



Automated Testing of VoIP Infrastructure

Lessons from the field



About me

- MSc. in Computer Science
- Bugs are interesting





About me

- Software tester at Vcare connect
 - (Tele-)communications for healthcare
 - Based in Enschede, the Netherlands
- We have our own VoIP infrastructure
- Software correctness is crucial

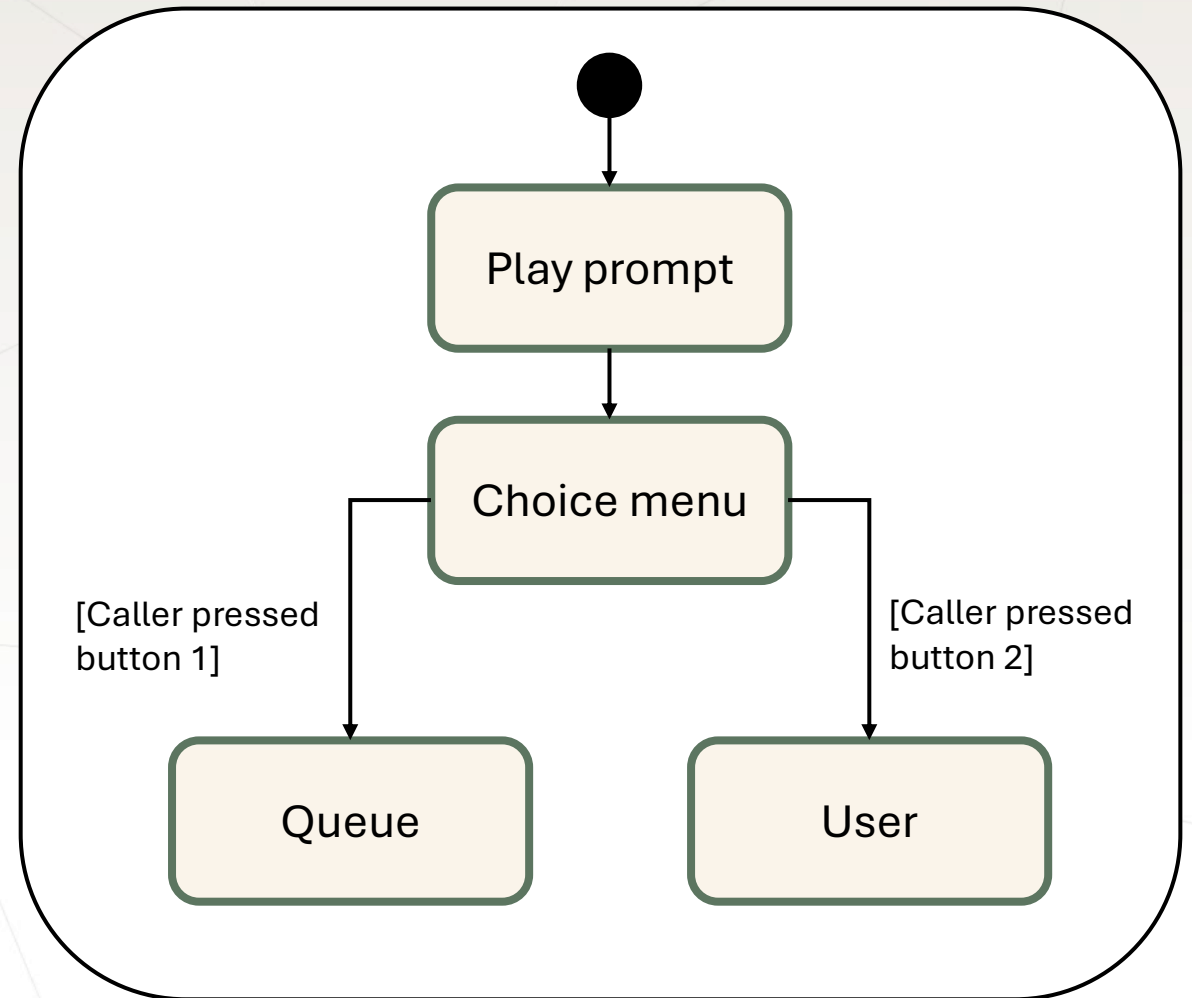




System under test

Functionality

- Call routing
- Dial plans
 - State machine
 - Route call to user
 - Queues
 - Play prompts
 - Choice menus
 - Conditional branching

