**Making reliable**

**distributed systems**

**in the presence of**

**software errors**

**Final version (with corrections) — last update 20 November 2003**

**Joe Armstrong**

**A Dissertation submitted to**

**the Royal Institute of Technology**

**in partial fulfilment of the requirements for**

**the degree of Doctor of Technology**

**The Royal Institute of Technology**

**Stockholm, Sweden**

**December 2003**

**Department of Microelectronics and Information Technology**

The work described in this thesis is the result of a research program

started in 1981 to find better ways of programming Telecom applica-

tions. These applications are large programs which despite careful

testing will probably contain many errors when the program is put into

service. We assume that such programs do contain errors, and investigate

methods for building reliable systems despite such errors.

The research has resulted in the development of a new programming

language (called Erlang), together with a design methodology, and set of

libraries for building robust systems (called OTP). At the time of writing

the technology described here is used in a number of major Ericsson, and

Nortel products. A number of small companies have also been formed

which exploit the technology.

The central problem addressed by this thesis is the problem of con-

structing reliable systems from programs which may themselves contain

errors. Constructing such systems imposes a number of requirements on

any programming language that is to be used for the construction. I discuss

these language requirements, and show how they are satisfied by Erlang.

Problems can be solved in a programming language, or in the stan-

dard libraries which accompany the language. I argue how certain of the

requirements necessary to build a fault-tolerant system are solved in the

language, and others are solved in the standard libraries. Together these

form a basis for building fault-tolerant sodware systems.

No theory is complete without proof that the ideas work in practice. To

demonstrate that these ideas work in practice I present a number of case

studies of large commercially successful products which use this technol-

ogy. At the time of writing the largest of these projects is a major Ericsson

product, having over a million lines of Erlang code. This product (the

AXD301) is thought to be one of the most reliable products ever made by

Ericsson.

Finally, I ask if the goal of finding better ways to program Telecom

applications was fulfilled—I also point to areas where I think the system

could be improved.