

BCT: A Toolset for Automated Failure Analysis

Leonardo Mariani

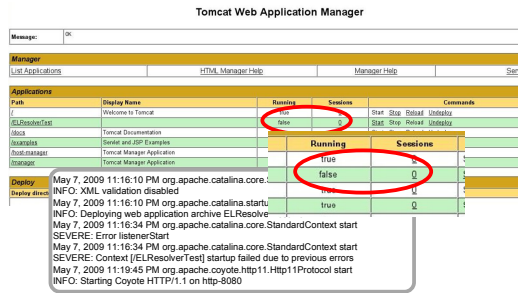
Fabrizio Pastore

Mauro Pezzè

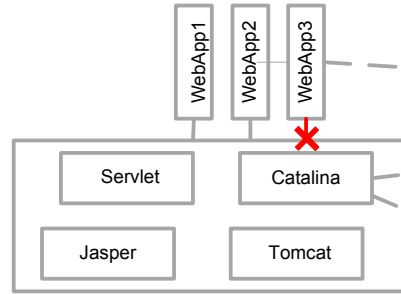
University of Milano Bicocca

University of Lugano

Failures in component based systems



A failure in Tomcat 6.0.4



Locating fault is difficult when failure is far from faulty interactions

Faulty components interactions

```

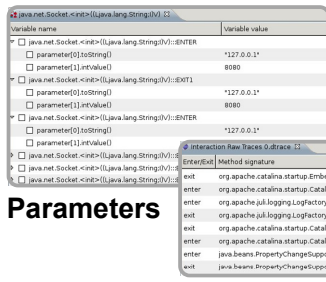
public class ChipsListener implements ServletContext.Listener {
    public void contextInitialized(ServletContextEvent ev) {
        JspFactory factory = JspFactory.getDefaultFactory();
        ServletContext context = ev.getServletContext();
        JspApplicationContext jspContext = factory.getJspApplicationContext(context);
    }
}

public final class JspRuntimeContext {
    static {
        JspFactoryImpl factory = new JspFactoryImpl();
        SecurityClassLoader securityClassLoader = factory.getClassLoader();
        JspFactory.setDefaultFactory(factory);
    }
}

public class JasperListener implements Lifecycle.Listener {
    public void lifecycleEvent(LifecycleEvent event) {
        ...
        this.getClass().getClassLoader().loadClass(
            "org.apache.jasper.compiler.JspRuntimeContext");
    }
}
    
```

BCT locates faults in 3 phases

Phase 1: Build behavioral models



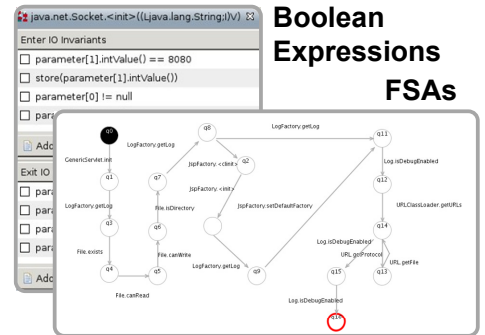
Parameters

Call sequences

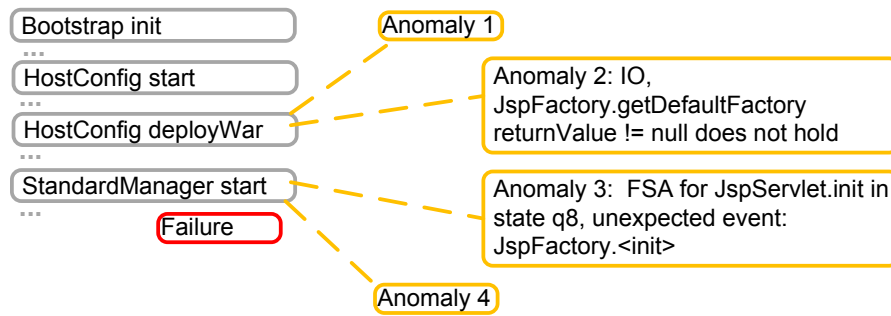
From correct execution traces

BCT Inference Engine

To models of correct behavior

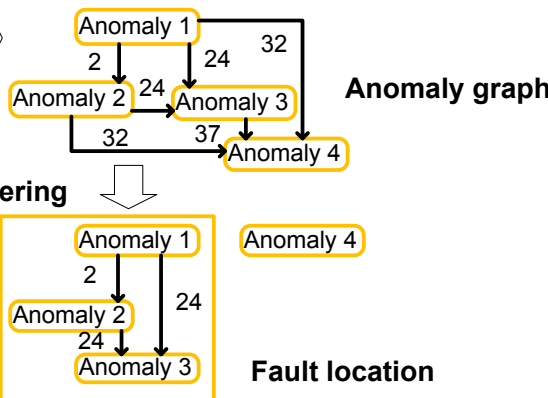


Phase 2: Trace faulty executions



Phase 3: Locate faults

From dynamic call tree to anomaly graph for fault localization



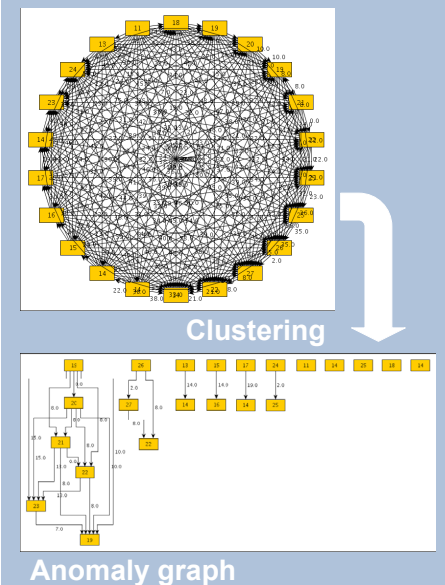
Dynamic call tree

Fault location

Case studies

Case study	ID	Connected Components	Inspected CC (false positives)
Eclipse 3.3	BugId: 181288	11	2(0)
Eclipse probekit chmod issue	ForumId 307527	1	1(0)
Eclipse probekit EMT64 issue	BugId 157486	1	1(0)
TID Orbj	BugId 413	1	1(0)
Tomcat 5.5.13	BugId 41939	2	1(0)
Tomcat 6.0.4	BugId 40820	2	1(0)

Eclipse 3.3 analysis



Anomaly graph