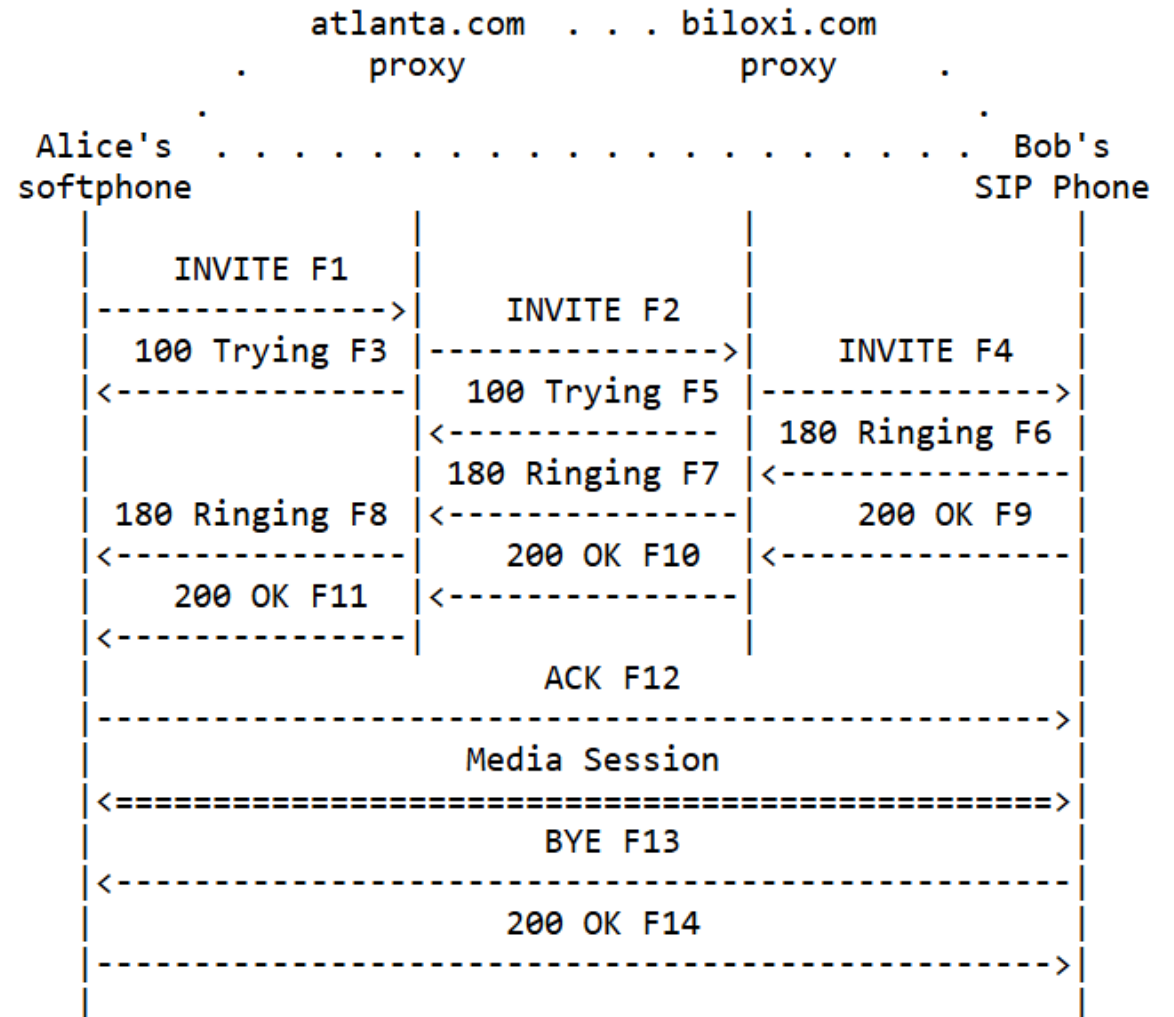




System under test

Session Initiation Protocol (SIP)

