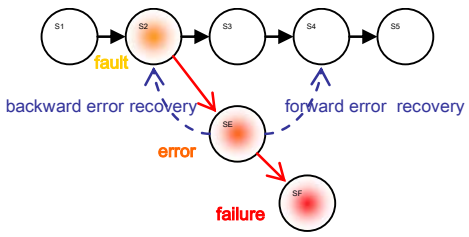
Developing a Fault Ontology Engine for the Testing and Evaluation of Service-Oriented Architectures

Introduction

This research brings together the concepts of ontology, Service-Oriented Architecture (SOA) and fault injection testing in order to develop a Fault Ontology Engine (FOE), and utilize information gathered from case studies of real-world systems.

An SOA is a type of software architecture comprising services, with emphasis on service interoperability and location transparency, and with the aim of achieving loose coupling among interacting software resources. However, SOA middleware obscures the nature of its components by design, and this can lead to systems that can be difficult to implement and evaluate.



The motivation for this research stems from the importance of computer system evaluation and the notion that the testing and evaluation of large, distributed computer systems such as SOA-based systems, and including Grid systems, could be better supported by autonomous machines.

Testing and Evaluating Systems

There are a number of ways to test and evaluate systems including simulation modelling, stress testing and observation over time.

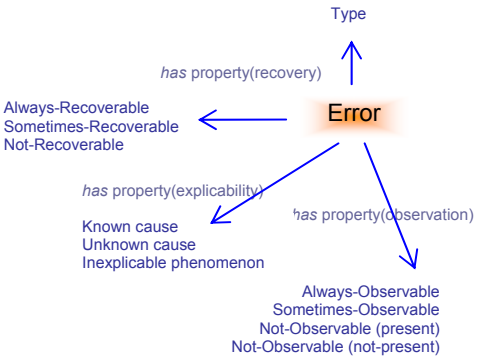
Fault injection is a method of testing both the hardware and software in systems by deliberately inserting artificial faults into systems in order to effectively reveal faults. It is a fast and logical way to detect faults and it can be used to test systems at run time and therefore is a suitable method for testing dynamic, distributed systems. Fault injection testing can also be non–invasive, using mechanisms such as interception of method calls and data corruption.

Machine Supported Evaluation

Systems are modelled in order to bridge the gap between real world problems and virtual solutions: portrayed as single or multiple elements of existence existing within a defined boundary, systems can be recursively defined through different levels of abstraction.

This research aims to facilitate the testing and evaluation of systems through novel ways of their modelling, by using ontologies to describe their domains and building machines capable of understanding the software and hardware entities systems contain. Evaluation and testing also have important roles to play in demonstrating how systems can affect their own environment and the natural world, and this research postulates modelling the environment as a single domain.

By definition, autonomous software machines are self-managing and are able to adapt to changing conditions, which means they can provide good support for the evaluation of large or complex distributed systems such as SOA systems and Grid systems, as these types of systems are generally dynamically modified during tasks. However, currently system testing is still mainly in the domain of human expertise,



experience and intuition and autonomous machines would require information that is formal, explicit, and in languages they are able interpret in order for them to acquire the intelligence necessary to make informed decisions during the process of testing and evaluation.

Ontologies are proven intelligent communication media for machines and are therefore appropriate means for describing the fault, error and failure domains of systems. They can be used to enable machines to acquire the kind of knowledge necessary to develop their own strategies for testing and evaluating unfamiliar or large, distributed systems, and by pairing together faults with resultant failures enable useful inferences to be drawn on the relationships that exist between entities within systems.

Current Status and Future Work

Currently, experimental work involves case studies that generate information for modelling the fault, error and failure domains of systems and the way entities existing in these domains may be linked or paired together. An active fault, or combination of active faults, can produce an error, or combination of errors, whose visibility outside a system boundary will be manifested as a failure. This is a well-established, recursive way to describe the relationship between failures, errors and faults. In distributed computer systems failures can occur due to internal and external attacks, arbitrary resource reallocation, resource failures, transmission failures and external and environmental phenomena.

Acquiring knowledge to develop new strategies for system testing and evaluation, by pairing together faults with resultant failures to establish relationships in fault, error and failure domains

In experiments information can be gleaned from real-world case study systems and used in a Fault-Ontology engine. In consequence this can be used in future to design and support new testing and evaluation mechanisms for large, distributed systems, including producing benchmarks where systems are previously partly or fully unevaluated.

Summary

This research therefore focuses on the development and use of ontologies of the fault and failure domains of systems, importantly including the novel notion of pairing together faults with resultant failures and making useful inferences, and considering the environment as an atomic domain. This research uses software fault injection as a mechanism for generating the data used in the development of the system ontologies

By gathering together information about the fault, error and failure domains of familiar systems into the FOE, in future it is hoped to exploit the information it contains to guide and discover novel testing and evaluation methods for complex systems such as SOA systems and Grid systems.

Further Information

Contact: Binka Gwynne
email: binka@comp.leeds.ac.uk