Automated Runtime Software Repair

Improving resilience in presence of errors

Benoit Cornu, Martin Monperrus, Lionel Seinturier

Context

The server encountered an internal error ...

> Observation: many applications crash due to unexpected exceptions.
> Goal: catch those exceptions in an automated manner to prevent them to crash the application.
> Techniques: code transformation, empirical analysis, code synthesis, data-mining.

Thesis Overview

Exception Localization Or Injection

Legacy Mode

Runtime Exception Analysis Framework

Resilient Catch Taxonomy

Patch Synthesis

Patch Selection

Patch Injection

Resilience Mode

Runtime Exception Analysis

Applicative Goals
- analyze the exception behavior specification
- isolate the catches needed to perform an action
- isolate the catches which are not(or cannot be) used

Technical Goals
- detection of exception sources and their context
- capture of catch block execution and their interplay
- analyze the relationship between causes and treatments

The Spoon library is used to instrument source code

```java
try {
    Framework.learn("try-start");
    //developer code
    Framework.learn("try-end");
} catch (RuntimeException re) {
    Framework.learn("catch-start", re);
    //developer code
    Framework.learn("catch-end");
}
```

Ongoing Work

Provide the cause of the error at runtime.
- paper submitted at ISSTA 2015.

Propose a solution to the errors.
- generating patches using SMT.

Provide an alternative solution at runtime
- is it possible to perform other operations To avoid the crash of the software?

Results

Exception handling analysis and transformation using fault injection: Study of resilience against unanticipated exceptions: (IST 2015)

- 10 applications under study
- 767 resilience mechanism existing
- 241 resilience mechanism executed (and studied)
- 92 of them present resilience properties
- 84 out of the 92 have been “stretch” to improve the resilience of the application