Fault-Tolerant Design in the OUN Specification Language

J.Vain[1], O.Owe[2], E.B.Johnsen[2], E.Munthe-Kaas[2]
[1] Tallinn Technical University, ESTONIA
[2] University of Oslo, NORWAY

The capability to incorporate results of dependability analysis in terms of potential program errors and hardware failures and, on the other hand, descriptions corresponding to faults correcting measures in program specifications, constitute a significant challenge that must be addressed before any formal technique can become a feasible technique for large-scale software development.

The specification language OUN [1] is aimed at dealing with the main aspects of object oriented open distributed systems including mobility, flexibility, reflection and dynamic reconfiguration, and to support reasoning of correctness properties of such systems. OUN allows specification of contracts, interfaces and classes using high level object oriented concepts like multiple inheritance of interfaces and classes, and allowing implementation of several interfaces by a class. Interface specifications characterise the observable behaviour of one object, contracts restrict the behaviour of a subsystems with two or more objects, while class specifications characterise local attributes as well as internal and external behaviour. The behavioural part of interface and contract specifications is given in terms of communication histories satisfying assume and guarantee clauses of the specification. Both clauses are expressed using first order logic and regular expressions. In this work we focus particularly on the development of interface and contract specifications where occurrences of faults and tolerance measures become explicit as abstract specification evolves towards implementation of object methods.

Despite of large variety of specific fault classes and fault tolerance techniques treated in the literature (see [2] for overview), it turns out that on an abstract level the majority of faults and tolerances may be captured into a few basic classes. Generally we say that a system is fault tolerant if it satisfies its specification in the presence of faults. Depending on the degree the required properties are satisfied, three fault-tolerance classes have been proposed in [3]: *masking*, *non-masking,* and *fail-safe* fault tolerance. The masking fault tolerance preserves both safety and liveness properties. Non-masking tolerance guarantees that the system state being deviated from safety region after the occurrence of faults will return to the state satisfying both safety and liveness requirements. Fail-safe fault tolerance preserves only safety properties and may not return, after faults have ceased, to a state where liveness is guaranteed.

A multitolerant systems design technique applying the above given notions of fault tolerance classes was proposed by Arora and Kulkarni [3]. According to this model, possible fault classes are introduced and corresponding to them tolerances are designed incrementally one-by-one. Each such step is accompanied by a correctness proof showing that all tolerances achieved in previous steps are also preserved in the current step. Monotonicity of reduction of intolerances guarantees that the final system is multitolerant to all fault classes considered.

Although the incremental technique possesses several advantages specially concerning the reduction of verification complexity, we do not restrict application of OUN to only one technique. The hierarchical specification and proof method proposed by Lamport and Merz [4] considers (fault-tolerant) implementation of higher level specifications by larger steps where several fault-tolerances are introduced all together. To comply with both techniques three types of relations between specifications have to be characterised in OUN.

Consider an abstract level interface specification and two concrete level specifications, respectively an intolerant and a fault tolerant implementation of the abstract specification, we study the relations between these specifications. The relation between the abstract and the intolerant specifications is the usual OUN relation for stepwise refinement of interfaces. An interface *I'* is said to refine an interface *I* if the trace set of *I* contains the projection of the trace set of *I'* onto the alphabet of *I*.

Faults are represented in OUN through extension of the interface alphabet and through introduction of additional non-determinism into the trace set. The relation between the abstract specification and its fault-tolerant implementation depends on the fault tolerance-class the implementation provides and can be defined within the semantics of the OUN formalism using standard trace operations.

Finally the relation between the intolerant and the fault tolerant specifications defines the proof obligation for stepwise inclusion of fault tolerance in the specification in accordance with the fault tolerance-class considered.

The relations are all defined within trace semantics. An application of the method is demonstrated on an ATM system design example where dispenser mechanism and line failures constitute the set of faults and non-masking tolerance is designed for both. As to further work, extension to deal with real time is planned for the OUN formalism.

References

1. O. Owe, I. Ryl. On Combining Object Orientation, Openness and Reliability. In *Proceedings of the Norsk Informatikk Konferense NIK'99*, November 1999, Norway.

2. J.Rushby. Critical System Properties: Survey and Taxonomy. *Reliability Eng. and Systems*, vol. 43, pp.180-219,1994.

3. A.Arora and S.S. Kulkarni. Component Based Design of Multitolerant Systems. *IEEE Transactions on Software Engineering*, Vol. 24, No. 1, 1998.

4. L.Lamport and S.Mertz. Specifying and Verifying Fault-Tolerant Systems. In *Proceedings of the Third International Symposium on Formal Techniques in Real Time and Fault Tolerant Systems*, 19-23 September 1994, Lübec, Germany.