Incremental Modeling Under Large-Scale Distributed Interaction

Horst F. Wedde, Arnim Wedig, Anca Lazarescu,
Ralf Paaschen, and Elisei Rotaru

University of Dortmund, Informatik III, 44221 Dortmund, Germany
isystems@ls3.cs.uni-dortmund.de

Abstract. We present I–Systems as a formal constraint-based approach for modeling and analyzing both autonomous and reactive behavior in a distributed system. Essentially it is a formalism of *interacting finite automata*. We demonstrate its incremental potential by stepwise modeling a solution for a synchronous communication problem.

1 I–Systems

In practice, interaction in a distributed system is based on local (or regional) cooperation between a small number of components. Global effects arise from a propagation of influences originating from constraints on local or regional cooperation. The component behavior exhibits two different types of events: *enforced* events *will* occur, either due to local control decisions or through external influences that trigger these events, *free* events *may* (or may never) occur based on autonomous local decisions, or because of incomplete information about yet unknown external influences.

Based on the observation that components in distributed systems may be reactive or active, we distinguish between *reactive (inert)* and *autonomous* components called *parts*. Parts have a constituting set of local states (that are relevant for the interaction) which are called *phases*. *Also, parts are in exactly one phase at any time*. In this way *parts are finite automata*. In contrast to communicating automata the cooperation, influences, decisions and their propagation, even the internal behavioral details within a component are defined based only on two types of binary relations (denoted as *coupling and excitement relation*, see section 2) between parts. It turns out that the I-System model has a higher specificational power than communicating finite automata.

We Will Present I-Systems as Interacting Finite Automata. The interaction is specified though local *action rules* which describe the frame of actions in parts (i.e. occurrences of phase transitions), as well as the ensuing influence on, or from, other parts. A technical comprehensive presentation of the static structure, the dynamics, and the semantics of I-Systems can be found in [3,4].

F. Wang (Ed.): FORTE 2005, LNCS 3731, pp. 542–546, 2005.

2 Modeling of Sequential Processes

As mentioned above each part of an I–System can be interpreted as a finite automaton. In order to specify its transition structure we focus on installing *interaction primitives* (instead of communication primitives for communicating automata). As a result, free and enforced phase transitions, as well as their enforcing influences, can be modeled explicitly.

As an example let us assume an autonomous part b_1 with four phases $p_1, \ldots, p_4$. Our goal is to impose a cyclic behavior structure in b1 such that only free phase transitions from p_1 to p_2 and from p_3 to p_4, and enforced phase transitions from p_2 to p_3 and from p_4 to p_1 may/will occur. We realize this by simply adding an additional inert part $\underline{b}_2$ with phases $e_1, \ldots, e_4$, and by stepwise adapting the coupling and excitement relation as depicted in Fig. 1.

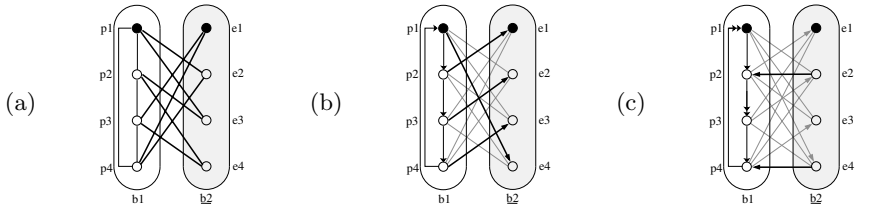

Fig. 1. Incremental Design of Sequential Processes

Step 1 (Fig. 1.a): Adding coupling relations between b_1 and $\underline{b}_2$
The symmetrical coupling relation specifies pairs of *mutually exclusive phases*.
Result: We restrict the local behavior to phase transitions between p_i and $p_{(i+1)\ mod\ 4}$ and back, induced by the reactive part $\underline{b}_2$.

Step 2 (Fig. 1.b): Adding excitement relations directed from b_1 to b_2
In general, an element $(p, q) \in E$ where E is the *excitement relation* expresses a ***potential excitation from phase p in part b to phase q in part b′***. The *main idea* is that if b is in p and b' is in q then b exerts an influence on b' to leave p, and b' will leave p unless prevented through other external influences. B, in turn, has to stay in p as long as b' is in q.
Result: We install directions in b_1, i.e. phase transitions only occur from p_i to $p_{(i+1)\ mod\ 4}$.

Step 3 (Fig. 1.c): Adding excitement relations directed from b_2 to b_1
Result: We enforce outgoing phase transitions from p_2 as well as from p_4.

Major Result: *In general each organizational form of behavior in a part as specified through autonomous decision effects or dutiful steps according to an organizational role can be modeled through the incremental standard construction explained in the example above [3,4].*

3 Application: Synchronous Communication

We model, for a correct implementation, the synchronous communication concept of CSP, cf. [1,2]. In order to establish such a communication between a process (part) P_1 and a process (part) P_2, P_1 executes a *tie* command (when arriving in phase ti_1). As a result a virtual channel is opened which would be connected to a corresponding virtual channel on the side of P_2 (in phase ti_2). In this way the real communication between P_1 and P_2 could start (phases co_1, co_2). After completion the virtual channels will be abandoned by executing an *untie* command (in phases un_1, un_2).

