality we can assume that x, y, and z are basic terms. To prove this theorem the following propositions are proven simultaneously with induction on the general form of the basic terms x, y, and z.

$$(x \,\mathbf{L}\circ_{\mathrm{iw}} y) \,\mathbf{L}\circ_{\mathrm{iw}} z = x \,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) \tag{1}$$
$$(x \,\mathbf{R}\circ_{\mathrm{iw}} y) \,\mathbf{L}\circ_{\mathrm{iw}} z = x \,\mathbf{R}\circ_{\mathrm{iw}} (y \,\mathbf{L}\circ_{\mathrm{iw}} z) \tag{2}$$
$$(x \circ_{\mathrm{iw}} y) \,\mathbf{R}\circ_{\mathrm{iw}} z = x \,\mathbf{R}\circ_{\mathrm{iw}} (y \,\mathbf{R}\circ_{\mathrm{iw}} z) \tag{3}$$
$$(x \circ_{\mathrm{iw}} y) \circ_{\mathrm{iw}} z = x \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) \tag{4}$$

This way of proving associativity of interworking sequencing is similar to the way in which associativity of parallel composition is proven in ACP. Similar equations in the setting of ACP are usually called the Axioms of Standard Concurrency [BW90].

Let

$$x = \sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon,$$
$$y = \sum_{l \in L} b_l \cdot y_l + \delta_F + \sum_{n \in N} \varepsilon,$$
$$z = \sum_{o \in O} c_o \cdot z_o + \delta_G + \sum_{q \in Q} \varepsilon,$$

for some finite index sets I, K, L, N, O, Q, $a_i, b_l, c_o \in A$, $E, F, G \subseteq EID$ and basic terms x_i, y_l, z_o.

The following identities are used in the proofs of these four equations. Their proofs are omitted.

$$(\varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} y) \,\mathbf{L}\circ_{\mathrm{iw}} z = \varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) \tag{a}$$
$$(x \,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon) \,\mathbf{L}\circ_{\mathrm{iw}} z = x \,\mathbf{R}\circ_{\mathrm{iw}} (\varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} z) \tag{b}$$
$$(x \circ_{\mathrm{iw}} y) \,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon = x \,\mathbf{R}\circ_{\mathrm{iw}} (y \,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon) \tag{c}$$

Proof of (1):

$$
\begin{aligned}
& (x \,\mathbf{L}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z \\
= \;& \{\text{ass. } x,\ \text{distribution laws}\} \\
& \sum_{i\in I}(a_i \cdot x_i \,\mathbf{L}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z + (\delta_E \,\mathbf{L}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z + \sum_{k\in K}(\varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z \\
= \;& \{\text{L6, L5}\} \\
& \textstyle\sum_{i\in I} a_i \cdot ((x_i \circ_{\mathrm{iw}} y) \circ_{\mathrm{iw}} z) + \delta_{E\cup AE(y)\cup AE(z)} + \sum_{k\in K}(\varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z \\
= \;& \{\text{Induction hypothesis (4), } AE(y)\cup AE(z) = AE(y\circ_{\mathrm{iw}} z),\ \text{(a)}\} \\
& \textstyle\sum_{i\in I} a_i \cdot (x_i \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z)) + \delta_{E\cup AE(y\circ_{\mathrm{iw}} z)} + \sum_{k\in K} \varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) \\
= \;& \{\text{L6, L5}\} \\
& \textstyle\sum_{i\in I} a_i \cdot x_i \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) + \delta_E \,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) + \sum_{k\in K} \varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) \\
= \;& \{\text{distribution laws, ass. } x\} \\
& x \,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z)
\end{aligned}
$$

Proof of (2): Let $L' = \{l \in L \mid AE(b_l) \cap AE(x) = \emptyset\}$ and $L'' = L \setminus L'$.

$$
\begin{aligned}
& (x \,\mathbf{R}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z \\
= \;& \{\text{ass. } y,\ \text{distribution laws}\} \\
& \sum_{l\in L}(x \,\mathbf{R}\circ_{\mathrm{iw}} b_l \cdot y_l)\,\mathbf{L}\circ_{\mathrm{iw}} z + (x \,\mathbf{R}\circ_{\mathrm{iw}} \delta_F)\,\mathbf{L}\circ_{\mathrm{iw}} z + \sum_{n\in N}(x \,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon)\,\mathbf{L}\circ_{\mathrm{iw}} z \\
= \;& \{\text{R6a, R6b, L6, R5, L5}\} \\
& \sum_{l\in L'} b_l \cdot ((x \circ_{\mathrm{iw}} y_l) \circ_{\mathrm{iw}} z) + \sum_{l\in L''} \delta_{AE(x)\cup AE(b_l\cdot y_l)\cup AE(z)} + \delta_{F\cup AE(x)\cup AE(z)} \\
& + \sum_{n\in N}(x \,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon)\,\mathbf{L}\circ_{\mathrm{iw}} z \\
= \;& \{\text{Induction hypothesis (4), (b)}\} \\
& \sum_{l\in L'} b_l \cdot (x \circ_{\mathrm{iw}} (y_l \circ_{\mathrm{iw}} z)) + \sum_{l\in L''} \delta_{AE(x)\cup AE(b_l\cdot y_l)\cup AE(z)} + \delta_{F\cup AE(x)\cup AE(z)} \\
& + \sum_{n\in N} x \,\mathbf{R}\circ_{\mathrm{iw}} (\varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} z) \\
= \;& \{\text{R6a, R6b, L6, R5, L5}\} \\
& \sum_{l\in L} x \,\mathbf{R}\circ_{\mathrm{iw}} (b_l \cdot y_l \,\mathbf{L}\circ_{\mathrm{iw}} z) + x \,\mathbf{R}\circ_{\mathrm{iw}} (\delta_F \,\mathbf{L}\circ_{\mathrm{iw}} z) + \sum_{n\in N} x \,\mathbf{R}\circ_{\mathrm{iw}} (\varepsilon \,\mathbf{L}\circ_{\mathrm{iw}} z) \\
= \;& \{\text{distribution laws, ass. } y\} \\
& x \,\mathbf{R}\circ_{\mathrm{iw}} (y \,\mathbf{R}\circ_{\mathrm{iw}} z)
\end{aligned}
$$

Proof of (3): Let $O' = \{o \in O \mid AE(c_o) \cap AE(x \circ_{\mathrm{iw}} y) = \emptyset\}$ and $O'' = O \setminus O'$.

$$
\begin{array}{ll}
 & (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} z \\
= & \{\text{ass. } z,\ \text{distribution laws}\} \\
 & \sum_{o\in O} (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} c_o \cdot z_o + (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} \delta_G + \sum_{q\in Q} (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon \\
= & \{\mathrm{R6a},\ \mathrm{R6b}, \mathrm{R5}\} \\
 & \sum_{o\in O'} c_o \cdot ((x \circ_{\mathrm{iw}} y) \circ_{\mathrm{iw}} z_o) + \sum_{o\in O''} \delta_{AE(x\circ_{\mathrm{iw}} y)\cup AE(c_o\cdot z_o)} + \delta_{G\cup AE(x\circ_{\mathrm{iw}} y)} \\
 & + \sum_{q\in Q} (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon \\
= & \{\text{Induction hypothesis (4)},\ AE(x \circ_{\mathrm{iw}} y) = AE(x) \cup AE(y),\ \text{(c)}\} \\
 & \sum_{o\in O'} c_o \cdot (x \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z_o)) + \sum_{o\in O''} \delta_{AE(x)\cup AE(y)\cup AE(c_o\cdot z_o)} \\
 & + \delta_{G\cup AE(x)\cup AE(y)} + \sum_{q\in Q} x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon) \\
= & \{\mathrm{R6a},\ \mathrm{R6b},\ \mathrm{R5}\} \\
 & \sum_{o\in O} x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{R}\circ_{\mathrm{iw}} c_o \cdot z_o) + x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{R}\circ_{\mathrm{iw}} \delta_G) + \sum_{q\in Q} x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{R}\circ_{\mathrm{iw}} \varepsilon) \\
= & \{\text{distribution laws},\ \text{ass. } z\} \\
 & x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{R}\circ_{\mathrm{iw}} z)
\end{array}
$$

Proof of (4):

$$
\begin{array}{ll}
 & (x \circ_{\mathrm{iw}} y) \circ_{\mathrm{iw}} z \\
= & \{\mathrm{S}\} \\
 & (x \circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z + (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} z \\
= & \{\mathrm{S}\} \\
 & (x\,\mathbf{L}\circ_{\mathrm{iw}} y + x\,\mathbf{R}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z + (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} z \\
= & \{\mathrm{L7}\} \\
 & (x\,\mathbf{L}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z + (x\,\mathbf{R}\circ_{\mathrm{iw}} y)\,\mathbf{L}\circ_{\mathrm{iw}} z + (x \circ_{\mathrm{iw}} y)\,\mathbf{R}\circ_{\mathrm{iw}} z \\
= & \{\text{Induction hypotheses (1), (2), (3)}\} \\
 & x\,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) + x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{L}\circ_{\mathrm{iw}} z) + x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{R}\circ_{\mathrm{iw}} z) \\
= & \{\mathrm{R7}\} \\
 & x\,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) + x\,\mathbf{R}\circ_{\mathrm{iw}} (y\,\mathbf{L}\circ_{\mathrm{iw}} z + y\,\mathbf{R}\circ_{\mathrm{iw}} z) \\
= & \{\mathrm{S}\} \\
 & x\,\mathbf{L}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) + x\,\mathbf{R}\circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z) \\
= & \{\mathrm{S}\} \\
 & x \circ_{\mathrm{iw}} (y \circ_{\mathrm{iw}} z)
\end{array}
$$

⊠

Finally we give two more identities. They correspond to the axioms $\circ_{\mathrm{iw}}1$ and $\circ_{\mathrm{iw}}3$ from Table 2.

Proposition 4 For $E, F \subseteq EID$ and $a \in A$ such that $AE(a) \cap E \neq \emptyset$ we have

$$
\begin{aligned}
\delta_E \circ_{\mathrm{iw}} a &= \delta_{E\cup AE(a)}, \\
\delta_E \circ_{\mathrm{iw}} \delta_F &= \delta_{E\cup F}.
\end{aligned}
$$

Proof. For the first identity consider

$$\begin{aligned}\delta_E \circ_{\mathrm{iw}} a &= \delta_E \mathbf{L}\circ_{\mathrm{iw}} a + \delta_E \mathbf{R}\circ_{\mathrm{iw}} a = \delta_{E\cup AE(a)} + \delta_E \mathbf{R}\circ_{\mathrm{iw}} a \cdot \varepsilon \\ &= \delta_{E\cup AE(a)} + \delta_{E\cup AE(a)} = \delta_{E\cup AE(a)},\end{aligned}$$

and for the second consider

$$\delta_E \circ_{\mathrm{iw}} \delta_F = \delta_E \mathbf{L}\circ_{\mathrm{iw}} \delta_F + \delta_E \mathbf{R}\circ_{\mathrm{iw}} \delta_F = \delta_{E\cup F} + \delta_{E\cup F} = \delta_{E\cup F}.$$

⊠

Observe that we have now shown that all identities on closed $IWD(\circ)$-terms that are derivably equal in the process algebra $IWD(\circ)$, are also derivably equal in the process algebra $IWD(\circ, \cdot, +)$.

6 The E-interworking merge

Now that we have defined the process algebras $BPA(\delta_E)$ and $IWD(\circ, \cdot, +)$ which include operators for alternative and sequential composition, we aim at extending them with the merge operator. For technical reasons, we do this in two steps: First we will define the E-interworking merge in this section and in the next section we will extend the obtained process algebra to its final shape.

The E-interworking merge of x and y, denoted by $x \parallel^E_{\mathrm{iw}} y$, is the parallel execution of the processes x and y with the restriction that the processes must synchronise on all atomic actions which are defined on entities from the set E. This set E is static, which means that it remains unchanged during calculations on a term which contains the E-interworking merge operator. The resulting process algebra is called $IWD(\circ, +, \cdot, \parallel^E)$.

The deduction rules defining the operational semantics of the E-interworking merge are given in Table 7. The E-interworking merge of two processes can terminate if and only if both operands can terminate. The second and third rule in Table 7 say that if an operand can perform an action, the merge can perform the same action, provided that the action is not supposed to synchronise (i.e. the sender and receiver are not both in E). The fourth rule expresses the behaviour of a merge in case a synchronised action is possible.

Definition 10 (Active entities) For arbitrary $i, j \in EID$, $m \in MID$, $E \subseteq EID$, $x, y \in \mathcal{C}(\Sigma_{IWD(\circ, +, \cdot, \parallel^E)})$, and $\odot \in \{\circ_{\mathrm{iw}}, \mathbf{L}\circ_{\mathrm{iw}}, \mathbf{R}\circ_{\mathrm{iw}}, +, \cdot, \parallel^E_{\mathrm{iw}}, \between^E_{\mathrm{iw}}, |^E_{\mathrm{iw}} \mid E \subseteq EID\}$, we define the mapping $AE : \mathcal{C}(\Sigma_{IWD(\circ, +, \cdot, \parallel^E)}) \to I\!P(EID)$ inductively as follows:

$$\begin{aligned} AE(c(i,j,m)) &= \{i, j\}, \\ AE(\varepsilon) &= \emptyset, \\ AE(\delta_E) &= E, \\ AE(x \odot y) &= AE(x) \cup AE(y). \end{aligned}$$

$$\frac{x\downarrow \quad y\downarrow}{x \parallel^E_{\mathrm{iw}} y\downarrow} \qquad \frac{x \xrightarrow{a} x' \quad AE(a) \not\subseteq E}{x \parallel^E_{\mathrm{iw}} y \xrightarrow{a} x' \parallel^E_{\mathrm{iw}} y} \qquad \frac{y \xrightarrow{a} y' \quad AE(a) \not\subseteq E}{x \parallel^E_{\mathrm{iw}} y \xrightarrow{a} x \parallel^E_{\mathrm{iw}} y'}$$

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{a} y' \quad AE(a) \subseteq E}{x \parallel^E_{\mathrm{iw}} y \xrightarrow{a} x' \parallel^E_{\mathrm{iw}} y'}$$

Table 7: Deduction rules for E-interworking merge ($a \in A$, $E \subseteq EID$)

For the axiomatisation of the E-interworking merge we need two auxiliary operators, similar to the axiomatisation of the communication merge of ACP. These additional operators are $\lfloor\!\lfloor^E_{\mathrm{iw}}$ (E-interworking left-merge) and $|^E_{\mathrm{iw}}$ (E-interworking synchronisation merge). The process $x \lfloor\!\lfloor^E_{\mathrm{iw}} y$ behaves like the process $x \parallel^E_{\mathrm{iw}} y$ with the restriction that the first action must come from process x and that action cannot synchronise with an action from y. The process $x |^E_{\mathrm{iw}} y$ behaves as the process $x \parallel^E_{\mathrm{iw}} y$ with the restriction that the first action must be a synchronisation. This is formalised by the deduction rules in Table 8. The term deduction system $T(IWD(\circ, +, \cdot, \|^E))$ consists of the deduction rules of Tables 1, 3, 5, 7 and 8.

$$\frac{x\downarrow \quad y\downarrow}{x \lfloor\!\lfloor^E_{\mathrm{iw}} y\downarrow} \qquad \frac{x \xrightarrow{a} x' \quad AE(a) \not\subseteq E}{x \lfloor\!\lfloor^E_{\mathrm{iw}} y \xrightarrow{a} x' \parallel^E_{\mathrm{iw}} y}$$

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{a} y' \quad AE(a) \subseteq E}{x |^E_{\mathrm{iw}} y \xrightarrow{a} x' \parallel^E_{\mathrm{iw}} y'}$$

Table 8: Deduction rules for auxiliary operators of E-interworking merge ($a \in A$, $E \subseteq EID$)

Table 9 presents the axioms defining the E-interworking merge and its auxiliary operators. Axiom M states that either one of the two operands executes a non-synchronised action ($x \lfloor\!\lfloor^E_{\mathrm{iw}} y + y \lfloor\!\lfloor^E_{\mathrm{iw}} x$), or that a synchronised action takes place ($x |^E_{\mathrm{iw}} y$). The definition of the $\lfloor\!\lfloor^E_{\mathrm{iw}}$ operator (LM1-LM7) is very similar to the definition of the $\mathbf{L}\circ_{\mathrm{iw}}$ operator in Section 5 (Table 6, axioms L1-L7). The only difference is that axiom L6 is unconditional, whereas axiom LM6b has to take care of eliminating actions which are supposed to synchronise. Axioms CM1-CM7 define the $|^E_{\mathrm{iw}}$ operator. This operator enables all actions that can be performed by both operands and which must synchronise. In all other cases it yields a deadlock, where the scope of the deadlock can be derived from the operands.