- Based on HTTP
- Messaging back-and-forth to initiate a call

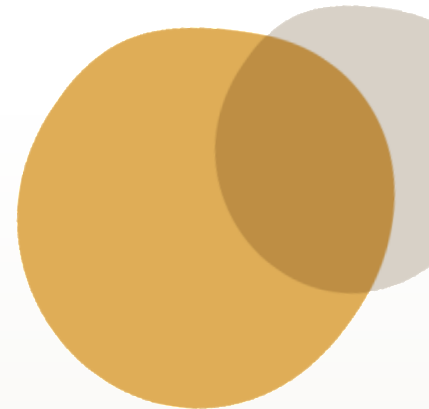


Source: SIP RFC 3261 (<https://www.rfc-editor.org/rfc/rfc3261.html>)



Basic test setup

- Written in .NET
- ~1000 system / integration tests for VoIP calling
- Test pattern
 - Arrange
 - Act
 - Assert

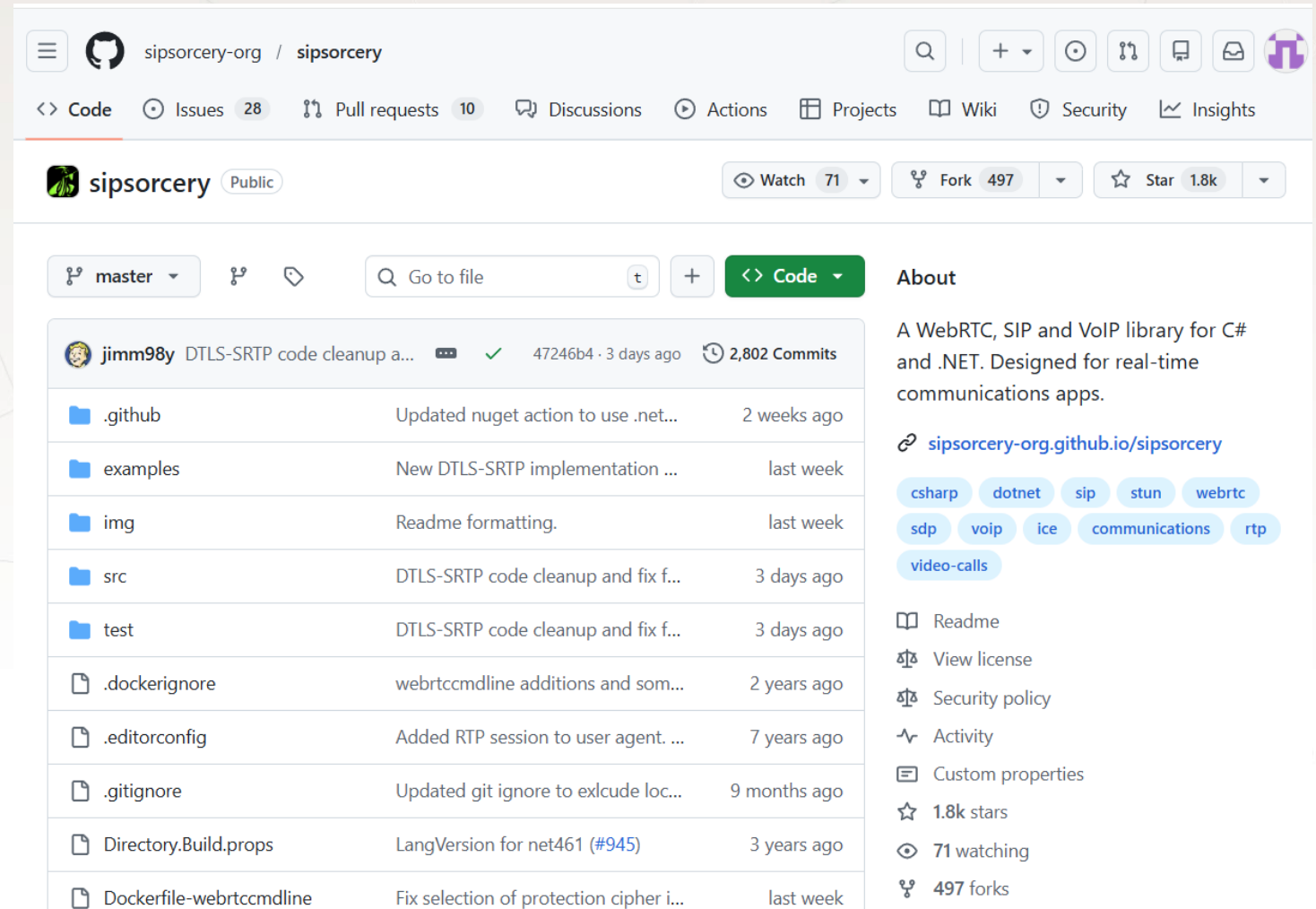




Test framework

SIPSorcery

- Use SIPSorcery to make VoIP calls
- Written in .NET
- Open-source
- Customizable out-of-the-box
- Active development



