Restoring the Concept of Observability in E-LOTOS

M. Kapus-Kolar [1]

Jožef Stefan Institute, POB 3000, SI-1001 Ljubljana, Slovenia

Abstract

In E-LOTOS, signals are a kind of external actions of a process that the environment of the process is not allowed to synchronise upon. They can only be trapped as exceptions, but that prevents observation of the subsequent actions. Consequently, the coupling of a process and its tester can not always be adequately described in E-LOTOS. To solve the problem, we introduce the concept of passive observers.

Key words: Formal methods; LOTOS; E-LOTOS; Specification; Testing

1 Introduction

In the client/server paradigm, the only really relevant property of an object in the role of a server is the service it offers to its clients. Its internal behaviour is of secondary importance, just implementing the external behaviour.

According to the ISO Reference Model for Open Distributed Processing (ODP) [5], an object is an abstract representation of some real or abstract thing of interest, e.g. of a system as a whole, of a subsystem, or of a particular view of a system. The behaviour of an object is defined as the actions in which the object can engage, together with the constraints on when the object is ready for them. The external actions of an object are interactions between the object and its environment through their interface. The internal actions of an object serve for example for externally invisible interaction between its constituting objects. The objects in the environment of an object act as its observers, i.e. its more or less efficient testers, even if not designed exactly with that purpose in mind. By connecting to some objects constituting its environment, an object

[1] Email: monika.kapus-kolar@ijs.si

can become a part of a composite object, whose observers are the objects outside the group.

Long before ODP standards, the needs of the rapidly developing field of telecommunications have stimulated ISO to promote formal languages for specification of communicating objects. The languages standardised for the purpose have been SDL [2], Estelle [3] and LOTOS [1]. Among the three, LOTOS is the one most clearly capturing the semantics of the above basic concepts of ODP, particularly the concept of action observability.

The subject of our paper is a semantical problem in E-LOTOS [6], an enhanced successor of LOTOS particularly intended for specification of ODP and currently approaching standardisation. The problematic point is that E-LOTOS treats actions in a non-uniform way. Although it declaratively allows urgent actions visible to the environment of an object, they are not visible in the full sense of the word, for the environment is not allowed to synchronise on them as it can on non-urgent actions. Urgent actions can only be trapped as exceptions, but that prevents observation of the subsequent actions. Consequently, the coupling of an object and its tester can not always be adequately described in E-LOTOS. In practice, that makes E-LOTOS inadequate for specification of testers and for reasoning about testing activities. In theory, that corrupts the concept of action observability and makes E-LOTOS in that aspect inferior to LOTOS. To solve the problem, we introduce the concept of passive observers.

2 Specifying Action-observing in LOTOS

In LOTOS, the entities corresponding to the ODP objects are called processes. An external action of a process is defined to be an action executed on a gate visible to the environment of the process. By enabling it, the process offers to interact with the other processes connected to the gate. An interaction is executable when all the partners are ready for it, as their common action. An internal action of a process might be internal by its nature, or be an action on a gate that has been hidden from the environment by the LOTOS gate-hiding operator.

If a tester connects to a visible gate of a process and is always ready to participate in the actions on it, the actions are executed exactly when so decided by the tested process itself. Obviously it is possible to observe external actions of a process without interfering with them. Hence the external actions of a process are exactly its observable actions, as expected from their name.

A tester (or a client, or a partner) of a process usually connects only to those

external gates that are of its interest (an exhaustive tester to all of them). LOTOS defines a parallel composition operator by which such coupling of a process and its observer can be specified. The operator takes a pair of processes and specifies through which of their gates they are connected. On a non-connected gate, an individual process can execute actions on its own, without waiting for co-operation of the partner. Likewise, a process can independently execute its internal actions. If a group of processes is connected on a gate, but the gate is not hidden, it is (like the gates not used for intra-group communication) a visible gate of the group, available for synchronisation with processes in its environment. In that manner, multiway synchronisation can be specified, although the LOTOS parallel composition operator is only binary.