It turns out that IWD-bisimilarity is a congruence for the operators in the signature of the process algebra $IWD(\circ, +, \cdot, \|^E)$. Furthermore, $IWD(\circ, +, \cdot, \|^E)$ is a sound and complete axiomatisation of IWD-bisimilarity on closed $IWD(\circ, +, \cdot, \|^E)$-terms. The proofs are based on the meta-theory

M	$x \,\|\|^E_{\mathrm{iw}}\, y = x \between^E_{\mathrm{iw}} y + y \between^E_{\mathrm{iw}} x + x \mid^E_{\mathrm{iw}} y$	
LM1	$\varepsilon \between^E_{\mathrm{iw}} \varepsilon = \varepsilon$	
LM2	$\varepsilon \between^E_{\mathrm{iw}} \delta_F = \delta_F$	
LM3	$\varepsilon \between^E_{\mathrm{iw}} a \cdot x = \delta_{AE(a \cdot x)}$	
LM4	$\varepsilon \between^E_{\mathrm{iw}} (x + y) = \varepsilon \between^E_{\mathrm{iw}} x + \varepsilon \between^E_{\mathrm{iw}} y$	
LM5	$\delta_F \between^E_{\mathrm{iw}} x = \delta_{F \cup AE(x)}$	
LM6a	$a \cdot x \between^E_{\mathrm{iw}} y = a \cdot (x \,\|\|^E_{\mathrm{iw}}\, y)$	if $AE(a) \not\subseteq E$
LM6b	$a \cdot x \between^E_{\mathrm{iw}} y = \delta_{AE(a \cdot x) \cup AE(y)}$	if $AE(a) \subseteq E$
LM7	$(x + y) \between^E_{\mathrm{iw}} z = x \between^E_{\mathrm{iw}} z + y \between^E_{\mathrm{iw}} z$	
CM1	$\varepsilon \mid^E_{\mathrm{iw}} x = \delta_{AE(x)}$	
CM2	$x \mid^E_{\mathrm{iw}} \varepsilon = \delta_{AE(x)}$	
CM3	$\delta_F \mid^E_{\mathrm{iw}} x = \delta_{F \cup AE(x)}$	
CM4	$x \mid^E_{\mathrm{iw}} \delta_F = \delta_{F \cup AE(x)}$	
CM5a	$a \cdot x \mid^E_{\mathrm{iw}} b \cdot y = a \cdot (x \,\|\|^E_{\mathrm{iw}}\, y)$	if $a \equiv b \wedge AE(a) \subseteq E$
CM5b	$a \cdot x \mid^E_{\mathrm{iw}} b \cdot y = \delta_{AE(a \cdot x) \cup AE(b \cdot y)}$	if $a \not\equiv b \vee AE(a) \not\subseteq E$
CM6	$(x + y) \mid^E_{\mathrm{iw}} z = x \mid^E_{\mathrm{iw}} z + y \mid^E_{\mathrm{iw}} z$	
CM7	$x \mid^E_{\mathrm{iw}} (y + z) = x \mid^E_{\mathrm{iw}} y + x \mid^E_{\mathrm{iw}} z$	

Table 9: Axioms of E-interworking merge ($a, b \in A$, $E, F \in \mathit{EID}$)

presented in [BV95, Ver95].

Theorem 15 (Congruence) IWD-bisimilarity is a congruence for E-interworking merge and the auxiliary operators.

Proof. The term deduction system is in path format and hence IWD-bisimilarity is a congruence for all operators. ⊠

Theorem 16 (Soundness) The process algebra $IWD(\circ, +, \cdot, \|^E)$ is a sound axiomatisation of IWD-bisimilarity on closed $IWD(\circ, +, \cdot, \|^E)$-terms.

Proof. For the axioms LM1-LM5, LM6b, CM1-CM4, and CM5b the IWD-bisimulations that witness the soundness are trivial. If "$s = t$ if b" represents such an axiom, then the IWD-bisimulation is $R = \{(s,t),(t,s) \mid b\}$.

For the axioms M, LM7, CM6, and CM7 the IWD-bisimulation is given by $R = \{(s,t),(t,s) \mid b\} \cup I$ if the axiom is given as "$s = t$ if b".

For the axioms LM6a and CM5a the IWD-bisimulations are

$$\begin{aligned} R \;=\; & \{(a \cdot x \between^E_{\mathrm{iw}} y, a \cdot (x \|^E_{\mathrm{iw}} y)), (\varepsilon \cdot x \|^E_{\mathrm{iw}} y, \varepsilon \cdot (x \|^E_{\mathrm{iw}} y)), (\varepsilon \cdot x \|^E_{\mathrm{iw}} q, x \|^E_{\mathrm{iw}} q) \\ & \mid AE(a) \not\subseteq E, q \in \mathcal{C}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})\}^S \cup I \end{aligned}$$

and

$$\begin{aligned} R \;=\; & \{(a \cdot x \mid^E_{\mathrm{iw}} b \cdot y, a \cdot (x \|^E_{\mathrm{iw}} y)), (\varepsilon \cdot x \|^E_{\mathrm{iw}} \varepsilon \cdot y, \varepsilon \cdot (x \|^E_{\mathrm{iw}} y)), \\ & (p \|^E_{\mathrm{iw}} \varepsilon \cdot y, p \|^E_{\mathrm{iw}} y), (\varepsilon \cdot x \|^E_{\mathrm{iw}} q, x \|^E_{\mathrm{iw}} q) \\ & \mid a \equiv b, AE(a) \subseteq E, p, q \in \mathcal{C}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})\}^S \cup I \end{aligned}$$

respectively. ⊠

Lemma 6 For arbitrary basic terms b_1 and b_2 we have the existence of basic terms c_1, c_2, and c_3 such that $b_1 \between^E_{\mathrm{iw}} b_2 = c_1$, $b_1 \mid^E_{\mathrm{iw}} b_2 = c_2$ and $b_1 \|^E_{\mathrm{iw}} b_2 = c_3$.

Proof. These statements are proven simultaneously with induction on the total number of symbols of the basic terms b_1 and b_2.

1. By case distinction on the structure of basic term b_1.
   (a) $b_1 \equiv \varepsilon$. By case distinction on the structure of basic term b_2.
      i. $b_2 \equiv \varepsilon$. Then $b_1 \between^E_{\mathrm{iw}} b_2 = \varepsilon \between^E_{\mathrm{iw}} \varepsilon = \varepsilon$.
      ii. $b_2 \equiv \delta_{F_2}$ for some $F_2 \subseteq EID$. Then $b_1 \between^E_{\mathrm{iw}} b_2 = \varepsilon \between^E_{\mathrm{iw}} \delta_{F_2} = \delta_{F_2}$.

iii. $b_2 \equiv a_2 \cdot b_2'$ for some $a_2 \in A$ and $b_2' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. Then

$$b_1 \between{E}{\mathrm{iw}} b_2 = \varepsilon \between{E}{\mathrm{iw}} a_2 \cdot b_2' = \delta_{AE(a_2 \cdot b_2')}.$$

iv. $b_2 \equiv b_2' + b_2''$ for some $b_2', b_2'' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. By induction we have the existence of basic terms c_1' and c_1'' such that $\varepsilon \between{E}{\mathrm{iw}} b_2' = c_1'$ and $\varepsilon \between{E}{\mathrm{iw}} b_2'' = c_1''$. Then

$$b_1 \between{E}{\mathrm{iw}} b_2 = \varepsilon \between{E}{\mathrm{iw}} (b_2' + b_2'') = \varepsilon \between{E}{\mathrm{iw}} b_2' + \varepsilon \between{E}{\mathrm{iw}} b_2'' = c_1' + c_1''.$$

(b) $b_1 \equiv \delta_{F_1}$ for some $F_1 \subseteq EID$. Then $b_1 \between{E}{\mathrm{iw}} b_2 = \delta_{F_1} \between{E}{\mathrm{iw}} b_2 = \delta_{F_1 \cup AE(b_2)}$.

(c) $b_1 \equiv a_1 \cdot b_1'$ for some $a_1 \in A$ and $b_1' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. By induction we have the existence of basic term c_1' such that $b_1' \|^E_{\mathrm{iw}} b_2 = c_1'$. Then, if $AE(a) \not\subseteq E$,

$$b_1 \between{E}{\mathrm{iw}} b_2 = a_1 \cdot b_1' \between{E}{\mathrm{iw}} b_2 = a_1 \cdot (b_1' \|^E_{\mathrm{iw}} b_2) = a_1 \cdot c_1'.$$

If $AE(a_1) \subseteq E$, then $b_1 \between{E}{\mathrm{iw}} b_2 = a_1 \cdot b_1' \between{E}{\mathrm{iw}} b_2 = \delta_{AE(a_1 \cdot b_1') \cup AE(b_2)}$.

(d) $b_1 \equiv b_1' + b_1''$ for some $b_1', b_1'' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. By induction we have the existence of basic terms c_1' and c_1'' such that $b_1' \between{E}{\mathrm{iw}} b_2 = c_1'$ and $b_1'' \between{E}{\mathrm{iw}} b_2 = c_1''$. Then

$$b_1 \between{E}{\mathrm{iw}} b_2 = (b_1' + b_1'') \between{E}{\mathrm{iw}} b_2 = b_1' \between{E}{\mathrm{iw}} b_2 + b_1'' \between{E}{\mathrm{iw}} b_2 = c_1' + c_1''.$$

2. By case distinction on the structure of basic term b_1.

(a) $b_1 \equiv \varepsilon$. Then $b_1 \mid^E_{\mathrm{iw}} b_2 = \varepsilon \mid^E_{\mathrm{iw}} b_2 = \delta_{AE(b_2)}$.

(b) $b_1 \equiv \delta_{F_1}$ for some $F_1 \subseteq EID$. Then $b_1 \mid^E_{\mathrm{iw}} b_2 = \delta_{F_1} \mid^E_{\mathrm{iw}} b_2 = \delta_{F_1 \cup AE(b_2)}$.

(c) $b_1 \equiv a_1 \cdot b_1'$ for some $a_1 \in A$ and $b_1' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. By case distinction on the structure of basic term b_2.

i. $b_2 \equiv \varepsilon$. Then $b_1 \mid^E_{\mathrm{iw}} b_2 = b_1 \mid^E_{\mathrm{iw}} \varepsilon = \delta_{AE(b_1)}$.

ii. $b_2 \equiv \delta_{F_2}$ for some $F_2 \subseteq EID$. Then $b_1 \mid^E_{\mathrm{iw}} b_2 = b_1 \mid^E_{\mathrm{iw}} \delta_{F_2} = \delta_{AE(b_1) \cup F_2}$.

iii. $b_2 \equiv a_2 \cdot b_2'$ for some $a_2 \in A$ and $b_2' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. By induction we have the existence of basic term c_3 such that $b_1' \|^E_{\mathrm{iw}} b_2' = c_3$. Then, if $a_1 \equiv a_2 \wedge AE(a_1) \subseteq E$,

$$b_1 \mid^E_{\mathrm{iw}} b_2 = a_1 \cdot b_1' \mid^E_{\mathrm{iw}} a_2 \cdot b_2' = a_1 \cdot (b_1' \|^E_{\mathrm{iw}} b_2') = a_1 \cdot c_3.$$

If $a_1 \not\equiv a_2 \vee AE(a_1) \not\subseteq E$, then

$$b_1 \mid^E_{\mathrm{iw}} b_2 = a_1 \cdot b_1' \mid^E_{\mathrm{iw}} a_2 \cdot b_2' = \delta_{AE(a_1 \cdot b_1') \cup AE(a_1 \cdot b_2')}.$$

iv. $b_2 \equiv b_2' + b_2''$ for some $b_2', b_2'' \in \mathcal{B}(\Sigma_{IWD(\circ,+,\cdot,\|^E)})$. By induction we have the existence of basic terms c_2' and c_2'' such that $b_1 \mid^E_{\mathrm{iw}} b_2' = c_2'$ and $b_1 \mid^E_{\mathrm{iw}} b_2'' = c_2''$. Then

$$b_1 \mid^E_{\mathrm{iw}} b_2 = b_1 \mid^E_{\mathrm{iw}} (b_2' + b_2'') = b_1 \mid^E_{\mathrm{iw}} b_2' + b_1 \mid^E_{\mathrm{iw}} b_2'' = c_2' + c_2''.$$

(d) $b_1 \equiv b_1' + b_1''$ for some $b_1', b_1'' \in \mathcal{B}(\Sigma_{IWD(\circ, +, \cdot, \|^E)})$. By induction we have the existence of basic terms c_2' and c_2'' such that $b_1' \mid^E_{\mathrm{iw}} b_2 = c_2'$ and $b_1'' \mid^E_{\mathrm{iw}} b_2 = c_2''$. Then

$$b_1 \mid^E_{\mathrm{iw}} b_2 = (b_1' + b_1'') \mid^E_{\mathrm{iw}} b_2 = b_1' \mid^E_{\mathrm{iw}} b_2 + b_1'' \mid^E_{\mathrm{iw}} b_2 = c_2' + c_2''.$$

3. By the previous two items we have the existence of basic terms c_1', c_1'', and c_2 such that $b_1 \between^E_{\mathrm{iw}} b_2 = c_1'$, $b_2 \between^E_{\mathrm{iw}} b_1 = c_1''$, and $b_1 \mid^E_{\mathrm{iw}} b_2 = c_2$. Then,

$$b_1 \|^E_{\mathrm{iw}} b_2 = b_1 \between^E_{\mathrm{iw}} b_2 + b_2 \between^E_{\mathrm{iw}} b_1 + b_1 \mid^E_{\mathrm{iw}} b_2 = c_1' + c_1'' + c_2.$$

⊠

Theorem 17 (Elimination) For every closed $IWD(\circ, +, \cdot, \|^E)$-term x there is a derivably equal basic term s.