The screenshot shows the GitHub repository for SIPSorcery. The repository is owned by sipsorcery-org and is public. It has 28 issues, 10 pull requests, and 1.8k stars. The repository is described as a WebRTC, SIP and VoIP library for C# and .NET. The file list shows various folders and files, including .github, examples, img, src, test, .dockerignore, .editorconfig, .gitignore, Directory.Build.props, and Dockerfile-webrtcmdline. The repository is actively maintained, with the last commit being a DTLS-SRTP code cleanup and fix f... by jimm98y, 3 days ago.

File/Folder	Description	Last Commit
.github	Updated nuget action to use .net...	2 weeks ago
examples	New DTLS-SRTP implementation ...	last week
img	Readme formatting.	last week
src	DTLS-SRTP code cleanup and fix f...	3 days ago
test	DTLS-SRTP code cleanup and fix f...	3 days ago
.dockerignore	webrtcmdline additions and som...	2 years ago
.editorconfig	Added RTP session to user agent. ...	7 years ago
.gitignore	Updated git ignore to exlcude loc...	9 months ago
Directory.Build.props	LangVersion for net461 (#945)	3 years ago
Dockerfile-webrtcmdline	Fix selection of protection cipher i...	last week

Source: SIPSorcery GitHub (<https://github.com/sipsorcery-org/sipsorcery>)



```
// Create a SIP user agent and media session
var userAgent = new SIPUserAgent(new SIPTransport(), null);
var voipMediaSession = new VoIPMediaSession();

// Start a call to DESTINATION
var callResponse = await userAgent.Call(DESTINATION, null, null, voipMediaSession);

// Send DTMF tone '3'
await userAgent.SendDtmf(0x03);

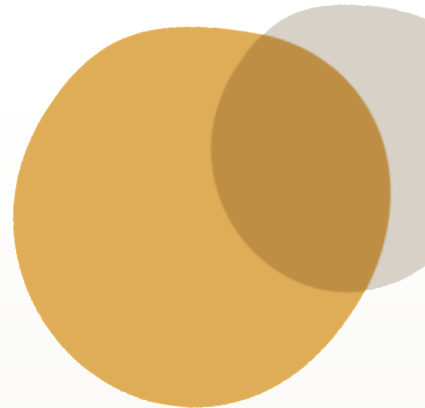
// Hang up the call
userAgent.Hangup();
```



Challenges

Challenges

- Realistic phone behaviour
- Audio correctness
- Time-sensitive testing

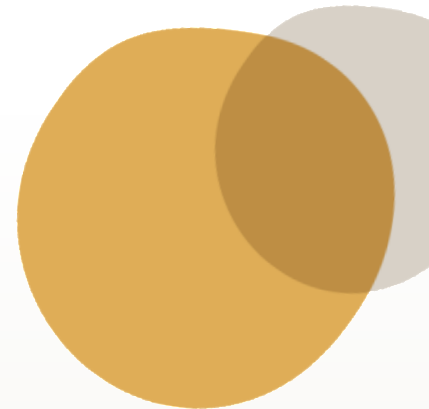




Challenges

Realistic phone behaviour

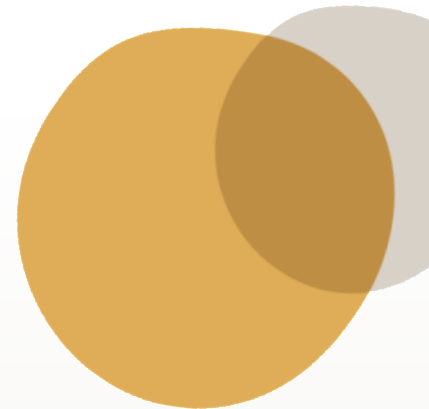
- Permanently plugged into the wall
 - Reuse same SIPSorcery phones across tests
 - Wipe history and ignore call messages from previous test
- Custom header overrides
 - Possible as SIPSorcery is customizable and open-source
- Unique bugs found due to differences
 - Sending DTMF too fast





