Pushing the Boundaries of Testing and Continuous Integration

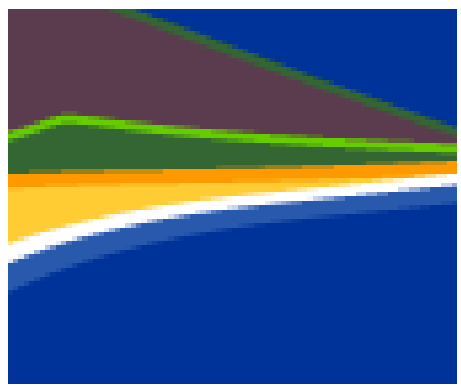
Fabrizio Cannizzo
Raghav Ramesh
Robbie Clutton

Who are we?

http://flickr.com/photos/kerrybuckley/1356791754/in/pool-web21c

http://flickr.com/photos/zoonabar/159442457/

Web21C SDK Do Less: Achieve More



ALOHA

Agile to the Core!

XP/Scrum

- TDD
- Pair programming
- Continuous Build/Integration
  - Unit/Acceptance Test
  - Reporting

http://flickr.com/photos/mattimattila/2188826561/

What's Missing?

Robustness

Performance

Scalability

Failover

http://flickr.com/photos/alan-light/2435511982/

But we didn't want to wait for formal QA testing

http://flickr.com/photos/luchilu/677786684/

http://flickr.com/photos/22243977@N06/2457787000/

What next?

Create scenarios

Define expectations

Automate

Continuously test

So... how does it **really** work?

Technologies

Cruise Control

Java Spring Framework

Hand crafted Java code

JMX Console

UltraMonkey

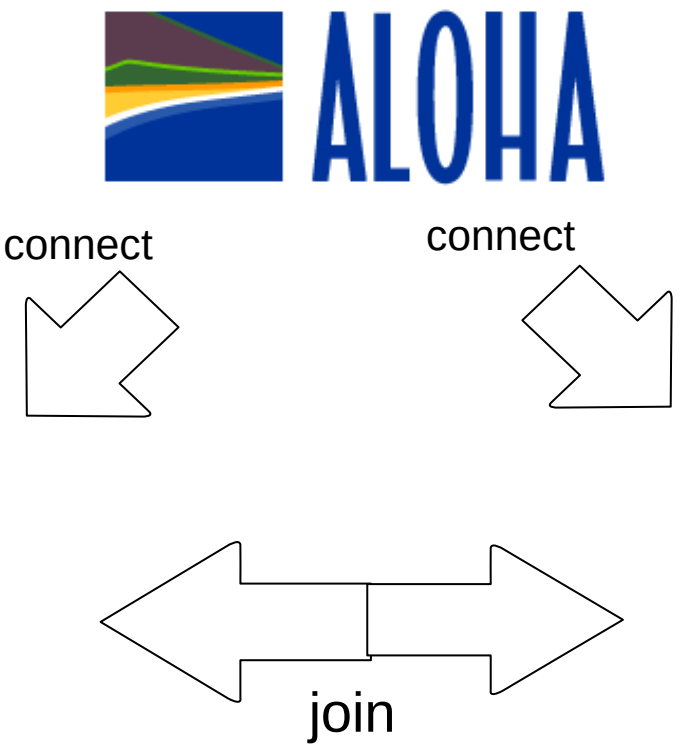
SIPp

JainSIP

Mockphones

Cruise control is the engine that continuously runs typical usage scenarios.

http://flickr.com/photos/_fxr/2142296214/



http://flickr.com/photos/mike-burns/35534789/

Ensure the entire scenario played out as expected

Why are scenarios relevant?

Robustness

Performance

Scalability

Failover

We **automated** phone behaviour using SIPp...