Proof. This theorem is proven by induction on the structure of closed $IWD(\circ, +, \cdot, \|^E)$-term x. All cases except for $x \equiv x' \between^E_{\mathrm{iw}} x''$, $x \equiv x' \mid^E_{\mathrm{iw}} x''$, and $x \equiv x' \|^E_{\mathrm{iw}} x''$ have already been treated in the proof of Theorem 12. In the remaining three cases we find the existence of basic terms b_1 and b_2 such that $x_1 = b_1$ and $x_2 = b_2$. Using the previous lemma we find the desired result. ⊠

Theorem 18 (Conservativity) The process algebra $IWD(\circ, +, \cdot, \|^E)$ is a conservative extension of the process algebra $IWD(\circ, \cdot, +)$.

Proof. The proof of this theorem uses the approach of [Ver95]. The conservativity follows from the following observations:

1. IWD-bisimilarity is definable in terms of predicate and relation symbols only,
2. $IWD(\circ, \cdot, +)$ is a complete axiomatisation of IWD-bisimilarity on closed $IWD(\circ, \cdot, +)$-terms,
3. $IWD(\circ, +, \cdot, \|^E)$ is a sound axiomatisation of IWD-bisimilarity on closed $IWD(\circ, +, \cdot, \|^E)$-terms (see Theorem 16),
4. the term deduction system for $IWD(\circ, \cdot, +)$ is pure, well-founded and in path format, and
5. the term deduction system for $IWD(\circ, +, \cdot, \|^E)$ is in path format.

⊠

Theorem 19 (Completeness) The process algebra $IWD(\circ, +, \cdot, \|^E)$ is a complete axiomatisation of IWD-bisimilarity on closed $IWD(\circ, +, \cdot, \|^E)$-terms.

Proof. By the General Completeness Theorem of [Ver94], the completeness of the process algebra $IWD(\circ, +, \cdot, \|^E)$ follows immediately from the properties mentioned in the proof of Theorem 18 and the fact that $IWD(\circ, +, \cdot, \|^E)$ has the elimination property for $BPA(\delta_E)$(see Theorem 8) and hence also for $IWD(\circ, \cdot, +)$. ⊠

When defining an operator for parallel composition, several properties are desirable, such as commutativity, the existence of a unit element, and associativity (under some condition). The proof of associativity in the process algebra is quite complicated.

Proposition 5 (Commutativity $\|^E_{\mathbf{iw}}$) For closed $IWD(\circ, +, \cdot, \|^E)$-terms x, y, and a set of entities E we have

$$x \,|^E_{\mathrm{iw}}\, y = y \,|^E_{\mathrm{iw}}\, x,$$
$$x \,\|^E_{\mathrm{iw}}\, y = y \,\|^E_{\mathrm{iw}}\, x.$$

Proof. The propositions are proven simultaneously with induction on the general structure of basic terms x and y. Let

$$x = \sum_{i\in I} a_i \cdot x_i + \delta_E + \sum_{k\in K} \varepsilon, \qquad y = \sum_{l\in L} b_l \cdot y_l + \delta_F + \sum_{n\in N} \varepsilon,$$

for some finite index sets I, K, L, N, $a_i, b_l \in A$, $E, F \subseteq EID$ and basic terms x_i, y_l. Then,

$$\begin{aligned}
x \,|^E_{\mathrm{iw}}\, y =& \sum_{i\in I}\sum_{l\in L} a_i \cdot x_i \,|^E_{\mathrm{iw}}\, b_l \cdot y_l + x \,|^E_{\mathrm{iw}}\, \delta_F + \sum_{n\in N} x \,|^E_{\mathrm{iw}}\, \varepsilon + \delta_E \,|^E_{\mathrm{iw}}\, y + \sum_{k\in K} \varepsilon \,|^E_{\mathrm{iw}}\, y \\
=& \sum_{i\in I} \sum_{l\in L, a_i\equiv b_l, AE(a_i)\subseteq E} a_i \cdot (x_i \,\|^E_{\mathrm{iw}}\, y_l) \\
&+ \sum_{i\in I} \sum_{l\in L, a_i\not\equiv b_l \vee AE(a_i)\not\subseteq E} \delta_{AE(a_i\cdot x_i)\cup AE(b_l\cdot y_l)} \\
&+ \delta_{AE(x)\cup F} + \sum_{n\in N} \delta_{AE(x)} + \delta_{E\cup AE(y)} + \sum_{k\in K} \delta_{AE(y)} \\
=& \sum_{l\in L} \sum_{i\in I, b_l\equiv a_i, AE(b_l)\subseteq E} b_l \cdot (y_l \,\|^E_{\mathrm{iw}}\, x_i) \\
&+ \sum_{l\in L} \sum_{i\in I, b_l\not\equiv a_i \vee AE(b_l)\not\subseteq E} \delta_{AE(b_l\cdot y_l)\cup AE(a_i\cdot x_i)} \\
&+ \delta_F \,|^E_{\mathrm{iw}}\, x + \sum_{n\in N} \varepsilon \,|^E_{\mathrm{iw}}\, x + y \,|^E_{\mathrm{iw}}\, \delta_E + \sum_{k\in K} y \,|^E_{\mathrm{iw}}\, \varepsilon \\
=& \sum_{l\in L}\sum_{i\in I} b_l \cdot y_l \,|^E_{\mathrm{iw}}\, a_i \cdot x_i + \delta_F \,|^E_{\mathrm{iw}}\, x + \sum_{n\in N} \varepsilon \,|^E_{\mathrm{iw}}\, x + y \,|^E_{\mathrm{iw}}\, \delta_E + \sum_{k\in K} y \,|^E_{\mathrm{iw}}\, \varepsilon \\
=& y \,|^E_{\mathrm{iw}}\, x
\end{aligned}$$

and

$$x \,\|^E_{\mathrm{iw}}\, y = x \between^E_{\mathrm{iw}} y + y \between^E_{\mathrm{iw}} x + x \,|^E_{\mathrm{iw}}\, y = y \between^E_{\mathrm{iw}} x + x \between^E_{\mathrm{iw}} y + y \,|^E_{\mathrm{iw}}\, x = y \,\|^E_{\mathrm{iw}}\, x.$$

$\boxtimes$

Proposition 6 For closed $IWD(\circ, +, \cdot, \|^E)$-terms x we have

$$x \,\|^{\emptyset}_{\mathrm{iw}}\, \varepsilon = x,$$
$$\varepsilon \,\|^{\emptyset}_{\mathrm{iw}}\, x = x.$$

Proof. The first proposition is proven with induction on the general structure of basic term x. Let

$$x = \sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon,$$

for some finite index sets I, K, $a_i \in A$, $E \subseteq EID$ and basic terms x_i. The induction hypothesis is $x_i \,\|^{\emptyset}_{\mathrm{iw}}\, \varepsilon = x_i$ for all $i \in I$. Then,

$$\begin{aligned} x \between^{\emptyset}_{\mathrm{iw}} \varepsilon &= \sum_{i \in I} a_i \cdot x_i \between^{\emptyset}_{\mathrm{iw}} \varepsilon + \delta_E \between^{\emptyset}_{\mathrm{iw}} \varepsilon + \sum_{k \in K} \varepsilon \between^{\emptyset}_{\mathrm{iw}} \varepsilon \\ &= \sum_{i \in I} a_i \cdot (x_i \,\|^{\emptyset}_{\mathrm{iw}}\, \varepsilon) + \delta_E + \sum_{k \in K} \varepsilon \\ &= \sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon \\ &= x \end{aligned}$$

and

$$\begin{aligned} \varepsilon \between^{\emptyset}_{\mathrm{iw}} x &= \varepsilon \between^{\emptyset}_{\mathrm{iw}} (\sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon) \\ &= \sum_{i \in I} \varepsilon \between^{\emptyset}_{\mathrm{iw}} a_i \cdot x_i + \varepsilon \between^{\emptyset}_{\mathrm{iw}} \delta_E + \sum_{k \in K} \varepsilon \between^{\emptyset}_{\mathrm{iw}} \varepsilon \\ &= \sum_{i \in I} \delta_{AE(a_i \cdot x_i)} + \delta_E + \sum_{k \in K} \varepsilon. \end{aligned}$$

Using these two subcomputations we obtain:

$$\begin{aligned} x \,\|^{\emptyset}_{\mathrm{iw}}\, \varepsilon &= x \between^{\emptyset}_{\mathrm{iw}} \varepsilon + \varepsilon \between^{\emptyset}_{\mathrm{iw}} x + x \,|^{\emptyset}_{\mathrm{iw}}\, \varepsilon \\ &= x + \sum_{i \in I} \delta_{AE(a_i \cdot x_i)} + \delta_E + \sum_{k \in K} \varepsilon + \delta_{AE(x)} \\ &= x. \end{aligned}$$

The other part of the proposition is obtained using the commutativity of $\|^{\emptyset}_{\mathrm{iw}}$. $\boxtimes$

The following proposition serves our needs in proving interworking merge associative in the next section.

Proposition 7 (Associativity of $\|^E_{\mathbf{iw}}$) For closed $IWD(\circ, +, \cdot, \|^E)$-terms x, y, and z, and sets of entities E_1, E_2, and E_3 such that $AE(x) \subseteq E_1$, $AE(y) \subseteq E_2$, and $AE(z) \subseteq E_3$, we have

$$(x \|^{E_1 \cap E_2}_{\mathrm{iw}} y) \|^{(E_1 \cup E_2) \cap E_3}_{\mathrm{iw}} z = x \|^{E_1 \cap (E_2 \cup E_3)}_{\mathrm{iw}} (y \|^{E_2 \cap E_3}_{\mathrm{iw}} z).$$

Proof. Without loss of generality we can assume that x, y, and z are basic terms. We use the following shorthands: $S = E_1 \cap E_2 \cap E_3$, $E_{12} = (E_1 \cap E_2) \setminus S$, $E_{23} = (E_2 \cap E_3) \setminus S$, and $E_{13} = (E_1 \cap E_3) \setminus S$. In Figure 16 these sets are indicated in a Venn diagram.

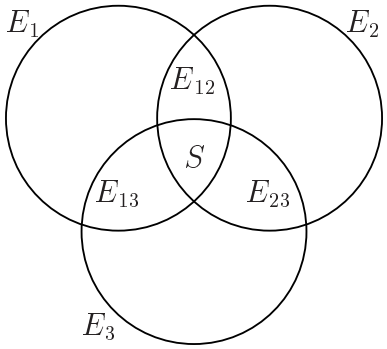

Figure 16: Explanation of shorthands

We prove the following propositions simultaneously with induction on the structure of the basic terms x, y, and z.

$$(x \between^{E_{12} \cup S}_{\mathrm{iw}} y) \between^{E_{13} \cup E_{23} \cup S}_{\mathrm{iw}} z = x \between^{E_{12} \cup E_{13} \cup S}_{\mathrm{iw}} (y \|^{E_{23} \cup S}_{\mathrm{iw}} z) \tag{5}$$

$$(x |^{E_{12} \cup S}_{\mathrm{iw}} y) |^{E_{13} \cup E_{23} \cup S}_{\mathrm{iw}} z = x |^{E_{12} \cup E_{13} \cup S}_{\mathrm{iw}} (y |^{E_{23} \cup S}_{\mathrm{iw}} z) \tag{6}$$

$$(x |^{E_{12} \cup S}_{\mathrm{iw}} y) \between^{E_{13} \cup E_{23} \cup S}_{\mathrm{iw}} z = x |^{E_{12} \cup E_{13} \cup S}_{\mathrm{iw}} (y \between^{E_{23} \cup S}_{\mathrm{iw}} z) \tag{7}$$

$$(x \|^{E_{12} \cup S}_{\mathrm{iw}} y) \|^{E_{13} \cup E_{23} \cup S}_{\mathrm{iw}} z = x \|^{E_{12} \cup E_{13} \cup S}_{\mathrm{iw}} (y \|^{E_{23} \cup S}_{\mathrm{iw}} z) \tag{8}$$

Let

$$x = \sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon,$$

$$y = \sum_{l \in L} b_l \cdot y_l + \delta_F + \sum_{n \in N} \varepsilon,$$

$$z = \sum_{o \in O} c_o \cdot z_o + \delta_G + \sum_{q \in Q} \varepsilon,$$

for some finite index sets I, K, L, N, O, Q, $a_i, b_l, c_o \in A$, $E, F, G \subseteq EID$ and basic terms x_i, y_l, z_o.

We only give the proofs for (6) and (8). The proof for (6) uses induction on the general form of basic terms x, y, and z.

From $AE(x) \subseteq E_1$, $AE(y) \subseteq E_2$, and $AE(z) \subseteq E_3$ we obtain $AE(x_i) \subseteq E_1$, $AE(y_l) \subseteq E_2$, and $AE(z_o) \subseteq E_3$ for all $i \in I$, $l \in L$, and $o \in O$. This means that we are allowed to use

$$(x_i \parallel_{\mathrm{iw}}^{E_{12} \cup S} y_l) \parallel_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} z_o = x_i \parallel_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} (y_l \parallel_{\mathrm{iw}}^{E_{23} \cup S} z_o).$$

First, we give a number of subcomputations. These are used in proving equation (6).

