Speculative Analysis: Exploring Future States of Software

Yuriy Brun ♦️ Reid Holmes 🍁
Michael D. Ernst ♦️ David Notkin ♦️

♦️ University of Washington 🍁University of Waterloo
Have you ever made a mistake while programming and only realized it later?

- design decision
- refactoring
- repeated someone else’s work
Speculative analysis: Predict the future and analyze it
Speculative analysis: Predict the future and analyze it
Speculative analysis: Predict the future and analyze it
Speculative analysis: Predict the future and analyze it
Speculative analysis: Predict the future and analyze it

current program
speculate
analyze

speculative analysis

speculate
Speculative analysis: Predict the future and analyze it

speculate

current program

inform developer

speculative analysis

analyze
Speculative Quick Fix

```java
public class QFSTest {
    private String name;
    public void clearName() {
        name = null;
    }
    public void setName(String arg) {
        name = arg;
    }
    public String returnName() {
        return name;
    }
    public void useName() {
        String temp = name;
    }
    public void passName() {
        testMethod(name);
    }
    public void testMethod(String arg) {}
}
```
Speculative Quick Fix

```java
public class QFSTest {
    private String name;
    public QFSTest(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public String setName(String name) {
        this.name = name;
        return name;
    }
    public void printName(String name) {
        System.out.println(name);
    }
}
```

- Create class 'Str'
- Create interface 'Str'
- Change to 'String' (java.lang)
- Change to 'STRING' (javax.print.DocFlavor)
- Change to 'Stroke' (java.awt)
- Change to 'Struct' (java.sql)
- Change to 'Stub' (javax.rmi.CORBA)
- Create enum 'Str'
- Add type parameter 'Str' to 'QFSTest'
- Fix project setup...

Press 'Ctrl+1' to go to original position
Speculative Quick Fix

public class QFSTest {
    private String name;
    public QFSTest () {
        this.name = "test";
    }
    public void test() {
        System.out.println("Hello World! " + this.name);
    }
}

(0) Change to 'String' (java.lang)
(4) Change to 'Stub' (javax.rmi.CORBA)
(4) Change to 'Struct' (java.sql)
(4) Change to 'Stroke' (java.awt)
(4) Add type parameter 'Str' to 'QFSTest'
(6) Create class 'Str'
(6) Create interface 'Str'
(6) Create enum 'Str'
(6) Fix project setup...
(N/A) Change to 'STRING' (javax.print.DocFlavor)

Press 'Ctrl+1' to go to original position
Speculative Quick Fix

```java
public class QFSTest {
    private String name;
    public void clearName() {
        name = null;
    }
    public void setName(String arg) {
        name = arg;
    }
    public String returnName() {
        return name;
    }
}
```

QFSTest.java:4:15 (0) Change to 'String' (java.lang)
(6) Change to 'Node' (org.w3c.dom)
(6) Change to 'Naming' (java.rmi)
(6) Change to 'Name' (javax.naming)
(6) Change to 'Name' (javax.xml.soap)
(6) Change to 'Name' (javax.xml.soap)
(6) Change to 'NameList' (org.w3c.dom)
(6) Change to 'Name' (javax.lang.model.element)
(6) Create class 'name'
(6) Create interface 'name'
Contributions

- Speculative analysis
- Speculative analysis for collaborative development
  Crystal: prototype tool
- Problem space ideal for search-based solutions
The Gates conflict
The Gates conflict

M

T
The Gates conflict

The information was all there, but the developers didn't know it.
The Gates conflict

The information was all there, but the developers didn't know it.
The Gates conflict

The information was all there, but the developers didn't know it.
The Gates conflict

The information was all there, but the developers didn't know it.
The Gates conflict

The information was all there, but the developers didn’t know it.
The Gates conflict

The information was all there, but the developers didn’t know it.
The Gates conflict

The information was all there, but the developers didn't know it.
The Gates conflict

The information was all there, but the developers didn’t know it.
The Gates conflict

The information was all there, but the developers didn’t know it.
What could well-informed developers do?

- Avoid conflicts
What could well-informed developers do?

- Avoid conflicts
- Reduce conflict severity
Introducing Crystal: A proactive conflict detector

DEMO
Introducing Crystal: A proactive conflict detector

DEMO

http://crystalvc.googlecode.com
Speculative analysis in collaborative development
Reducing false positives in conflict prediction

Collaborative awareness

- Palantír [Sarma et al. 2003]
- FASTDash [Biehl et al. 2007]
- Syde [Hattori and Lanza 2010]
- CollabVS [Dewan and Hegde 2007]
- Safe-commit [Wloka et al. 2009]
- SourceTree [Streeting 2010]
Reducing false positives in conflict prediction

Collaborative awareness

- Palantír [Sarma et al. 2003]
- FASTDash [Biehl et al. 2007]
- Syde [Hattori and Lanza 2010]
- CollabVS [Dewan and Hegde 2007]
- Safe-commit [Wloka et al. 2009]
- SourceTree [Streeting 2010]

Crystal analyzes **concrete artifacts**, eliminating false positives and false negatives.
Utility of proactive collaborative conflict detection

- Are textual collaborative conflicts a real problem?
  - 16% of the merges have textual conflicts.
  - Conflicts live a mean of 9.8 days.

- How dangerous are safe merges?
  - 93% of textual conflicts developed from safe merges.
  - 20% of textually-safe merges developed into conflicts.

- Do higher-order collaborative conflicts exist?
  - One in three conflicts are of higher-order.

[Brun et al. 2011]
Crystal is in the wild

“Crystal handles several projects and users effortlessly and presents the necessary information in a simple and understandable way.”
– a user

Microsoft Beacon

- A centralized version control-based tool.
- Microsoft product groups will use Beacon to help identify conflicts earlier in the development process.
- We will conduct user studies to measure effects on developers.
Open problem:
How to search speculative analysis’ many possible futures?
Contributions

- Introduced **speculative analysis** to guide future actions.
- Developed Crystal to **precisely** detect conflicts and **unobtrusively** inform developers.
- Identified a problem space ideal for search-based solutions.

http://crystalvc.googlecode.com