Different **configurations** are implemented using different spring application contexts and network setups

Scenario runners are java applications

Scenarios are POJOs loaded from spring application context

CruiseControl at 172.25.58.147 *[28/04/08 12:01]*

Project	Status *(since)*	Last failure	Last successful	Label	
callflow-robustness	**building *(12:00)***		09:31	callflow-robustness-build.185	Build
RobustnessDatabase	*waiting (08:31)*		17:29	RobustnessDatabase-build.334	Build
RobustnessPerformance	*waiting (08:31)*		17:33	RobustnessPerformance-build.242	Build
RobustnessPerformanceTerracotta	*waiting (10:48)*		10:34	RobustnessPerformanceTerracotta-build.232	Build
SipStoneTerracotta	*waiting (11:03)*		10:48	SipStoneTerracotta-build.29	Build
SipStoneDatabase	*waiting (11:20)*		11:04	SipStoneDatabase-build.55	Build
SipStoneMemory	*waiting (11:36)*		11:20	SipStoneMemory-build.46	Build
SipStoneDatabaseHA	*waiting (11:52)*		11:36	SipStoneDatabaseHA-build.17	Build
RobustnessMemory	*waiting (11:56)*		11:52	RobustnessMemory-build.381	Build
RobustnessDatabaseMultistack	*waiting (12:00)*		11:56	RobustnessDatabaseMultistack-build.185	Build

RSS

Build Results | Test Results | XML Log File | Metrics | Config | Control Panel

BUILD COMPLETE - RobustnessPerformance-build.242

Date of build: 04/27/2008 17:33:28
Time to build: 8 minutes 30 seconds
Last changed: 03/14/2008 09:57:49
Last log entry:

Build Artifacts

Errors/Warnings: (27270)

log4j:WARN No appenders could be found for logger (org.springframework.context.support.ClassPathXmlApplicationContext).

log4j:WARN Please initialize the log4j system properly.

2008-04-27 17:33:38,205 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext1() 138] - ######## Returning APPCTX - 1

2008-04-27 17:33:38,210 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext1() 138] - ######## Returning APPCTX - 1

2008-04-27 17:33:38,211 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext2() 143] - ######## Returning APPCTX - 2

2008-04-27 17:33:38,211 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext2() 143] - ######## Returning APPCTX - 2

2008-04-27 17:33:38,211 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext1() 138] - ######## Returning APPCTX - 1

2008-04-27 17:33:38,212 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext1() 138] - ######## Returning APPCTX - 1

2008-04-27 17:33:38,213 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext2() 143] - ######## Returning APPCTX - 2

2008-04-27 17:33:38,213 BatchrunSipStack[main] INFO [PerformanceBatchTest.main() 165] - Loading application context

2008-04-27 17:33:38,213 BatchrunSipStack[main] INFO [MultistackApplicationContextManager.getApplicationContext1() 138] - ######## Returning APPCTX - 1

2008-04-27 17:33:38,416 BatchrunSipStack[main] INFO [PerformanceBatchTest.logSystemInformation() 273] - ----------

2008-04-27 17:33:38,418 BatchrunSipStack[main] INFO [PerformanceBatchTest.logSystemInformation() 274] - OS Name: Linux, version: i386, architecture: 2.6.5-7.202.7-smp

Java version: 1.5.0_13, vendor: Sun Microsystems Inc.

Database driver: org.postgresql.Driver, url: jdbc:postgresql://localhost:5432/springring

Collections Information - Dialogs: com.bt.sdk.callcontrol.sip.dialog.collections.DialogCollectionImpl, Calls: com.bt.sdk.callcontrol.sip.call.collections.PersistedCallCollectionImpl

Jain Sip Jars: jain-sip-api-1.2.jar, jain-sip-ri-1.2.jar

Jain Sip Thread Count: 200, Simple Sip Stack Thread Count: 200, Database Pool Thread Count: 20

Number of scenarios to be run: 30, number of application threads: 16

2008-04-27 17:33:38,418 BatchrunSipStack[main] INFO [PerformanceBatchTest.logSystemInformation() 275] - ----------

2008-04-27 17:33:38,419 BatchrunSipStack[main] INFO [PerformanceBatchTest.main() 186] - Running tests with 16 concurrent threads

2008-04-27 17:33:38,423 BatchrunSipStack[pool-3-thread-1] INFO [BatchTestScenarioBase.start() 144] - Starting scenario createCallTerminateCallScenario:61e7139b8caa297688ef75c4ed094b90, createCallTerminateCallScenario

2008-04-27 17:33:38,424 BatchrunSipStack[pool-3-thread-3] INFO [BatchTestScenarioBase.start() 144] - Starting scenario createCallTerminateCallScenario:487134addc7778cdc15884d5e3b10f40, createCallTerminateCallScenario