Subcomputation 1:

$$\begin{aligned}
&(a_i \cdot x_i \mid_{\mathrm{iw}}^{E_{12} \cup S} b_l \cdot y_l) \mid_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} c_o \cdot z_o \\
=&\begin{cases} a_i \cdot (x_i \parallel_{\mathrm{iw}}^{E_{12} \cup S} y_l) \mid_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} c_o \cdot z_o & \text{if } a_i \equiv b_l \wedge AE(a_i) \subseteq E_{12} \cup S \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l)} \mid_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} c_o \cdot z_o & \text{otherwise} \end{cases} \\
=&\begin{cases} a_i \cdot ((x_i \parallel_{\mathrm{iw}}^{E_{12} \cup S} y_l) \parallel_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} z_o) & \text{if } a_i \equiv b_l \wedge AE(a_i) \subseteq E_{12} \cup S \\ & \wedge\ a_i \equiv c_o \wedge AE(a_i) \subseteq E_{13} \cup E_{23} \cup S \\ \delta_{AE(a_i) \cup AE(x_i) \cup AE(y_l) \cup AE(c_o \cdot z_o)} & \text{if } a_i \equiv b_l \wedge AE(a_i) \subseteq E_{12} \cup S \\ & \wedge\ (a_i \not\equiv c_o \vee AE(a_i) \not\subseteq E_{13} \cup E_{23} \cup S) \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l) \cup AE(c_o \cdot z_o)} & \text{otherwise} \end{cases} \\
=&\begin{cases} a_i \cdot ((x_i \parallel_{\mathrm{iw}}^{E_{12} \cup S} y_l) \parallel_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} z_o) & \text{if } a_i \equiv b_l \wedge AE(a_i) \subseteq E_{12} \cup S \\ & \wedge\ a_i \equiv c_o \wedge AE(a_i) \subseteq E_{13} \cup E_{23} \cup S \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l) \cup AE(c_o \cdot z_o)} & \text{otherwise} \end{cases} \\
=&\begin{cases} a_i \cdot ((x_i \parallel_{\mathrm{iw}}^{E_{12} \cup S} y_l) \parallel_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} z_o) & \text{if } a_i \equiv b_l \equiv c_o \wedge AE(a_i) \subseteq S \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l) \cup AE(c_o \cdot z_o)} & \text{otherwise} \end{cases} \\
=&\begin{cases} a_i \cdot (x_i \parallel_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} (y_l \parallel_{\mathrm{iw}}^{E_{23} \cup S} z_o)) & \text{if } a_i \equiv b_l \equiv c_o \wedge AE(a_i) \subseteq S \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l) \cup AE(c_o \cdot z_o)} & \text{otherwise} \end{cases} \\
=&\begin{cases} a_i \cdot (x_i \parallel_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} (y_l \parallel_{\mathrm{iw}}^{E_{23} \cup S} z_o)) & \text{if } a_i \equiv b_l \wedge AE(a_i) \subseteq E_{12} \cup E_{13} \cup S \\ & \wedge\ b_l \equiv c_o \wedge AE(b_l) \subseteq E_{23} \cup S \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l) \cup AE(y_l) \cup AE(z_o)} & \text{if } (a_i \not\equiv b_l \vee AE(a_i) \not\subseteq E_{12} \cup E_{13} \cup S) \\ & \wedge\ b_l \equiv c_o \wedge AE(b_l) \subseteq E_{23} \cup S \\ \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l) \cup AE(c_o \cdot z_o)} & \text{otherwise} \end{cases} \\
=&\begin{cases} a_i \cdot x_i \mid_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} b_l \cdot (y_l \parallel_{\mathrm{iw}}^{E_{23} \cup S} z_o) & \text{if } b_l \equiv c_o \wedge AE(b_l) \subseteq E_{23} \cup S \\ a_i \cdot x_i \mid_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} \delta_{AE(b_l \cdot y_l) \cup AE(c_o \cdot z_o)} & \text{otherwise} \end{cases} \\
=&a_i \cdot x_i \mid_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} (b_l \cdot y_l \mid_{\mathrm{iw}}^{E_{23} \cup S} c_o \cdot z_o)
\end{aligned}$$

Subcomputation 2:

$$\begin{aligned}
(\delta_E \mid_{\mathrm{iw}}^{E_{12} \cup S} y) \mid_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} z &= \delta_{E \cup AE(y)} \mid_{\mathrm{iw}}^{E_{13} \cup E_{23} \cup S} z \\
&= \delta_{E \cup AE(y) \cup AE(z)} \\
&= \delta_E \mid_{\mathrm{iw}}^{E_{12} \cup E_{13} \cup S} (y \mid_{\mathrm{iw}}^{E_{23} \cup S} z)
\end{aligned}$$

Similarly we obtain

$$(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, \delta_F) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z = x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (\delta_F \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z)$$

and

$$(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, \delta_G = x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, \delta_G).$$

Subcomputation 3:

$$\begin{aligned}(\varepsilon \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z &= \delta_{AE(y)} \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z \\ &= \delta_{AE(y)\cup AE(z)} \\ &= \varepsilon \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z)\end{aligned}$$

Similarly we obtain

$$(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, \varepsilon) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z = x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (\varepsilon \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z)$$

and

$$(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, \varepsilon = x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, \varepsilon).$$

Then, using these subcomputations, we obtain

$$\begin{aligned}&(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z \\ =&\sum_{i\in I}\sum_{l\in L}\sum_{o\in O}(a_i\cdot x_i \,|_{\mathrm{iw}}^{E_{12}\cup S}\, b_l\cdot y_l) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, c_o\cdot z_o \\ &+ (\delta_E \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z + (x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, \delta_F) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z \\ &+ (x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, \delta_G + \sum_{k\in K}(\varepsilon \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z \\ &+ \sum_{n\in N}(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, \varepsilon) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, z + \sum_{q\in Q}(x \,|_{\mathrm{iw}}^{E_{12}\cup S}\, y) \,|_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S}\, \varepsilon \\ =&\sum_{i\in I}\sum_{l\in L}\sum_{o\in O} a_i\cdot x_i \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (b_l\cdot y_l \,|_{\mathrm{iw}}^{E_{23}\cup S}\, c_o\cdot z_o) \\ &+ \delta_E \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z) + x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (\delta_F \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z) \\ &+ x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, \delta_G) + \sum_{k\in K}\varepsilon \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z) \\ &+ \sum_{n\in N} x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (\varepsilon \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z) + \sum_{q\in Q} x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, \varepsilon) \\ =&x \,|_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S}\, (y \,|_{\mathrm{iw}}^{E_{23}\cup S}\, z).\end{aligned}$$

Finally, equation (8) is proven as follows:

$$
\begin{aligned}
&(x \parallel_{\mathrm{iw}}^{E_{12}\cup S} y) \parallel_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z\\
=&(x \parallel_{\mathrm{iw}}^{E_{12}\cup S} y) \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z + z \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} (x \parallel_{\mathrm{iw}}^{E_{12}\cup S} y) + (x \parallel_{\mathrm{iw}}^{E_{12}\cup S} y) \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z\\
=&(x \between_{\mathrm{iw}}^{E_{12}\cup S} y + y \between_{\mathrm{iw}}^{E_{12}\cup S} x + x \mid_{\mathrm{iw}}^{E_{12}\cup S} y) \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z + z \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} (x \parallel_{\mathrm{iw}}^{E_{12}\cup S} y)\\
&+ (x \between_{\mathrm{iw}}^{E_{12}\cup S} y + y \between_{\mathrm{iw}}^{E_{12}\cup S} x + x \mid_{\mathrm{iw}}^{E_{12}\cup S} y) \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z\\
=&(x \between_{\mathrm{iw}}^{E_{12}\cup S} y) \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z + (y \between_{\mathrm{iw}}^{E_{12}\cup S} x) \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z + (x \mid_{\mathrm{iw}}^{E_{12}\cup S} y) \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z\\
&+ z \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} (x \parallel_{\mathrm{iw}}^{E_{12}\cup S} y) + (x \between_{\mathrm{iw}}^{E_{12}\cup S} y) \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z + (y \between_{\mathrm{iw}}^{E_{12}\cup S} x) \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z\\
&+ (x \mid_{\mathrm{iw}}^{E_{12}\cup S} y) \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} z\\
=&x \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) + y \between_{\mathrm{iw}}^{E_{12}\cup E_{23}\cup S} (x \parallel_{\mathrm{iw}}^{E_{13}\cup S} z) + x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \between_{\mathrm{iw}}^{E_{23}\cup S} z)\\
&+ z \between_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} (y \parallel_{\mathrm{iw}}^{E_{12}\cup S} x) + z \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} (x \between_{\mathrm{iw}}^{E_{12}\cup S} y) + z \mid_{\mathrm{iw}}^{E_{13}\cup E_{23}\cup S} (y \between_{\mathrm{iw}}^{E_{12}\cup S} x)\\
&+ x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \mid_{\mathrm{iw}}^{E_{23}\cup S} z)\\
=&x \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) + y \between_{\mathrm{iw}}^{E_{12}\cup E_{23}\cup S} (z \parallel_{\mathrm{iw}}^{E_{13}\cup S} x) + x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \between_{\mathrm{iw}}^{E_{23}\cup S} z)\\
&+ (z \between_{\mathrm{iw}}^{E_{23}\cup S} y) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x + (z \mid_{\mathrm{iw}}^{E_{13}\cup S} x) \between_{\mathrm{iw}}^{E_{12}\cup E_{23}\cup S} y + (z \mid_{\mathrm{iw}}^{E_{23}\cup S} y) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x\\
&+ x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \mid_{\mathrm{iw}}^{E_{23}\cup S} z)\\
=&x \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) + (y \between_{\mathrm{iw}}^{E_{23}\cup S} z) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x + x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \between_{\mathrm{iw}}^{E_{23}\cup S} z)\\
&+ (z \between_{\mathrm{iw}}^{E_{23}\cup S} y) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x + (x \mid_{\mathrm{iw}}^{E_{13}\cup S} z) \between_{\mathrm{iw}}^{E_{12}\cup E_{23}\cup S} y + (y \mid_{\mathrm{iw}}^{E_{23}\cup S} z) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x\\
&+ x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \mid_{\mathrm{iw}}^{E_{23}\cup S} z)\\
=&x \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) + (y \between_{\mathrm{iw}}^{E_{23}\cup S} z + z \between_{\mathrm{iw}}^{E_{23}\cup S} y + y \mid_{\mathrm{iw}}^{E_{23}\cup S} z) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x\\
&+ x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \between_{\mathrm{iw}}^{E_{23}\cup S} z) + x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (z \between_{\mathrm{iw}}^{E_{23}\cup S} y) + x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \mid_{\mathrm{iw}}^{E_{23}\cup S} z)\\
=&x \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) + (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x\\
&+ x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \between_{\mathrm{iw}}^{E_{23}\cup S} z + z \between_{\mathrm{iw}}^{E_{23}\cup S} y + y \mid_{\mathrm{iw}}^{E_{23}\cup S} z)\\
=&x \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) + (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z) \between_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} x + x \mid_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z)\\
=&x \parallel_{\mathrm{iw}}^{E_{12}\cup E_{13}\cup S} (y \parallel_{\mathrm{iw}}^{E_{23}\cup S} z).
\end{aligned}
$$

⊠

By taking $E_1 = E_2 = E_3$ we obtain associativity of $\parallel_{\mathrm{iw}}^{E}$.

The final property which we prove is the correspondence between the $\circ_{\mathrm{iw}}$ and $\parallel_{\mathrm{iw}}^{E}$ operators. This formalises the resemblance between the axiomatic definitions of these operators.

Proposition 8 For closed $IWD(\circ, +, \cdot, \parallel^{E})$-terms x and y such that $AE(x) \cap AE(y) = \emptyset$ we have

$$x \parallel_{\mathrm{iw}}^{\emptyset} y = x \circ_{\mathrm{iw}} y.$$

Proof. Let

$$x = \sum_{i \in I} a_i \cdot x_i + \delta_E + \sum_{k \in K} \varepsilon,$$

$$y = \sum_{l \in L} b_l \cdot y_l + \delta_F + \sum_{n \in N} \varepsilon$$

for some finite index sets I, K, L, N, $a_i, b_l \in A$, $E, F \subseteq EID$ and basic terms x_i, y_l. The induction hypotheses are $x_i \parallel^{\emptyset}_{\mathrm{iw}} y = x_i \circ_{\mathrm{iw}} y$ for all $i \in I$ and $x \parallel^{\emptyset}_{\mathrm{iw}} y_l = x \circ_{\mathrm{iw}} y_l$ for all $l \in L$. Then

$$\begin{aligned}
x \between^{\emptyset}_{\mathrm{iw}} y =& \sum_{i \in I} a_i \cdot x_i \between^{\emptyset}_{\mathrm{iw}} y + \delta_E \between^{\emptyset}_{\mathrm{iw}} y + \sum_{k \in K} \varepsilon \between^{\emptyset}_{\mathrm{iw}} y \\
=& \sum_{i \in I} a_i \cdot (x_i \parallel^{\emptyset}_{\mathrm{iw}} y) + \delta_{E \cup AE(y)} + \sum_{k \in K} \varepsilon \between^{\emptyset}_{\mathrm{iw}} y \\
=& \sum_{i \in I} a_i \cdot (x_i \circ_{\mathrm{iw}} y) + \delta_{E \cup AE(y)} + \sum_{k \in K} \sum_{n \in N} \varepsilon + \delta_{AE(y)} \\
=& \sum_{i \in I} a_i \cdot (x_i \circ_{\mathrm{iw}} y) + \delta_E \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y + \sum_{k \in K} \varepsilon \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y \\
=& \sum_{i \in I} a_i \cdot x_i \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y + \delta_E \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y + \sum_{k \in K} \varepsilon \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y \\
=& x \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y.
\end{aligned}$$

In the following computations we use that $AE(b_l) \subseteq AE(y)$ and $AE(x) \cap AE(y) = \emptyset$ imply $AE(b_l) \cap AE(x) = \emptyset$:

$$\begin{aligned}
y \between^{\emptyset}_{\mathrm{iw}} x =& \sum_{l \in L} b_l \cdot y_l \between^{\emptyset}_{\mathrm{iw}} x + \delta_F \between^{\emptyset}_{\mathrm{iw}} x + \sum_{n \in N} \varepsilon \between^{\emptyset}_{\mathrm{iw}} x \\
=& \sum_{l \in L} b_l \cdot (y_l \parallel^{\emptyset}_{\mathrm{iw}} x) + \delta_{F \cup AE(x)} + \sum_{n \in N} \sum_{k \in K} \varepsilon + \delta_{AE(x)} \\
=& \sum_{l \in L} b_l \cdot (x \parallel^{\emptyset}_{\mathrm{iw}} y_l) + \delta_{F \cup AE(x)} + \sum_{n \in N} \sum_{k \in K} \varepsilon + \delta_{AE(x)} \\
=& \sum_{l \in L} b_l \cdot (x \circ_{\mathrm{iw}} y_l) + x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} \delta_F + \sum_{n \in N} x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} \varepsilon \\
=& \sum_{l \in L} x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} b_l \cdot y_l + x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} \delta_F + \sum_{n \in N} x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} \varepsilon \\
=& x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} y,
\end{aligned}$$

$$\begin{aligned}
x \mid^{\emptyset}_{\mathrm{iw}} y =& \sum_{i \in I} \sum_{l \in L} a_i \cdot x_i \mid^{\emptyset}_{\mathrm{iw}} b_l \cdot y_l + \delta_E \mid^{\emptyset}_{\mathrm{iw}} y + x \mid^{\emptyset}_{\mathrm{iw}} \delta_F + \sum_{k \in K} \varepsilon \mid^{\emptyset}_{\mathrm{iw}} y + \sum_{n \in N} x \mid^{\emptyset}_{\mathrm{iw}} \varepsilon \\
=& \sum_{i \in I} \sum_{l \in L} \delta_{AE(a_i \cdot x_i) \cup AE(b_l \cdot y_l)} + \delta_{E \cup AE(y)} + \delta_{F \cup AE(x)} + \delta_{AE(y)} + \delta_{AE(x)} \\
=& \delta_{AE(x) \cup AE(y)},
\end{aligned}$$

and therefore

$$\begin{aligned}
x \parallel^{\emptyset}_{\mathrm{iw}} y =& x \between^{\emptyset}_{\mathrm{iw}} y + y \between^{\emptyset}_{\mathrm{iw}} x + x \mid^{\emptyset}_{\mathrm{iw}} y \\
=& x \mathbin{\mathbf{L}\circ_{\mathrm{iw}}} y + x \mathbin{\mathbf{R}\circ_{\mathrm{iw}}} x + \delta_{AE(x) \cup AE(y)} \\
=& x \circ_{\mathrm{iw}} y + \delta_{AE(x) \cup AE(y)}
\end{aligned}$$

$$= x \circ_{\mathrm{iw}} y.$$

$\boxtimes$

7 Process Algebra for Interworkings

In the previous section we introduced the E-interworking merge. This operator was parameterised with the set of entities on which the processes should synchronise. In order for the interworking merge to be generally applicable, the set E must be determined from the actual operands of the interworking merge. Therefore, we have to generalise the E-interworking merge to the interworking merge operator.