A specific external action in LOTOS is δ, the action denoting successful termination of a process. After its execution, a process by definition becomes inactive. A group of processes running in parallel can by definition execute δ only as a common action, i.e. their δ gates are by definition connected. Hence for a thorough testing of a process, its tester must be connected to it on all its external gates and on δ.

After a process successfully terminates, its successor process is activated, if any. There is a special operator for connecting processes in sequences. In the points where δ serves for sequential transfer of control, it is by definition a hidden action. Such a δ of a process is not directly observable, but one can infer that it has been executed from observing that the subsequent process has been activated.

3 Specifying Action-observing in E-LOTOS

E-LOTOS allows modelling of time-sensitive systems. It classifies actions into urgent and non-urgent ones. When a process enables an urgent action, it must by no means be prevented from executing it immediately, but it may of course decide to immediately execute an alternative action instead. Speaking formally, an urgent action has a higher priority than the passage of time, but not a higher priority than other actions.

A process can not be prevented from executing an internal action, for it can always execute it on its own. So E-LOTOS simply defines that all internal actions are urgent, to allow processes to quickly proceed to the subsequent external actions, i.e. the actions constituting the service they are offering.

δ is also defined to be urgent, to allow quick sequential transfer of control. We have mentioned, however, that a process B can execute a δ only in co-operation with the processes that are in parallel composition with itself. As

some of the processes might not be immediately ready, the E-LOTOS semantics exceptionally allows B to wait until the co-operation is available.

If an external urgent action belongs to a gate other than δ, we have a problem. E-LOTOS, more precisely its parallel composition operator, does not allow other processes to connect to the gate, for that would allow them prevent the urgent action by non-co-operation. The only way E-LOTOS offers for observing such an action is trapping.

Trapping of an urgent action A in a process B diverts control from B to a B', the handler of A in B, so that by observing actions in B' one can usually infer that A in B has been enabled. The transfer of control is atomic, i.e. A acts as an implicit guard of B' without occurring as an explicit action preceding B'. A can also be interpreted as a place-holder for B', much like a name p of a separately specified process with behaviour B' would be. The main difference between trapping of A and instantiation of p is that trapping permanently diverts control from B, while after a successful termination of p B would be resumed. Trapping is also available for δ, for the ordinary sequential transfer of control from a successfully terminated B to its successor process. In other words, trapping of δ is equivalent to sequential composition.

If the trapped A is an exception raised within B, then permanent diverting of control to B' is an adequate means for facilitating its observation, for A is by itself a (non-successful) termination of B. If, however, A is just a signal, it might be followed by other actions in B, that are by the trapping of A made irrelevant, for B never regains control to enable them.

Hence we have a controversial situation: 1) We are not allowed to observe A by synchronising on it. 2) If we observe it by trapping instead, we interfere with the subsequent actions of B. 3) If we don't trap A, we might still be able to observe the subsequent actions, but we can never know whether B has preceded them with A or not.

As an example, take a simple process P with the specified behaviour

A_1;**i**;**signal** A_2;**i**;**signal** A_2; A_3

i.e. expected to execute a non-urgent A_1, twice an internal **i** followed by an urgent A_2, and a non-urgent A_3 in a sequence, where the two signals report the two internal actions, as suggested in the tutorial section of [6].

- If synchronisation on all kinds of external actions was allowed, a tester could observe the external behaviour of P simply by connecting to all the three gates A_1, A_2 and A_3, and enabling "$A_1; A_2; A_2; A_3$".
- In the current version of E-LOTOS, a tester is only allowed to synchronise on the non-urgent actions A_1 and A_3, so that its observation would be

"$A_1; A_3$", without knowing whether the testee has preceded A_3 with two A_2 or not.

- Alternatively, we could treat actions A_2 as exceptions and attach to P a handler for them. The composite process could for example be
  trap exception A_2 **is** A_4 **endexn in** P
  i.e. the first signal A_2 would be trapped and reported by a non-urgent A_4, but that would for ever divert control from P, preventing normal execution of "**i**;**signal** $A_2; A_3$". A tester synchronised to the composite process would at best observe $A_1; A_4$, concluding that the testee has enabled an A_2 after A_1.
- Even if we combined the knowledge gained from the two (an possibly some other) testing approaches, we could never test whether the first signal A_2 was followed by another, because trapping always detects just the first signal of a particular type. The problem would not exist if the second signal was given another name, but that would in practice signify that we expect systems never to flash any of their signalling lights more that once, if their development is to be based on E-LOTOS tools. Moreover, someone observing P not for testing purposes, but in the role of its client, can never benefit from its service in its entirety.

