Christoph Josef Amrein

Graduate Candidate	Christoph Josef Amrein
Examiner	Prof. Dr. Luc Bläser
Co-Examiner	- -
Subject Area	Software and Systems

Detection of Concurrency Bug Patterns

Roslyn-Based Static Code Analysis for C#

```
22    public void Add(int amount) {
23      lockObject.EnterWriteLock();
24      stock += a
25      lockObject
26    }
27
28    public bool TryAdd(int amount) {
29      lockObject.EnterReadLock();
30      try {
31        if (stock + amount > capacity) {
32          return false;
33        }
34
35        Add(amount);
36        return true;
37      } finally {
38        lockObject.ExitReadLock();
39      }
40    }
```

(field) ReaderWriterLockSlim Warehouse.lockObject

'lockObject.EnterWriteLock' is nested within a read lock.

Just-In-Time Code Analysis in Visual Studio

Build Succeeded

Build 20160520.13

Ran for 40 seconds (Hosted), completed 7 da

Summary Timeline Tests

samples.cli\LowLevel.cs (16, 14)
A memory barrier could be necessary between the writing to 'samples.cli.LowLevel.a' and reading at '!b'.

samples.cli\Monitors\AbortManualMonitor.cs (21, 13)
Consider using lock(...) instead of 'System.Threading.Monitor.Enter()'.

samples.cli\Monitors\AbortManualMonitor.cs (25, 15)
Consider using lock(...) instead of

Integration of the Analysis into TFS Online

This work presents a static checker for the detection of common concurrent bug patterns in C#. A collection of bug patterns has been collected, describing specific code constellations that can lead to different concurrency-related bugs. The checker and the catalog motivate the use of numerous design principles that can help reducing the risk of concurrency bugs.

The checker uses Roslyn to perform its code analysis and consists of three components. (1) A command line interface for quick and installation-free scans of solutions. (2) A NuGet package enabling straightforward integration into continuous integration systems. (3) A Visual Studio plugin for just-in-time code analysis that provides immediate feedback to the developer.

During the experimental evaluation, the checker has been used to analyze the source code of various open source projects. The results have been manually reviewed to verify the overall quality of the analysis. This has shown that projects of different scales and maturity suffer from concurrency bugs. In other words, the evaluation confirms that the checker can be a valuable utility to reduce the chance of bugs when developing concurrent code.