There is a technical complication which makes this generalisation non-trivial: we have to explicitly attribute every process term with the set of entities that it contains. The reason for this is revealed by the examples in Figures 12 and 13 (see Section 2.4).

Using the definitions from the previous sections, interworking $X2$ from Figure 12 has the following semantical representation: $c(c, d, m)$. There is no explicit mention of the empty entity b. Indeed, this interpretation is exactly the same as the interpretation of interworking $X2'$ from Figure 13.

In a context with only $+$ and $\circ_{\mathrm{iw}}$ operators, this identification would be completely harmless, however Figures 12 and 13 show that there is a merge context which makes a distinction between $X2$ and $X2'$.

The reason for this anomaly is that we did not take empty entities into consideration. Therefore, in order to properly define the interworking merge, we have to extend our semantical representation with information about the entities contained.

There are several ways to achieve this. A first option would be to attribute the empty process ε with a set of entities. Empty entity b would then be represented by $\varepsilon_{\{b\}}$. A second option would be to label a complete process term with the set of entities which it ranges over. The semantical representation of $X2$ would then become $\langle c(c, d, m), \{b, c, d\}\rangle$, whereas $X2'$ would be represented by $\langle c(c, d, m), \{c, d\}\rangle$.

For technical reasons we choose to elaborate on the second option. An Interworking with a dynamical behaviour denoted by x over the entities from E is denoted by $\langle x, E\rangle$. Such a tuple $\langle x, E\rangle$ will be called an entity-labeled process.

Definition 11 (Signature) The signature of the process algebra $IWE(\circ, +, \|)$ consists of the operators $\langle \_, \_ \rangle$, $+$, $\circ_{\mathrm{iw}}$, and $\|_{\mathrm{iw}}$.

For $\langle x, E\rangle$ to be a well-formed expression we do not require that the active entities from x are all contained in E. All entities in E which are not active entities in x are empty entities. The

active entities of $\langle x, E\rangle$ can be determined from x solely. The complete set of entities of $\langle x, E\rangle$, denoted by $Ent(\langle x, E\rangle)$, contains the active entities from x and the entities from E.

Definition 12 (Active entities, entities) For closed $IWD(\circ, +, \cdot, \|^E)$-term x, $E \subseteq EID$, and closed $IWE(\circ, +, \|)$-terms s and t we define the mappings $AE : \mathcal{C}(\Sigma_{IWE(\circ, +, \|)}) \to I\!P(EID)$ and $Ent : \mathcal{C}(\Sigma_{IWE(\circ, +, \|)}) \to I\!P(EID)$ inductively as follows:

$$\begin{array}{lcl}
AE(\langle x, E\rangle) & = & AE(x), \\
AE(s + t) & = & AE(s) \cup AE(t), \\
AE(s \circ_{\mathrm{iw}} t) & = & AE(s) \cup AE(t), \\
AE(s \|_{\mathrm{iw}} t) & = & AE(s) \cup AE(t),
\end{array}$$

$$\begin{array}{lcl}
Ent(\langle x, E\rangle) & = & E \cup AE(x), \\
Ent(s + t) & = & Ent(s) \cup Ent(t), \\
Ent(s \circ_{\mathrm{iw}} t) & = & Ent(s) \cup Ent(t), \\
Ent(s \|_{\mathrm{iw}} t) & = & Ent(s) \cup Ent(t).
\end{array}$$

On entity-labeled processes we define the operators interworking sequencing and interworking merge. The set of all entity-labeled processes is called $IWE(\circ, +, \|)$. The definition of the interworking sequencing on entity-labeled processes is straightforward.

Before we give axioms for the process algebra $IWE(\circ, +, \|)$, we define a operational semantics. The operational semantics of entity-labeled processes, as expressed in Table 10, is similar to the operational semantics of non-labeled processes.

The first two rules relate the domains of non-labeled processes and entity labeled processes. In the second rule we have to take care that we do not loose information about the involved entities after executing an action. It may happen that some active entity from x which does not occur in E is not active anymore in x' since the last action from that entity has been executed. Therefore, we have to extend the entity label of x' with the active entities of x. The rules for the interworking merge correspond to the rules for the E-interworking merge but the condition $AE(a) \subseteq E$ is replaced by $AE(a) \subseteq Ent(s) \cap Ent(t)$. The set $Ent(s) \cap Ent(t)$ contains the shared entities from s and t, so this is the set of entities which should synchronise.

$$\frac{x\downarrow}{\langle x,E\rangle\downarrow} \qquad \frac{x\xrightarrow{a}x'}{\langle x,E\rangle\xrightarrow{a}\langle x',E\cup AE(x)\rangle}$$

$$\frac{s\downarrow}{s+t\downarrow} \qquad \frac{t\downarrow}{s+t\downarrow} \qquad \frac{s\xrightarrow{a}s'}{s+t\xrightarrow{a}s'\circ_{\mathrm{iw}}\langle\varepsilon,Ent(t)\rangle} \qquad \frac{t\xrightarrow{a}t'}{s+t\xrightarrow{a}t'\circ_{\mathrm{iw}}\langle\varepsilon,Ent(s)\rangle}$$

$$\frac{s\downarrow \quad t\downarrow}{s\circ_{\mathrm{iw}}t\downarrow} \qquad \frac{s\xrightarrow{a}s'}{s\circ_{\mathrm{iw}}t\xrightarrow{a}s'\circ_{\mathrm{iw}}t} \qquad \frac{AE(a)\cap AE(s)=\emptyset \quad t\xrightarrow{a}t'}{s\circ_{\mathrm{iw}}t\xrightarrow{a}s\circ_{\mathrm{iw}}t'}$$

$$\frac{s\xrightarrow{a}s' \quad AE(a)\not\subseteq Ent(s)\cap Ent(t)}{s\,\|_{\mathrm{iw}}\,t\xrightarrow{a}s'\,\|_{\mathrm{iw}}\,t} \qquad \frac{t\xrightarrow{a}t' \quad AE(a)\not\subseteq Ent(s)\cap Ent(t)}{s\,\|_{\mathrm{iw}}\,t\xrightarrow{a}s\,\|_{\mathrm{iw}}\,t'}$$

$$\frac{s\downarrow \quad t\downarrow}{s\,\|_{\mathrm{iw}}\,t\downarrow} \qquad \frac{s\xrightarrow{a}s' \quad t\xrightarrow{a}t' \quad AE(a)\subseteq Ent(s)\cap Ent(t)}{s\,\|_{\mathrm{iw}}\,t\xrightarrow{a}s'\,\|_{\mathrm{iw}}\,t'}$$

Table 10: Operational semantics of entity-labeled processes ($a \in A$, $E \subseteq EID$, x, x' $IWD(\circ, +, \cdot, \|^E)$-terms, s, s', t, t' entity-labeled processes)

For the "correctness" of the deduction rules for interworking merge it is necessary that the set of entities of a process does not change by executing actions (Lemma 8). This is guaranteed by the deduction rules. We first prove that the set of active entities does not expand due to the execution of actions.

Lemma 7 For all $a \in A$ and closed $IWD(\circ, +, \cdot, \|^E)$-terms x and x' we have: if $x \xrightarrow{a} x'$, then $AE(x) \supseteq AE(x')$.

Proof. This lemma is proven with induction on the structure of closed $IWD(\circ, +, \cdot, \|^E)$-term x. Suppose that $x \xrightarrow{a} x'$.

1. $x \equiv \varepsilon$. This case cannot occur as $\varepsilon \not\xrightarrow{a}$.
2. $x \equiv \delta_E$ for some $E \subseteq EID$. This case cannot occur as $\delta_E \not\xrightarrow{a}$.
3. $x \equiv b$ for some $b \in A$. Then it must be the case that $b \equiv a$ and $x' \equiv \varepsilon$. Clearly $AE(x) = AE(b) \supseteq \emptyset = AE(\varepsilon) = AE(x')$.
4. $x \equiv x_1 + x_2$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2. Then it must be the case that either $x_1 \xrightarrow{a} x'$ or $x_2 \xrightarrow{a} x'$. By induction we thus have either $AE(x_1) \supseteq AE(x')$ or $AE(x_2) \supseteq AE(x')$. In either case we have $AE(x) = AE(x_1 + x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x')$.

5. $x \equiv x_1 \cdot x_2$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2. Then we can distinguish two cases. First, $x_1 \xrightarrow{a} x_1'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x_1' such that $x' \equiv x_1' \cdot x_2$. By induction we have $AE(x_1) \supseteq AE(x_1')$. Then $AE(x) = AE(x_1 \cdot x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1') \cup AE(x_2) = AE(x_1' \cdot x_2) = AE(x')$. Second, $x_1 \downarrow$ and $x_2 \xrightarrow{a} x'$. By induction we have $AE(x_2) \supseteq AE(x')$. Then $AE(x) = AE(x_1 \cdot x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1) \cup AE(x') \supseteq AE(x')$.

6. $x \equiv x_1 \circ_{\mathrm{iw}} x_2$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2. Then we can distinguish two cases. First, $x_1 \xrightarrow{a} x_1'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x_1' such that $x' \equiv x_1' \circ_{\mathrm{iw}} x_2$. By induction we have $AE(x_1) \supseteq AE(x_1')$. Then $AE(x) = AE(x_1 \circ_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1') \cup AE(x_2) = AE(x_1' \circ_{\mathrm{iw}} x_2) = AE(x')$. Second, $AE(a) \cap AE(x_1) = \emptyset$ and $x_2 \xrightarrow{a} x_2'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x_2' such that $x' \equiv x_1 \circ_{\mathrm{iw}} x_2'$. By induction we have $AE(x_2) \supseteq AE(x_2')$. Then $AE(x) = AE(x_1 \circ_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1) \cup AE(x_2') = AE(x_1 \circ_{\mathrm{iw}} x_2') = AE(x')$.

7. $x \equiv x_1 \|^E_{\mathrm{iw}} x_2$ for some $E \subseteq EID$ and closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2. Then we can distinguish three cases. First, $AE(a) \not\subseteq E$ and $x_1 \xrightarrow{a} x_1'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x_1' such that $x' \equiv x_1' \|^E_{\mathrm{iw}} x_2$. By induction we have $AE(x_1) \supseteq AE(x_1')$. Then $AE(x) = AE(x_1 \|^E_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1') \cup AE(x_2) = AE(x_1' \|^E_{\mathrm{iw}} x_2) = AE(x')$. Second, $AE(a) \not\subseteq E$ and $x_2 \xrightarrow{a} x_2'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x_2' such that $x' \equiv x_1 \|^E_{\mathrm{iw}} x_2'$. By induction we have $AE(x_2) \supseteq AE(x_2')$. Then $AE(x) = AE(x_1 \|^E_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1) \cup AE(x_2') = AE(x_1 \|^E_{\mathrm{iw}} x_2') = AE(x')$. Third, $AE(a) \subseteq E$, $x_1 \xrightarrow{a} x_1'$, and $x_2 \xrightarrow{a} x_2'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1' and x_2' such that $x' \equiv x_1' \|^E_{\mathrm{iw}} x_2'$. By induction we have $AE(x_1) \supseteq AE(x_1')$ and $AE(x_2) \supseteq AE(x_2')$. Then $AE(x) = AE(x_1 \|^E_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \supseteq AE(x_1') \cup AE(x_2') = AE(x_1' \|^E_{\mathrm{iw}} x_2') = AE(x')$.

⊠

Lemma 8 For all closed $IWE(\circ, +, \|)$-terms s and t and all $a \in A$ we have: if $s \xrightarrow{a} s'$, then $Ent(s) = Ent(s')$.

Proof. This lemma is proven with induction on the structure of closed $IWE(\circ, +, \|)$-term s.

1. $s \equiv \langle x, E \rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x and $E \subseteq EID$. Then $s \xrightarrow{a} s'$ must be due to $x \xrightarrow{a} x'$ for some x' such that $s' \equiv \langle x', E \cup AE(x) \rangle$. Clearly we have $Ent(s) = Ent(\langle x, E \rangle) = E \cup AE(x)$ and $Ent(s') = Ent(\langle x', E \cup AE(x) \rangle) = E \cup AE(x) \cup AE(x')$. Using Lemma 7 we obtain $Ent(s) = Ent(s')$.

2. $s \equiv s_1 + s_2$ for some closed $IWE(\circ, +, \|)$-terms s_1 and s_2. We can distinguish two cases. First, $s_1 \xrightarrow{a} s_1'$ for some closed $IWE(\circ, +, \|)$-term s_1' such that $s' \equiv s_1' \circ_{\mathrm{iw}} \langle \varepsilon, Ent(s_2) \rangle$. By induction we have $Ent(s_1) = Ent(s_1')$. Therefore, $Ent(s) = Ent(s_1 + s_2) = Ent(s_1) \cup Ent(s_2) = Ent(s_1') \cup Ent(s_2) = Ent(s_1') \cup Ent(\langle \varepsilon, Ent(s_2) \rangle) = Ent(s_1' \circ_{\mathrm{iw}} \langle \varepsilon, Ent(s_2) \rangle) = Ent(s')$. Second, $s_2 \xrightarrow{a} s_2'$ for some closed $IWE(\circ, +, \|)$-term s_2' such that $s' \equiv s_2' \circ_{\mathrm{iw}} \langle \varepsilon, Ent(s_1) \rangle$. This case is symmetrical to the first case.

3. $s \equiv s_1 \circ_{\mathrm{iw}} s_2$ for some closed $IWE(\circ, +, \|)$-terms s_1 and s_2. We can distinguish two cases. First, $s_1 \xrightarrow{a} s_1'$ for some closed $IWE(\circ, +, \|)$-term s_1' such that $s' \equiv s_1' \circ_{\mathrm{iw}} s_2$. By induction we have $Ent(s_1) = Ent(s_1')$. Therefore, $Ent(s) = Ent(s_1 \circ_{\mathrm{iw}} s_2) = Ent(s_1) \cup Ent(s_2) = Ent(s_1') \cup Ent(s_2) = Ent(s_1' \circ_{\mathrm{iw}} s_2) = Ent(s')$. Second, $AE(a) \cap AE(s_2) = \emptyset$ and $s_2 \xrightarrow{a} s_2'$ for some closed $IWE(\circ, +, \|)$-term s_2' such that $s' \equiv s_1 \circ_{\mathrm{iw}} s_2'$. By induction we have $Ent(s_2) = Ent(s_2')$. Therefore, $Ent(s) = Ent(s_1 \circ_{\mathrm{iw}} s_2) = Ent(s_1) \cup Ent(s_2) = Ent(s_1) \cup Ent(s_2') = Ent(s_1 \circ_{\mathrm{iw}} s_2') = Ent(s')$.