We conclude that E-LOTOS currently provides no adequate means for specifying in a general case a system of a process and its exhaustive tester, i.e. the concept of action observability, that is in LOTOS the key concept, is corrupted.

The above problem urgently needs a remedy. On the one hand, we don't want to give up urgent actions that are just signals, because we do meet such signals in reality, for example light flashes. On the other hand, system developers will certainly continue synchronising parallel processes on their external actions of all types, including the urgent ones, while no computer tool based on the current E-LOTOS semantics would be able to always predict the result of such a parallel composition, i.e. the quality of the tester or the meaning of individual testing outcomes. Thus in the following section, we propose a slight semantic extension to allow modelling of signal observations in a natural way.

4 The Concept of Passive Observers

The E-LOTOS parallel composition operator allows composition of an arbitrary number of processes. Although there has been further research [4] after [6], the idea that processes in E-LOTOS may synchronise only on non-urgent actions and on δ has not been abandoned.

E-LOTOS is currently based on the interleaving semantics, i.e. pretending

that actions always happen one after another, even if they actually happen simultaneously. In that semantics, a set S of processes in parallel composition evolves as follows:

- The current state of the composite process is determined by the states of the members of S.
- An evolution step of the composite process might be a time step, i.e. all processes in S idling for a particular amount of time. Such a step is executable when all the processes allow it, i.e. have no urgent actions specified in the time interval. The only effect that a time step has on the processes is their ageing, for E-LOTOS is based on the assumption that the mere flow of time can not influence the future behaviour of processes.
- An evolution step might also be an action A. In that case there must be an $S' \subseteq S$, the participants set of A, such that its members are capable to execute A as their immediate common action. First of all, each individual member of S' must be ready for A. The cardinality of S' might be 1 by definition, i.e. a process executing it as an urgent action or a non-urgent action on a gate not connected to any gate of the other processes in S. Alternatively, A might be an action on a gate which the parallel composition operator defines to be an interconnection gate for all the processes in S'. In LOTOS, an interaction A is executable only if S' includes all the processes in S for which the gate serves for communication within S. In E-LOTOS, however, we can specify that it is sufficient that only some of the partners are in S', i.e. that A is executable provided that S' has a predefined cardinality. For $A = \delta$, of course, $S' = S$. Upon A, the members of S' change their respective states as defined by their individual specifications, while the state of the rest of the processes remains unchanged.

An urgent action of a process is always an action that doesn't have idling as its alternative. If a member of S doesn't allow idling in its present state, because of an urgent action enabled, then the composite process is unable to idle, too. In other words, the urgent action is also urgent for the composite process as a whole, as naturally expected.

In E-LOTOS, a signal, i.e. an urgent action potentially problematic for testing, is hence always an action of an individual process. In reality, however, it may be an action of many processes, with one of them issuing the signal and the others acting as its passive observers. When, for example, a light flashes, it is seen by everybody currently looking in its direction. If a potential observer is currently not looking at the light, that can not prevent it from flashing. Hence a signal can well be urgent for its issuer. On the other hand, a signal must not be urgent for an observer, because he/she has no means to enforce its occurrence. If an observer is not ready for a signal in time, the signal is lost for him/her, like for example in SDL [2].

To implement the concept of passive observers in E-LOTOS, we propose to slightly extend the above described dynamic semantics of the parallel composition operator, by defining:

- An A that is currently an executable and urgent external action of a process B in S is executable by a participants set S' with B as its member, iff the other processes in S' are exactly the processes in S currently ready to execute A as their non-urgent external action.