2008-04-27 17:33:38,425 BatchrunSipStack[pool-3-thread-10] INFO [BatchTestScenarioBase.start() 144] - Starting scenario createCallTerminateCallScenario:f7615b48e084b1232efe16cd0814aca8, createCallTerminateCallScenari

2008-04-27 17:33:38,424 BatchrunSipStack[pool-3-thread-9] INFO [BatchTestScenarioBase.start() 144] - Starting scenario createCallTerminateCallScenario:7370eea39ae9c01ee5adc25e7d3cd18a, createCallTerminateCallScenario

2008-04-27 17:33:38,424 BatchrunSipStack[pool-3-thread-8] INFO [BatchTestScenarioBase.start() 144] - Starting scenario createCallTerminateCallScenario:8606b2e7dd321928c6edfadd52ead702, createCallTerminateCallScenario

Scenario createCallTerminateCallScenario:52ece01401be2d5b5e3041df88d07a73 succeeded

Scenario twoCallsSharingCallLegScenario:899e3c0cf0912b1c3d82692c854766db succeeded

Scenario callAnswerTimeoutScenario:7d1fbe093379f994ab67cdb4787347bd succeeded

Scenario createCallTerminateCallScenario:749ba899e03fac5f1f555ded00df529d succeeded

Scenario badAddressScenario:35388daa7ee83701650dcb8c16e7f94f succeeded

Scenario maxCallDurationScenario:068339c3b277ece6371569abd657dc5c succeeded

Scenario maxCallDurationScenario:3fcfb593e73708903417c8305e3216ab succeeded

Scenario callAnswerTimeoutScenario:f1d1b820cbcbcc8e3d53cd131f8a4ab9 succeeded

Scenario badAddressScenario:d22383187a43ab3e117298db58b1e07d succeeded

Scenario badAddressScenario:d48c3b7f31633f763752c54ebd6eb260 succeeded

Scenario maxCallDurationScenario:59a9ef66c464559e1819b97b05084123 succeeded

Scenario maxCallDurationScenario:a1b757ce5fceb68417ca5cac3ebd55c9 succeeded

Scenario createCallTerminateCallScenario:59ca1274ffc1f9f3b707a2684ac2b259 succeeded

Scenario twoCallsSharingCallLegScenario:17a6b95ae6c5693aa634e02bf906955c succeeded

500 tests run

Average delay between scenarios: 1202.360000ms

announcementCallScenario: 14/14 (100.0%)

badAddressScenario: 91/91 (100.0%)

callAnswerTimeoutScenario: 95/95 (100.0%)

createCallTerminateCallScenario: 104/104 (100.0%)

firstCallLegFailureScenario: 0/0 (0.0%)

maxCallDurationScenario: 119/119 (100.0%)

secondCallLegFailureScenario: 0/0 (0.0%)

twoCallsSharingCallLegScenario: 77/77 (100.0%)