4. $s \equiv s_1 \|_{\mathrm{iw}} s_2$ for some closed $IWE(\circ, +, \|)$-terms s_1 and s_2. We can distinguish three cases. First, $AE(a) \not\subseteq Ent(s_1) \cap Ent(s_2)$ and $s_1 \xrightarrow{a} s_1'$ for some closed $IWE(\circ, +, \|)$-term s_1' such that $s' \equiv s_1' \|_{\mathrm{iw}} s_2$. By induction we have $Ent(s_1) = Ent(s_1')$. Therefore, $Ent(s) = Ent(s_1 \|_{\mathrm{iw}} s_2) = Ent(s_1) \cup Ent(s_2) = Ent(s_1') \cup Ent(s_2) = Ent(s_1' \|_{\mathrm{iw}} s_2) = Ent(s')$. Second, $AE(a) \not\subseteq Ent(s_1) \cap Ent(s_2)$ and $s_2 \xrightarrow{a} s_2'$ for some closed $IWE(\circ, +, \|)$-term s_2' such that $s' \equiv s_1 \|_{\mathrm{iw}} s_2'$. By induction we have $Ent(s_2) = Ent(s_2')$. Therefore, $Ent(s) = Ent(s_1 \|_{\mathrm{iw}} s_2) = Ent(s_1) \cup Ent(s_2) = Ent(s_1) \cup Ent(s_2') = Ent(s_1 \|_{\mathrm{iw}} s_2') = Ent(s')$. Third, $AE(a) \subseteq Ent(s_1) \cap Ent(s_2)$, $s_1 \xrightarrow{a} s_1'$, and $s_2 \xrightarrow{a} s_2'$ for some closed $IWE(\circ, +, \|)$-terms s_1' and s_2' such that $s' \equiv s_1' \|_{\mathrm{iw}} s_2'$. By induction we have $Ent(s_1) = Ent(s_1')$ and $Ent(s_2') = Ent(s_2)$. Therefore, $Ent(s) = Ent(s_1 \|_{\mathrm{iw}} s_2) = Ent(s_1) \cup Ent(s_2) = Ent(s_1') \cup Ent(s_2') = Ent(s_1' \|_{\mathrm{iw}} s_2') = Ent(s')$.

⊠

Next, we adapt the definition of IWD-bisimilarity to take into account the set of entities of a process.

Definition 13 (Entity bisimilarity) A symmetric relation R on closed $IWE(\circ, +, \|)$-terms is an *entity bisimulation*, if and only if, for every pair $(s, t) \in R$ and $a \in A$, the following conditions hold:

1. $AE(s) = AE(t)$,
2. if $s \downarrow$, then $t \downarrow$,
3. if $s \xrightarrow{a} s'$, then there is a closed $IWE(\circ, +, \|)$-term t' such that $t \xrightarrow{a} t'$ and $(s', t') \in R$,
4. $Ent(s) = Ent(t)$.

The closed $IWE(\circ, +, \|)$-terms s and t are *entity bisimilar*, notation $s \underline{\leftrightarrow} t$, if and only if there exists an entity bisimulation R relating them.

Theorem 20 (Equivalence) Entity bisimilarity is an equivalence relation.

Proof. The proof is similar to the proof that IWD-bisimilarity is an equivalence (Theorem 1) and therefore omitted. ⊠

Theorem 21 (Congruence) Entity bisimilarity is a congruence for the function symbols in the signature of $IWE(\circ, +, \|)$ which are defined on $IWE(\circ, +, \|)$-terms.

Proof. Suppose $R : x \underline{\leftrightarrow}_{\text{iwd}} y$ and $E_x = E_y$. Now we must prove that there exists an entity bisimulation R' such that $R' : \langle x, E_x\rangle \underline{\leftrightarrow} \langle y, E_y\rangle$. Let $R' = \{(\langle p, E\rangle, \langle q, F\rangle) \mid pRq, E = F\}$. Let p and q be closed $IWD(\circ, +, \cdot, \|^E)$-terms such that pRq and $E, F \subseteq EID$ such that $E = F$. Since $p \underline{\leftrightarrow}_{\text{iwd}} q$ we have $AE(p) = AE(q)$.

1. $AE(\langle p, E\rangle) = AE(p) = AE(q) = AE(\langle q, F\rangle)$.
2. Suppose that $\langle p, E\rangle \xrightarrow{a} s$ for some closed $IWE(\circ, +, \|)$-term s. This must be due to $p \xrightarrow{a} p'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term p' such that $s \equiv \langle p', E \cup AE(p)\rangle$. As $p \underline{\leftrightarrow}_{\text{iwd}} q$, we have the existence of closed $IWD(\circ, +, \cdot, \|^E)$-term q' such that $q \xrightarrow{a} q'$ and $p'Rq'$. Then we also obtain $\langle q, F\rangle \xrightarrow{a} \langle q', F \cup AE(q)\rangle$. Clearly $\langle p', E \cup AE(p)\rangle R' \langle q', F \cup AE(q)\rangle$.
3. Suppose that $\langle p, E\rangle \downarrow$. This must be due to $p \downarrow$. As $p \underline{\leftrightarrow}_{\text{iwd}} q$, we have $q \downarrow$. Therefore, $\langle q, F\rangle \downarrow$.
4. $Ent(\langle p, E\rangle) = E \cup AE(p) = F \cup AE(q) = AE(\langle q, F\rangle)$.

Suppose $R_1 : s_1 \underline{\leftrightarrow} t_1$ and $R_2 : s_2 \underline{\leftrightarrow} t_2$. Let $R = \{(s_1 + t_1, s_2 + t_2), (p_1 \circ_{\text{iw}} \langle \varepsilon, E\rangle, q_1 \circ_{\text{iw}} \langle \varepsilon, E\rangle), (p_2 \circ_{\text{iw}} \langle \varepsilon, E\rangle, q_2 \circ_{\text{iw}} \langle \varepsilon, E\rangle) \mid p_1 R_1 q_1, p_2 R_2 q_2, E \subseteq EID\}$. Obviously, this relation is an entity bisimulation.

Suppose $R_1 : s_1 \underline{\leftrightarrow} t_1$ and $R_2 : s_2 \underline{\leftrightarrow} t_2$. Let $R = \{(p_1 \circ_{\text{iw}} p_2, q_1 \circ_{\text{iw}} q_2) \mid p_1 R_1 q_1, p_2 R_2 q_2\}$. Obviously this relation R is an entity bisimulation. The proof is similar to the proof that IWD-bisimilarity is a congruence for interworking sequencing (see Theorem 2).

Suppose $R_1 : s_1 \underline{\leftrightarrow} t_1$ and $R_2 : s_2 \underline{\leftrightarrow} t_2$. Let $R = \{(p_1 \|_{\text{iw}} p_2, q_1 \|_{\text{iw}} q_2) \mid p_1 R_1 q_1, p_2 R_2 q_2\}$. Obviously this relation R is an entity bisimulation. ⊠

As was done in [MvWW93], the interworking merge is expressed in terms of the E-interworking merge operator and the common entities of the operands. The axioms for entity-labeled processes are given in Table 11 for $E, F \subseteq EID$. The extension of $IWD(\circ, \cdot, +)$ with entity-labeled processes is denoted by $IWE(\circ, +, \|)$.

Axiom $IWE1$ describes the convention discussed before that the entity-part of an $IWE(\circ, +, \|)$-term contains at least the empty entities of the Interworking. Axioms $IWE2$-$IWE4$ describe how the other operators on $IWE(\circ, +, \|)$-terms can be defined in terms of their counterparts on $IWD(\circ, +, \cdot, \|^E)$-terms. It is also possible to define entity bisimulation in terms of IWD-bisimilarity of the process-parts and set equality of the entity-parts. Also for our final process algebra, $IWE(\circ, +, \|)$, we prove soundness and completeness.

Theorem 22 (Soundness) The process algebra $IWE(\circ, +, \|)$ is a sound axiomatisation of IWD-bisimilarity on closed $IWD(\circ, +, \cdot, \|^E)$-terms. The process algebra $IWE(\circ, +, \|)$ is a sound axiomatisation of entity bisimulation on closed $IWE(\circ, +, \|)$-terms.

$IWE1$	$\langle x, E\rangle = \langle x, E \cup AE(x)\rangle$	
$IWE2$	$\langle x, E\rangle + \langle y, F\rangle = \langle x + y, E \cup F\rangle$	
$IWE3$	$\langle x, E\rangle \circ_{\mathrm{iw}} \langle y, F\rangle = \langle x \circ_{\mathrm{iw}} y, E \cup F\rangle$	
$IWE4$	$\langle x, E\rangle \parallel_{\mathrm{iw}} \langle y, F\rangle = \langle x \parallel_{\mathrm{iw}}^{E\cap F} y, E \cup F\rangle$	if $AE(x) \subseteq E$ and $AE(y) \subseteq F$

Table 11: Axioms of entity-labeled processes ($E, F \subseteq EID$)

Proof. For the proof of the first proposition observe that we did not add any axioms relating closed $IWD(\circ, +, \cdot, \|^E)$-terms. We will prove the second proposition. Since entity bisimulation is a congruence for the closed $IWE(\circ, +, \|)$-terms (Theorem 21) we only have to show that the axioms from Table 11 are sound. Thereto, we provide an entity bisimulation relation for each axiom. For $IWE1$, the relation $R = \{(\langle x, E\rangle, \langle x, E \cup AE(x)\rangle)\}^S \cup I$ is an entity bisimulation. For the axiom $IWE2$ the relation $R = \{(\langle p, E\rangle + \langle q, F\rangle, \langle p + q, E \cup F\rangle), (\langle p, E\rangle \circ_{\mathrm{iw}} \langle \varepsilon, F\rangle, \langle p, E \cup F\rangle) \mid p, q \in \mathcal{C}(\Sigma_{IWD(\circ, +, \cdot, \|^E)}), E, F \subseteq EID\}^S$ is an entity bisimulation. For axiom $IWE3$ the relation $R = \{(\langle p, E'\rangle \circ_{\mathrm{iw}} \langle y, F\rangle, \langle p \circ_{\mathrm{iw}} y, E' \cup F \cup AE(y)\rangle), (\langle x, E\rangle \circ_{\mathrm{iw}} \langle q, F'\rangle, \langle x \circ_{\mathrm{iw}} q, E' \cup F' \cup AE(x)\rangle) \mid p, q \in \mathcal{C}(\Sigma_{IWD(\circ, +, \cdot, \|^E)}), E', F' \subseteq EID\}^S$ is an entity bisimulation. For $IWE4$, the relation $R = \{(\langle p, E\rangle \parallel_{\mathrm{iw}} \langle q, F\rangle, \langle p \parallel_{\mathrm{iw}}^{E\cap F} q, E \cup F\rangle \mid p, q \in \mathcal{C}(\Sigma_{IWD(\circ, +, \cdot, \|^E)}), E, F \subseteq EID, AE(p) \subseteq E, AE(q) \subseteq F\}^S$ is an entity bisimulation. ⊠

Definition 14 (Basic terms) The set of basic terms is the smallest set that satisfies: if x is a closed $IWD(\circ, +, \cdot, \|^E)$-term and $E \subseteq EID$ such that $AE(x) \subseteq E$, then $\langle x, E\rangle$ is a basic $IWE(\circ, +, \|)$-term. The set of all basic terms over the signature of $IWE(\circ, +, \|)$ is denoted by $\mathcal{B}(\Sigma_{IWE(\circ, +, \|)})$.

Theorem 23 (Elimination) For every closed $IWE(\circ, +, \|)$-term s we have the existence of a basic $IWE(\circ, +, \|)$-term t such that $IWE(\circ, +, \|) \vdash s = t$.

Proof. This theorem is proven with induction on the structure of a closed $IWE(\circ, +, \|)$-term. First, consider the case $s \equiv \langle x, E\rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x and $E \subseteq EID$. Then $s = \langle x, E\rangle = \langle x, E \cup AE(x)\rangle$. Clearly $AE(x) \subseteq E \cup AE(x)$ and hence $\langle x, E \cup AE(x)\rangle$ is a basic $IWE(\circ, +, \|)$-term. Then, consider the case $s \equiv s_1 + s_2$ for some closed $IWE(\circ, +, \|)$-terms s_1 and s_2. By induction we have the existence of basic terms $\langle x_1, E_1\rangle$ and $\langle x_2, E_2\rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2 and $E_1, E_2 \subseteq EID$ such that $AE(x_1) \subseteq E_1$ and $AE(x_2) \subseteq E_2$. Then, $s = s_1 + s_2 = \langle x_1, E_1\rangle + \langle x_2, E_2\rangle = \langle x_1 + x_2, E_1 \cup E_2\rangle$. Clearly $AE(x_1 + x_2) \subseteq E_1 \cup E_2$. Next, consider the case $s \equiv s_1 \circ_{\mathrm{iw}} s_2$ for some closed $IWE(\circ, +, \|)$-terms s_1 and s_2. By induction we have the existence of basic terms $\langle x_1, E_1\rangle$ and $\langle x_2, E_2\rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2 and $E_1, E_2 \subseteq EID$ such that $AE(x_1) \subseteq E_1$ and $AE(x_2) \subseteq E_2$. Then $s = s_1 \circ_{\mathrm{iw}} s_2 = \langle x_1, E_1\rangle \circ_{\mathrm{iw}} \langle x_2, E_2\rangle = \langle x_1 \circ_{\mathrm{iw}} x_2, E_1 \cup E_2\rangle$. Clearly

$AE(x_1 \circ_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \subseteq E_1 \cup E_2$. Finally, consider the case $s \equiv s_1 \parallel_{\mathrm{iw}} s_2$ for some s_1, s_2 closed $IWE(\circ, +, \|)$-terms. By induction we have the existence of basic terms $\langle x_1, E_1\rangle$ and $\langle x_2, E_2\rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1 and x_2 and $E_1, E_2 \subseteq EID$ such that $AE(x_1) \subseteq E_1$ and $AE(x_2) \subseteq E_2$. Then $s = s_1 \parallel_{\mathrm{iw}} s_2 = \langle x_1, E_1\rangle \parallel_{\mathrm{iw}} \langle x_2, E_2\rangle = \langle x_1 \parallel_{\mathrm{iw}}^{E_1 \cap E_2} x_2, E_1 \cup E_2\rangle$. Clearly $AE(x_1 \parallel_{\mathrm{iw}} x_2) = AE(x_1) \cup AE(x_2) \subseteq E_1 \cup E_2$. ⊠

Lemma 9 For basic $IWE(\circ, +, \|)$-terms $\langle x, E\rangle$ and $\langle y, F\rangle$ we have

$$\langle x, E\rangle \mathrel{\underline{\leftrightarrow}} \langle y, F\rangle \quad \text{iff} \quad x \mathrel{\underline{\leftrightarrow}}_{\mathrm{iwd}} y \quad \text{and} \quad E = F.$$

Proof. First, suppose that $R : \langle x, E\rangle \mathrel{\underline{\leftrightarrow}} \langle y, F\rangle$. Let $R' = \{(p, q) \mid \langle p, E'\rangle R \langle q, F'\rangle, E' = F'\}$. As $\langle x, E\rangle R \langle y, F\rangle$, $AE(x) \subseteq E$, and $AE(y) \subseteq F$, we have $E = E \cup AE(x) = Ent(x) = Ent(y) = F \cup AE(y) = F$. We will prove that R' is an IWD-bisimulation.