The **synchronization conditions for P_1** are formulated in the following way. They must hold **symmetrically for P_2**.

SC1: If P_1 has arrived in ti_1 while P_2 has not entered ti_2 or co_2, P_1 has to wait.
SC2: If P_1 has arrived in co_1 while P_2 is still in ti_2 then P_1 cannot proceed, and P_1 exerts an influence on P_2 as to leaving ti_2.
SC3: If P_1 has arrived in un_1 while P_2 is still in co_2 then P_1 cannot proceed, and P_1 exerts an influence on P_2 as to leaving co_2.
SC4: If P_1 is in ti_1 and P_2 is outside of its communication section (i.e. P_2 is in a remainder phase re_2) then there is no influence on P_2 from P_1 as to enter its communication section.

The local behavior in the parts P_1 and P_2 can be realized as in section 2, detailing free and enforced phase transitions. We stepwise realize SC1 - SC4 through the symmetric construction shown in the screenshot of Fig. 1.

Step 1: SC2 is realized through connecting P_1 and P_2 through the inert part $\underline{\boldsymbol{VE}}_1$.
Step 2: SC3 is realized through connecting P_1 and P_2 through the inert part $\underline{\boldsymbol{VE}}_2$.

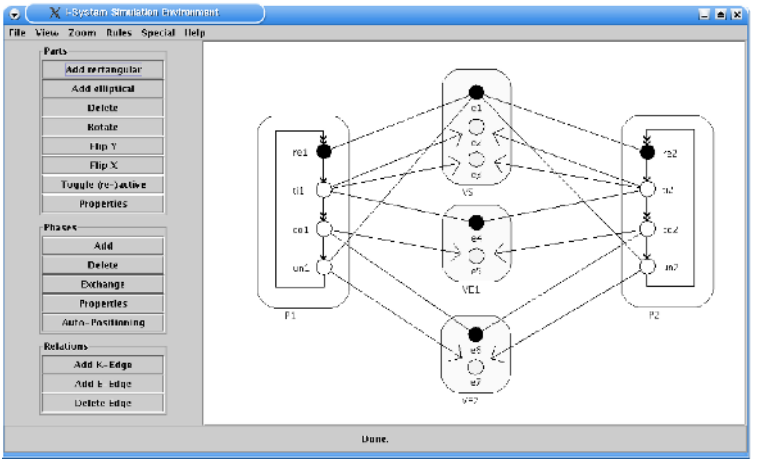

Fig. 2. Incremental Realization of Synchronous Communication

Step 3: SC1 and SC4 are both realized through connecting P_1 and P_2 through the inert part $\underline{\boldsymbol{VS}}$.

Major Property: *The interaction is minimal in the sense that through this connection no further restriction is imposed on* P_1 *or* P_2.

As we readily verify this property is easy to achieve through the explicit specification utilized above while in other models, including communicating automata, this has always been the weakest part of the proposed solutions (see e.g. [1]).

4 Conclusion

We have presented the advantages of I–Systems in modeling distributed interaction, a formalism corresponding to an extended model of interacting finite automata (as opposed to communicating finite automata).

We have demonstrated the stepwise realization of a local behavior structure through primitive interaction constructions, connecting the involved parts through a reactive component. An I–System model for a distributed system reflecting all kinds of cooperative requirements or constraints can be incrementally constructed in this way such that the absence of undesirable influences in subsystems can be guaranteed.

We have developed a fundamental formal framework for I–Systems, e.g.:

- We have developed a novel abstract axiom system for specifying and deriving the behavior of an I–System. Behavioral steps (phase transitions) are derived by local checks of the local constraint structure. The axiom system is motivated through a *distributed implementation* and an *efficient animator.*
- We have defined a trace-semantics that documents the effects of influences and allows to distinguish between *free* and *enforced actions* (phase transitions).
- We have defined a finite behavior graph that is equivalent to the (infinite) trace semantics in that they describe the same behavior.
- We currently investigate efficient analysis and model-checking techniques for I–Systems, i.e. finding cycles in the behavior graph in order to identify infinite traces in the trace-semantics.

The presentation of formal details is out of the scope of this short paper. They can be found in [3,4]. *An on-line animation including the examples as well as the example constructions described will be demonstrated at the conference.*

References

1. Castelli, G., De Cindio, F., De Michelis, G., Simone, C.: The GCP Language and its Implementation. In Proc. of the IFIP workshop *Languages for Automation*, New Orleans, October 1984.
2. Hoare, C. A. R.: Communicating Sequential Processes. CACM, vol. 21, no. 8, 1978.

3. Wedde, H. F., Wedig, A.: Explicit Modeling of Influences, and of Their Absence, in Distributed Systems. In *Tools and Algorithms for the Construction and Analysis of Systems (ETAPS/TACAS'02)*, volume 2280 of LNCS, pages 127–141. Springer, 2002. Ext. Version: `http://ls3-www.cs.uni-dortmund.de/I-Systems/P/techrep742.pdf`
4. Wedde, H. F., Wedig, A., Lazarescu, A., Paaschen, R., Rotaru, E.: Model Building and Model Checking under Large-Scale Distributed Interaction. Technical report, University of Dortmund, April 2005. `http://ls3-www.cs.uni-dortmund.de/I-Systems/P/techrep795.pdf`