In E-LOTOS, the readiness of a process to engage in an action of a particular type is specified by naming the action as a possible next action. E-LOTOS defines that urgent and non-urgent external actions are two strictly different syntactic categories. If a specification respects the rule, it can never happen that a set S of parallelly composed processes has a member with an A specified as an urgent, and another member with A as a non-urgent external action. Consequently, the cardinality of S' remains 1 for all urgent A, as necessary for the compatibility of the extension with the original E-LOTOS.

To implement passive observations, we obviously have to drop the above syntax rule. To solve the problem, we observe that it is quite sufficient if urgent and non-urgent actions are syntactically distinct within the external behaviour of each individual process. It would not be difficult to incorporate the requirement into the static semantics of E-LOTOS behaviour composition operators other than the parallel composition, because they are based exclusively on sequential and/or alternative execution of the composed processes or their parts.

For parallel composition, however, the requirement is more tricky. We implement it by the following static semantics rule:

- If a name A denotes non-urgent actions in the external behaviour of a process B, a member of a set S of parallelly composed processes, while for another member of S A denotes urgent external actions, then in the context of S, the readiness of B to participate in A should always be interpreted as its readiness to participate exclusively as a passive observer. In the opposite case, B is free to execute A as an ordinary external action, either on its own or in co-operation with other members of S, as specified by the original semantics of the parallel composition operator.

Hence like in the original E-LOTOS, an A can be non-urgent for the composite process only if it is non-urgent for each of the process in S. Likewise, the presence of passive observers doesn't invalidate the statement that an A urgent for a member of S is also urgent for the composite process. That is because idling in the critical states is prevented by the mere presence of the process pursuing A as urgent.

Let us also note that there might be several members of S occasionally enabling an A as urgent. With the above semantics that is not problematic, because when the composite process executes an A, exactly one of the processes executes it as urgent, i.e. signals never interfere.

To see the enhanced parallel composition operator in action, let's return to our example process P with behaviour "A_1;**i**;**signal** A_2;**i**;**signal** A_2; A_3".

- A possible tester would be "$A_1; A_2; A_2; A_3$" running in parallel to P and connected to its gates A_1 and A_3. For the A_2 of the tester, it is obvious that they are to be executed as passive observations, because they are urgent for P running in parallel. Because the tester would be ready for the signals in time, they would be observed, and so would be the entire external behaviour of P.
- If the tester enabled "$A_1; A_2; A_3$" instead, it would miss the second signal, but P would not be prevented from executing it on its own.
- If there was another tester running in parallel and ready to observe the signals of P, both testers would observe them.

Like for a light flash, observation of a signal by members of the group of parallel processes to which the signaller belongs does not make the signal invisible for processes outside the group. The principle is the same as for non-urgent intra-group interactions. For such interactions E-LOTOS provides an operator that makes them invisible to the environment of the composite process. With signals also serving for intra-group communication, it might be the case that we want them to be invisible to the environment, too. Therefore it would be appropriate to also enhance the hiding operator, to allow hiding of urgent actions. That would be easy, because the actions are already urgent, as expected for internal actions.

5 Conclusion

We have proposed a simple semantic enhancement to E-LOTOS to support the natural need for specification of passive signal observation. The enhancement seems to be indispensable particularly for specification of testers and for reasoning about testing activities. In addition, that would be a small step towards more uniform treating of urgent and non-urgent actions in E-LOTOS.

References

[1] T. Bolognesi and E. Brinksma, Introduction to the ISO specification language LOTOS, *Computer Networks and ISDN Systems* **14** (1987) 25–59.

[2] R. Bræk, SDL basics, *Computer Networks and ISDN Systems* **28** (1996) 1585–1602.

[3] S. Budkovski and P. Dembinski, An introduction to Estelle: A specification language for distributed systems, *Computer Networks and ISDN Systems* **14** (1987) 3–23.

[4] H. Garavel and M. Sighireanu, A graphical parallel composition operator for process algebras, in: *Proc. FORTE/PSTV'99*, Beijing, October 1999.

[5] ISO/IEC, *Open Distributed Processing – Reference Model*, IS 10746, ISO – Information Processing Systems, Genève, 1995.

[6] J. Quemada, editor, *Committee Draft on Enhancements to LOTOS*, ISO/IEC FCD 15437 (E-LOTOS) (also SC33 N188), April 1998.