1. $AE(p) = AE(\langle p, E'\rangle) = AE(\langle q, F'\rangle) = AE(q)$.
2. $p\!\downarrow$ iff $\langle p, E'\rangle\!\downarrow$ iff $\langle q, F'\rangle\!\downarrow$ iff $q\!\downarrow$.
3. Suppose that $p \xrightarrow{a} p'$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term p'. Then $\langle p, E'\rangle \xrightarrow{a} \langle p', E' \cup AE(p)\rangle$. So we have $\langle q, F'\rangle \xrightarrow{a} \langle q', F' \cup AE(q')\rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term q' such that $\langle p', E' \cup AE(p)\rangle R \langle q', F' \cup AE(q')\rangle$. From this we obtain that $E' \cup AE(p') = F' \cup AE(q')$. Thus $p' R' q'$.

The proof in the other direction is trivial. ⊠

Theorem 24 (Completeness) The process algebra $IWE(\circ, +, \|)$ is a complete axiomatisation of entity bisimulation on closed $IWE(\circ, +, \|)$-terms.

Proof. By the elimination theorem (Theorem 23) we only have to prove this theorem for basic $IWE(\circ, +, \|)$-terms. Let $\langle x, E_1\rangle$ and $\langle y, E_2\rangle$ be arbitrary basic $IWE(\circ, +, \|)$-terms such that $\langle x, E_1\rangle \mathrel{\underline{\leftrightarrow}} \langle y, E_2\rangle$. By Lemma 9 we have $x \mathrel{\underline{\leftrightarrow}}_{\mathrm{iwd}} y$ and $E_1 = E_2$. Since $IWD(\circ, +, \cdot, \|^E)$ is a complete axiomatisation of $IWD(\circ)$-bisimilarity on closed $IWD(\circ, +, \cdot, \|^E)$-terms, we have $x = y$, and hence $\langle x, E_1\rangle = \langle y, E_2\rangle$. ⊠

Theorem 25 (Conservativity) The process algebra $IWE(\circ, +, \|)$ is a conservative extension of the process algebra $IWD(\circ, +, \cdot, \|^E)$.

Proof. With respect to $IWD(\circ, +, \cdot, \|^E)$-terms, the process algebra $IWE(\circ, +, \|)$ and the process algebra $IWD(\circ, +, \cdot, \|^E)$have exactly the same axioms. Then clearly the same equalities can be derived between closed $IWD(\circ, +, \cdot, \|^E)$-terms. ⊠

We end our treatment of the semantics of Interworkings with some properties of Interworkings. The interworking sequencing is commutative under the assumption that the active entities of the operands are disjoint. Furthermore, it is associative. The interworking merge is both commutative and associative.

Proposition 9 (Unit elements) For closed $IWE(\circ, +, \|)$ terms s,

$$s \circ_{\mathrm{iw}} \langle \varepsilon, \emptyset \rangle = s, \tag{9}$$
$$\langle \varepsilon, \emptyset \rangle \circ_{\mathrm{iw}} s = s, \tag{10}$$
$$s \|_{\mathrm{iw}} \langle \varepsilon, \emptyset \rangle = s, \tag{11}$$
$$\langle \varepsilon, \emptyset \rangle \|_{\mathrm{iw}} s = s. \tag{12}$$

Proof. By the elimination theorem it is allowed to restrict the proof of the statements to basic terms. Let $s = \langle x, E \rangle$ for some closed $IWD(\circ, +, \cdot, \|^E)$-term x and $E \subseteq EID$ such that $AE(x) \subseteq E$. Then

$$\begin{aligned}
s \circ_{\mathrm{iw}} \langle \varepsilon, \emptyset \rangle &= \langle x, E \rangle \circ_{\mathrm{iw}} \langle \varepsilon, \emptyset \rangle = \langle x \circ_{\mathrm{iw}} \varepsilon, E \cup \emptyset \rangle = \langle x, E \rangle = s, \\
\langle \varepsilon, \emptyset \rangle \circ_{\mathrm{iw}} s &= \langle \varepsilon, \emptyset \rangle \circ_{\mathrm{iw}} \langle x, E \rangle = \langle \varepsilon \circ_{\mathrm{iw}} x, \emptyset \cup E \rangle = \langle x, E \rangle = s, \\
s \|_{\mathrm{iw}} \langle \varepsilon, \emptyset \rangle &= \langle x, E \rangle \|_{\mathrm{iw}} \langle \varepsilon, \emptyset \rangle = \langle x \|_{\mathrm{iw}}^{E \cap \emptyset} \varepsilon, E \cup \emptyset \rangle = \langle x \|_{\mathrm{iw}}^{\emptyset} \varepsilon, E \rangle = \langle x, E \rangle = s, \\
\langle \varepsilon, \emptyset \rangle \|_{\mathrm{iw}} s &= \langle \varepsilon, \emptyset \rangle \|_{\mathrm{iw}} \langle x, E \rangle = \langle \varepsilon \|_{\mathrm{iw}}^{\emptyset \cap E} x, \emptyset \cup E \rangle = \langle \varepsilon \|_{\mathrm{iw}}^{\emptyset} x, E \rangle = \langle x, E \rangle = s.
\end{aligned}$$

⊠

Proposition 10 (Commutativity and associativity of $\circ_{\mathbf{iw}}$ and $\|_{\mathbf{iw}}$) For arbitrary closed $IWE(\circ, +, \|)$-terms s, t, u we have

$$s \circ_{\mathrm{iw}} t = t \circ_{\mathrm{iw}} s, \qquad \text{if } AE(s) \cap AE(t) = \emptyset \tag{13}$$
$$(s \circ_{\mathrm{iw}} t) \circ_{\mathrm{iw}} u = s \circ_{\mathrm{iw}} (t \circ_{\mathrm{iw}} u), \tag{14}$$
$$s \|_{\mathrm{iw}} t = t \|_{\mathrm{iw}} s, \tag{15}$$
$$(s \|_{\mathrm{iw}} t) \|_{\mathrm{iw}} u = s \|_{\mathrm{iw}} (t \|_{\mathrm{iw}} u). \tag{16}$$

Proof. By the elimination theorem it is allowed to restrict the proof of the statements to basic terms. Let $s = \langle x_1, E_1 \rangle$, $t = \langle x_2, E_2 \rangle$, and $u = \langle x_3, E_3 \rangle$ for some $E_1, E_2, E_3 \subseteq EID$ and closed $IWD(\circ, +, \cdot, \|^E)$-terms x_1, x_2, and x_3 such that $AE(x_1) \subseteq E_1$, $AE(x_2) \subseteq E_2$, and $AE(x_3) \subseteq E_3$. Then

$$\begin{aligned}
s \circ_{\mathrm{iw}} t &= \langle x_1, E_1 \rangle \circ_{\mathrm{iw}} \langle x_2, E_2 \rangle \\
&= \langle x_1 \circ_{\mathrm{iw}} x_2, E_1 \cup E_2 \rangle \\
&= \langle x_2 \circ_{\mathrm{iw}} x_1, E_2 \cup E_1 \rangle
\end{aligned}$$

$$\begin{aligned}
&= \langle x_2, E_2\rangle \circ_{\mathrm{iw}} \langle x_1, E_1\rangle \\
&= t \circ_{\mathrm{iw}} s,
\end{aligned}$$

$$\begin{aligned}
(s \circ_{\mathrm{iw}} t) \circ_{\mathrm{iw}} u &= (\langle x_1, E_1\rangle \circ_{\mathrm{iw}} \langle x_2, E_2\rangle) \circ_{\mathrm{iw}} \langle x_3, E_3\rangle \\
&= \langle x_1 \circ_{\mathrm{iw}} x_2, E_1 \cup E_2\rangle \circ_{\mathrm{iw}} \langle x_3, E_3\rangle \\
&= \langle (x_1 \circ_{\mathrm{iw}} x_2) \circ_{\mathrm{iw}} x_3, (E_1 \cup E_2) \cup E_3\rangle \\
&= \langle x_1 \circ_{\mathrm{iw}} (x_2 \circ_{\mathrm{iw}} x_3), E_1 \cup (E_2 \cup E_3)\rangle \\
&= \langle x_1, E_1\rangle \circ_{\mathrm{iw}} \langle x_2 \circ_{\mathrm{iw}} x_3, E_2 \cup E_3\rangle \\
&= \langle x_1, E_1\rangle \circ_{\mathrm{iw}} (\langle x_2, E_2\rangle \circ_{\mathrm{iw}} \langle x_3, E_3\rangle) \\
&= s \circ_{\mathrm{iw}} (t \circ_{\mathrm{iw}} u),
\end{aligned}$$

$$\begin{aligned}
s \parallel_{\mathrm{iw}} t &= \langle x_1, E_1\rangle \parallel_{\mathrm{iw}} \langle x_2, E_2\rangle \\
&= \langle x_1 \parallel_{\mathrm{iw}}^{E_1 \cap E_2} x_2, E_1 \cup E_2\rangle \\
&= \langle x_2 \parallel_{\mathrm{iw}}^{E_2 \cap E_1} x_1, E_2 \cup E_1\rangle \\
&= \langle x_2, E_2\rangle \parallel_{\mathrm{iw}} \langle x_1, E_1\rangle \\
&= t \parallel_{\mathrm{iw}} s,
\end{aligned}$$

$$\begin{aligned}
(s \parallel_{\mathrm{iw}} t) \parallel_{\mathrm{iw}} u &= (\langle x_1, E_1\rangle \parallel_{\mathrm{iw}} \langle x_2, E_2\rangle) \parallel_{\mathrm{iw}} \langle x_3, E_3\rangle \\
&= \langle x_1 \parallel_{\mathrm{iw}}^{E_1 \cap E_2} x_2, E_1 \cup E_2\rangle \parallel_{\mathrm{iw}} \langle x_3, E_3\rangle \\
&= \langle (x_1 \parallel_{\mathrm{iw}}^{E_1 \cap E_2} x_2) \parallel_{\mathrm{iw}}^{(E_1 \cup E_2) \cap E_3} x_3, (E_1 \cup E_2) \cup E_3\rangle \\
&= \langle x_1 \parallel_{\mathrm{iw}}^{E_1 \cap (E_2 \cup E_3)} (x_2 \parallel_{\mathrm{iw}}^{E_2 \cap E_3} x_3), (E_1 \cup E_2) \cup E_3\rangle \\
&= \langle x_1, E_1\rangle \parallel_{\mathrm{iw}} \langle x_2 \parallel_{\mathrm{iw}}^{E_2 \cap E_3} x_3, E_2 \cup E_3\rangle \\
&= \langle x_1, E_1\rangle \parallel_{\mathrm{iw}} (\langle x_2, E_2\rangle \parallel_{\mathrm{iw}} \langle x_3, E_3\rangle) \\
&= s \parallel_{\mathrm{iw}} (t \parallel_{\mathrm{iw}} u).
\end{aligned}$$

⊠

Proposition 11 For closed $IWE(\circ, +, \|)$-terms s and t such that $Ent(s) \cap Ent(t) = \emptyset$ we have

$$s \parallel_{\mathrm{iw}} t = s \circ_{\mathrm{iw}} t.$$

Proof. By the elimination theorem it is allowed to restrict the proof of the statements to basic terms. Let $s = \langle x, E\rangle$ and $t = \langle y, F\rangle$ for some $E, F \subseteq EID$ and closed $IWD(\circ, +, \cdot, \|^E)$-terms x and y such that $AE(x) \subseteq E$ and $AE(y) \subseteq F$.

$$\begin{aligned}
s \parallel_{\mathrm{iw}} t &= \langle x, E\rangle \parallel_{\mathrm{iw}} \langle y, F\rangle \\
&= \langle x \parallel_{\mathrm{iw}}^{\emptyset} y, E \cup F\rangle \\
&= \langle x \circ_{\mathrm{iw}} y, E \cup F\rangle
\end{aligned}$$

$$= \langle x, E \rangle \circ_{\mathrm{iw}} \langle y, F \rangle.$$

$\boxtimes$

8 Conclusions

The starting point of the application described in this chapter was the informal drawing technique, called Interworkings. After analysing the informal meaning of the language and the way in which users applied this language, our aim was to formalise the Interworking language.

The assets of having a formal semantics are well-known. We mention the following. Formalisation yields a thorough understanding of the language and the aspects of the application domain which can be modeled; it allows for an unambiguous interpretation of expressions in the language; it enables formal analysis; and it can be used to derive, or even automatically generate supporting tools.

These points directly addressed the problems that users were confronted with when applying the language. The language organically grew from a collection of examples and it was not clear which constructs were exactly part of the language. For some diagrams even specialists disagreed on the exact interpretation. It was not clear under which precise conditions two Interworkings could be merged. And, finally, in order to efficiently work with collections of Interworkings tool support was required.

The research carried out helped to solve these issues to a large extent. The kernel of the work was the description of the formal semantics of the language by means of process algebra. This is the part of the research covered in this chapter.

Our choice was to use process algebra for the formal definition of Interworkings. This worked out quite successfully. The process algebraic approach even proved suitable to define the semantics of a similar, but much larger language (MSC'96). Although it showed very beneficial, we do not advocate that the process algebraic approach is the best or even the only suitable approach towards the formalisation of sequence chart languages. Other techniques, such as Petri nets and partial orders, have also been successfully applied, and when considering only the core of these sequence chart languages, the several approaches do not differ too much with respect to expressivity and simplicity. Only when extending the sequence chart language with specific features, such as recursion and interrupts, some approaches offer a more natural way of modeling.

The work presented here only describes the part of the project which has to do with the theoretical foundations of the project. The main point here was to identify the basic Interworking constructs and operators, and to give their operational and algebraic semantics. The extension with a theory of refinement or the derivation of computer tools is not in the focus of this handbook.

Although already an overwhelming variety of operators has been described in process algebra literature, we have introduced yet more operators. This is typical for the process algebraic

approach. For a specific application domain a specific algebra is needed. In the case of sequence charts, the standard operators for sequential and parallel composition do not properly describe the user's intuition. Because the synchronisation implied by strong sequential composition is in contradiction with intuition, we developed the interworking sequencing. Because the standard parallel composition operator could not deal with overlapping areas of an Interworking, we had to investigate a variation: interworking merge. Even though these are newly invented operators, their definitions resemble the definition of well-studied operators.

This approach of defining new operators and variations on existing operators has been illustrated in this chapter. We have treated all proof obligations, such as soundness and completeness in full detail. We have especially taken care of setting up our theory in a modular way. This means that we have first defined the kernel of the theory (i.e. the semantics of single Interworking diagrams) and subsequently extended this with other operators.