Overall success rate: 100.0%

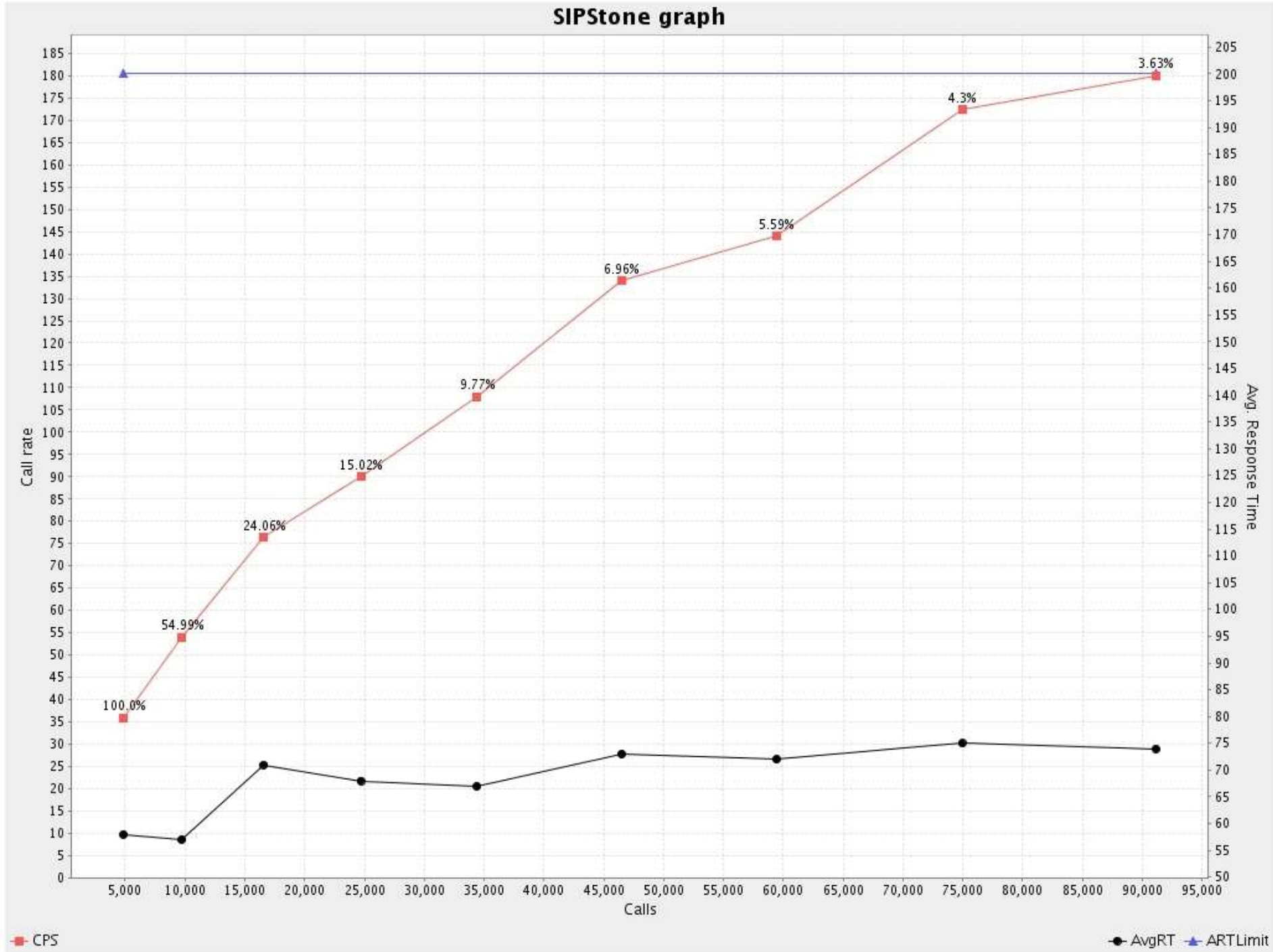
ReplaceInfo invocations at dialog level: 5410, at call level: 2460, at conference level: 0, combined: 7870

Concurrent Exceptions at dialog level: 12, at call level: 19, at conference level: 0, combined: 31

Unit Tests: (0)

No Tests Run

This project doesn't have any tests


SIPStone graph
Call rate
Avg. Response Time
Calls
100.0%
54.99%
24.06%
15.02%
9.77%
6.96%
5.59%
4.3%
3.63%
CPS
AvgRT
ARTLimit

TestRR

- Generic Framework
  - Robustness
  - Performance

- http://code.google.com/p/testrr/

Who else is talking about this?

James Bull - Pragmatic Performance Testing - http://is.gd/SUv

Dr. Dobbs - Continuous Integration and Performance Testing - http://is.gd/12Ds

Manfred Lange - Introducing Automated Performance Testing - http://is.gd/SUj

Paul King & Dalton Cranston – Open Source Performance Testing Tools For The Web

What we have learned that you could take away...

Automate **early** and **often**

unit and **acceptance** tests are an essential first step

maintaining these builds can have severe impact on **velocity**

the **customer** must buy in: formulate user stories to drive the creation of tests

... and also...

This drives creation of **robust code** earlier

Leverage existing **frameworks** for performance testing

Build up testing strategies **incrementally**

These tests are resource hungry, use an **appropriate environment**

Q & A

Fabrizio Cannizzo:
- http://smartrics.blogspot.com

Raghav Ramesh:
- http://ragstorooks.wordpress.com

Robbie Clutton:
- http://blog.iclutton.com

Aloha SIP A/S: https://trac.osmosoft.com/Aloha

Web21C SDK: http://web21c.bt.com