The kernel of our theory just consists of the interworking sequencing operator. This single operator already allows for the definition of the semantics of Interworking diagrams. After that, we defined the basic process algebra consisting of the standard operators for alternative and sequential composition, extended with a special constant for expressing partial deadlocks. The alternative composition operator is used to express alternative scenarios. This process algebra is independent of the previous one, and the next module simply consisted of the combination of these two theories. The interworking sequencing can now be expressed in terms of the other operators. The axioms defining the interworking sequencing in the first process algebra are now derivable properties. Finally, we extended this algebra with the interworking merge operator. This required two separate steps. First we introduced the E-interworking merge, which is parameterised by the set of entities which should synchronise. And next, we extended the semantical interpretation of Interworkings in order to be able to define the unparameterised interworking merge. This modular approach is illustrated in Figure 17.

In our opinion, such a modular approach brings several assets. A mathematical theory, just like a piece of software, requires maintenance. Parts of the theory may become obsolete due to new insights or new extensions may be required due to additional user requirements. A modular theory makes it easier to isolate the parts of the theory which are affected by such modifications. A modular design also reduces the impact of a misdesign of one or more concepts. The modules defining the other concepts can easily be reused, while replacing the inappropriate concepts. An example of such a misdesign could be the interworking merge. Contrary to the interworking sequencing, which seems to be very stable and well accepted, several alternatives for the interworking merge have been proposed in literature (such as the *environmental gate merge*, see [RGG95]). The part of the algebra describing the interworking merge can easily be replaced by a definition of another similar operator.

9 Bibliographical notes

In this section we will give a comprehensive overview of the relevant literature on Interworkings and the related language Message Sequence Chart (MSC).

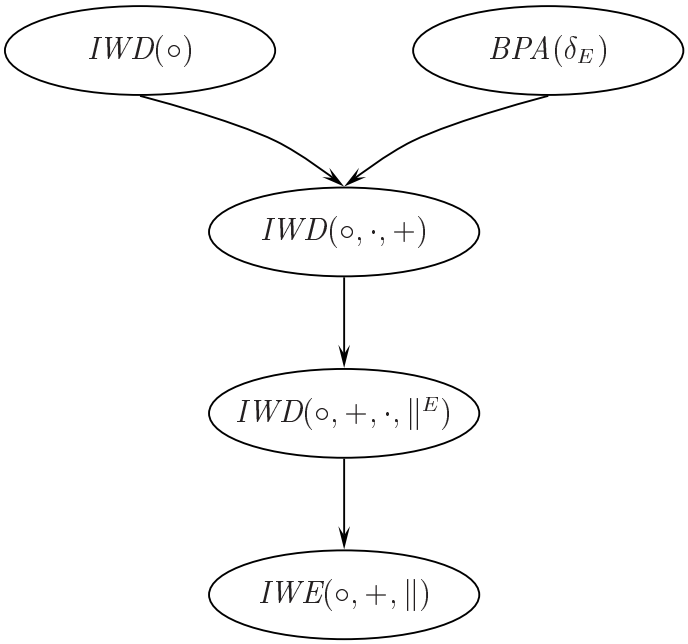

Figure 17: Overview of conservative extensions

Interworkings

In [MvWW92], Mauw, Van Wijk and Winter give a concrete textual syntax for the language IW and present both an informal and a formal definition of its semantics. The formal semantics does not consider entities without events (empty entities). A short version appeared as [MvWW93].

Based on the work on the formal semantics several prototype tools have been developed. A description of the prototype tool set is given in [MW93]. This tool set consists of three parts, the interworking processor (IWP), the intermediate language compiler (ILC) and a term rewriting system (TRS).

The formal semantics of Interworkings is not able to deal with empty entities and refinement. This has been solved in [MR95a, MR95b, MR96].

The deduction rules for $\lfloor\!\lfloor^E_{\mathrm{iw}}$ are different from the deduction rules used in [MR96] in the sense that the termination behaviour of $\|^E_{\mathrm{iw}}$ is coded in the termination behaviour of $\lfloor\!\lfloor^E_{\mathrm{iw}}$ instead of using the termination operator $\surd$ used there to describe the termination behaviour of E-interworking merge. There are two reasons for this change. First, in [Vra97] and [Ver97] also the termination behaviour is described with the left-merge operator. Second, it is easier to define the set of active entities of a process term $x \lfloor\!\lfloor^E_{\mathrm{iw}} y$ in this case.

In [MR96] we reported the following. For closed $IWD(\circ, +, \cdot, \|^E)$-terms x, y, z, and sets of entities E_1, E_2, E_3 we have

$$(x \|^{E_1 \cap E_2}_{\mathrm{iw}} y) \|^{(E_1 \cup E_2) \cap E_3}_{\mathrm{iw}} z = x \|^{E_1 \cap (E_2 \cup E_3)}_{\mathrm{iw}} (y \|^{E_2 \cap E_3}_{\mathrm{iw}} z).$$

This is not true. In the case that x can execute an action a such that $AE(a) \not\subseteq E_1$ and $AE(a) \subseteq E_2 \cap E_3$ the equation does not hold.

This can be explained as follows. The sets E_1, E_2, and E_3 are intended to model the instances of x, y, and z respectively. In the situation sketched above we have that x executes an action defined on an instance that does not belong to x! Here we presented an improved and correct version of this proposition.

The interworking merge as defined in [MvWW93] did not have the associativity property. This difference is a direct consequence of our decision to maintain the entities of an Interworking statically.

In [BG95] the language Interworking is extended with discrete absolute time features. Events can have a discrete time stamp or a discrete time interval associated with them. The authors describe the timed versions of interworking sequencing and interworking merge.

In [Fei99], Feijs uses Interworkings as a starting point for generating finite state machines. This is useful for obtaining feedback from a set of scenarios (Interworkings) during a system's definition phase or test phase.

In [Fei97], possibilities and impossibilities of using Interworkings are studied in the context of describing a service, a protocol, or a protocol entity in the OSI reference model on different levels of abstraction. The author concludes that Interworkings are useful for analysing a limited number of interesting cases such as test runs, simulation runs, and debug sessions, but also that Interworkings lack sufficient power to act as a specification formalism.

Message Sequence Charts

From the vast amount of graphical languages that resemble Interworking the language Message Sequence Chart, which is standardised by Study Group 10 of Question 9 of the Telecommunications Standardisation Sector of the International Telecommunication Union, is best known. The language MSC describes the asynchronous communication between instances (entities). The language is very rich in its syntax and has a standardised formal semantics [IT95, Ren99]. This formal semantics is inspired by the work on the formal semantics of Interworking. In [MR94a] a process algebra semantics of Basic MSC (only simple diagrams) is given. In [MvdM95], prototype tools are defined based on this formal semantics. In [Ren94, MR94b] the formal semantics of Basic MSCs is extended to the language MSC92 except for instance decomposition and conditions. Later, this semantics is standardised as Annex B to Recommendation Z.120 [IT95]. Also De Man [Man93] gives a process algebra semantics for Basic MSC. In [MR97], High-level Message Sequence Charts are treated. In [MR99, Ren99], an operational semantics for a large fragment of MSC96 is presented.

Besides the literature on the semantics of MSC based on process algebra, we also mention some other approaches. In [GGR93], an MSC is transformed into a Petri net. In [LL95], a semantics of Message Flow Graphs is presented that translates an MSC into a Büchi automaton. In [AHP96], Alur, Holzmann and Peled, present a partial order semantics for Basic Message Sequence Charts.

In the ITU Recommendation Z.120, the only assumption about communication between entities is that it is asynchronous and that sending of a message occurs before its receipt. In [EMR97], communication is discussed based on FIFO buffers. A variety of communication models is obtained by considering different ways of connecting entities through buffers. A hierarchy of these communication models is presented based on the possibility of implementing MSCs in the communication models. One of the communication models is identified with the synchronous communication in Interworkings.

References

[Aal99] W.M.P. van der Aalst. Interorganizational workflow: An approach based on Message Sequence Charts and Petri Nets. *Systems Analysis - Modelling - Simulation*, 34(3):335–367, 1999.

[AHP96] R. Alur, G.J. Holzmann, and D. Peled. An analyzer for Message Sequence Charts. *Software - Concepts and Tools*, 17(2):70–77, 1996.

[BG95] J. van den Brink and W.O.D. Griffioen. Formal semantics of Interworkings with discrete absolute time. In A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors, *Algebra of Communicating Processes, Utrecht 1994*, Workshops in Computing, pages 106–123. Springer-Verlag, 1995.

[BM95] J.C.M. Baeten and S. Mauw. Delayed choice: an operator for joining Message Sequence Charts. In D. Hogrefe and S. Leue, editors, *Formal Description Techniques VII*, Proceedings of the Seventh IFIP WG 6.1 International Conference on Formal Description Techniques, pages 340–354. Berne, Chapman & Hall, 1995.

[BV93] J.C.M. Baeten and C. Verhoef. A congruence theorem for structured operational semantics with predicates. In E. Best, editor, *CONCUR'93, International Conference on Concurrency Theory*, volume 715 of *Lecture Notes in Computer Science*, pages 477–492. Springer-Verlag, 1993.

[BV95] J.C.M. Baeten and C. Verhoef. Concrete process algebra. In S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, *Semantic Modelling*, volume 4 of *Handbook of Logic in Computer Science*, pages 149–268. Oxford University Press, 1995.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1990.

[EMR97] A. Engels, S. Mauw, and M.A. Reniers. A hierarchy of communication models for Message Sequence Charts. In T. Mizuno, N. Shiratori, T. Higashino, and A. Togashi, editors, *Formal Description Techniques and Protocol Specification, Testing and Verification*, Proceedings of FORTE X and PSTV XVII '97, pages 75–90, Osaka, Japan, November 1997. Chapman & Hall.

[Fei97] L.M.G. Feijs. Synchonous sequence charts in action. *Information and Software Technology*, 39:583–606, 1997.

[Fei99] L.M.G. Feijs. Generating FSMs from Interworkings. *Distributed Computing*, 12(1):31–40, 1999.

[GGR93] J. Grabowski, P. Graubmann, and E. Rudolph. Towards a Petri net based semantics definition for Message Sequence Charts. In O. Færgemand and A. Sarma, editors, *SDL'93 - Using Objects*, Proceedings of the Sixth SDL Forum, pages 179–190, Darmstadt, 1993. Amsterdam, North-Holland.

[IT93] ITU-T. *Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-T, Geneva, September 1993.

[IT95] ITU-T. *Recommendation Z.120 Annex B: Algebraic semantics of Message Sequence Charts*. ITU-T, Geneva, April 1995.

[LL95] P.B. Ladkin and S. Leue. Interpreting message flow graphs. *Formal Aspects of Computing*, 7(5):473–509, 1995.

[Man93] J. de Man. Towards a formal semantics of Message Sequence Charts. In O. Færgemand and A. Sarma, editors, *SDL'93 - Using Objects*, Proceedings of the Sixth SDL Forum, pages 157–165, Darmstadt, 1993. Amsterdam, North-Holland.

[MR94a] S. Mauw and M.A. Reniers. An algebraic semantics of Basic Message Sequence Charts. *The Computer Journal*, 37(4):269–277, 1994.

[MR94b] S. Mauw and M.A. Reniers. An algebraic semantics of Message Sequence Charts. Technical Report CSN 94/23, Eindhoven University of Technology, Department of Computing Science, Eindhoven, 1994.

[MR95a] S. Mauw and M.A. Reniers. Empty Interworkings and refinement - semantics of Interworkings revised. Technical Report CSR 95-12, Eindhoven University of Technology, Department of Computing Science, 1995.

[MR95b] S. Mauw and M.A. Reniers. Empty Interworkings and refinement - semantics of Interworkings revised. In *ACP'95, Proceedings of the Second Workshop on Algebra of Communicating Processes*, number CSR 95/14 in Computing Science Reports, pages 367–385. Eindhoven University of Technology, Department of Computing Science, 1995.

[MR96] S. Mauw and M.A. Reniers. Refinement in Interworkings. In U. Montanari and V. Sassone, editors, *CONCUR'96*, volume 1119 of *Lecture Notes in Computer Science*, pages 671–686, Pisa, Italy, 1996. Springer-Verlag.

[MR97] S. Mauw and M.A. Reniers. High-level Message Sequence Charts. In A. Cavalli and A. Sarma, editors, *SDL'97 : Time for Testing - SDL, MSC and Trends*, Proceedings of the Eighth SDL Forum, pages 291–306, Evry, France, 23-26 September 1997. Amsterdam, North-Holland.

[MR99] S. Mauw and M.A. Reniers. Operational semantics for MSC96. *Computer Networks amd ISDN Systems*, 31(17):1785–1799, June 1999. Special issue on Advanced topics on SDL and MSC, guest editor, A. Cavalli.

[MvdM95] S. Mauw and E.A. van der Meulen. Generating tools for Message Sequence Charts. In R. Bræk and A. Sarma, editors, *SDL'95 - with MSC in CASE*, Proceedings of the Seventh SDL Forum, pages 51–62, Oslo, 1995. Amsterdam, North-Holland.

[MvWW92] S. Mauw, M. van Wijk, and T. Winter. Syntax and semantics of synchronous Interworkings. Technical Report RWB-508-re-92436, Information and Software Technology, Philips Research, 1992.

[MvWW93] S. Mauw, M. van Wijk, and T. Winter. A formal semantics of synchronous Interworkings. In O. Færgemand and A. Sarma, editors, *SDL'93 - Using Objects*, Proceedings of the Sixth SDL Forum, pages 167–178, Darmstadt, 1993. Amsterdam, North-Holland.

[MW93] S. Mauw and T. Winter. A prototype toolset for Interworkings. *Philips Telecommunication Review*, 51(3):41–45, 1993.

[Ren94] M.A. Reniers. An algebraic semantics of Message Sequence Charts. Master's thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology, April 1994.

[Ren99] M.A. Reniers. *Message Sequence Chart: Syntax and Semantics.* PhD thesis, Eindhoven University of Technology, June 1999.

[RGG95] E. Rudolph, P. Graubmann, and J. Grabowski. Message Sequence Chart: composition techniques versus OO-techniques - 'tema con variazioni'. In R. Bræk and A. Sarma, editors, *SDL'95 - with MSC in CASE*, Proceedings of the Seventh SDL Forum, pages 77–88, Oslo, 1995. Amsterdam, North-Holland.

[RJB99] J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual.* Addison-Wesley, 1999.

[RW94] A. Rensink and H. Wehrheim. Weak sequential composition in process algebras. In B. Jonsson and J. Parrow, editors, *CONCUR'94: Concurrency Theory*, volume 836 of *Lecture Notes in Computer Science*, pages 226–241, Uppsala, 1994. Springer-Verlag.

[Ver94] C. Verhoef. A general conservative extension theorem in process algebra. In E.-R. Olderog, editor, *Programming Concepts, Methods and Calculi (PROCOMET'94)*, volume 56 of *IFIP Transactions A: Computer Science and Technology*, pages 149–168. Elsevier Science Publishers B.V., 1994.

[Ver95] C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2(2):274–302, 1995.

[Ver97] J.J. Vereijken. *Discrete-time process algebra.* PhD thesis, Eindhoven University of Technology, 1997.

[Vra97] J.L.M. Vrancken. The algebra of communicating processes with empty process. *Theoretical Computer Science*, 177(2):287–328, 1997.