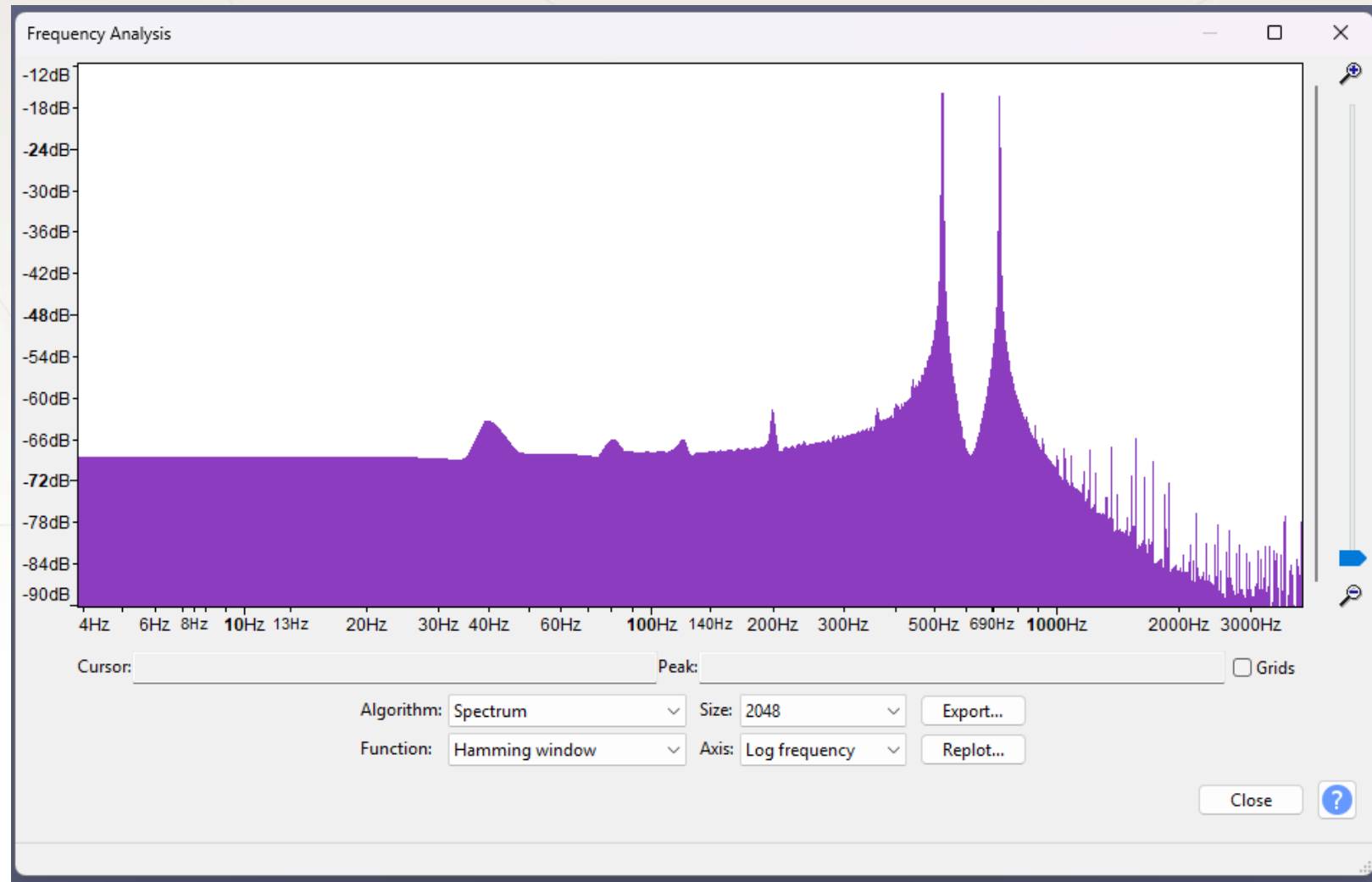
Audio correctness

- Each phone plays a tone in a different frequency
 - Built-in in SIPSorcery, but had to extent it slightly
- Store every audio sample SIPSorcery receives
- Asserting correctness
 - Frequency analysis
 - Fast Fourier Transform with Hamming window
 - Restrictions





Challenges



Source: Screenshot from Audacity's frequency analysis tool (<https://github.com/audacity/audacity>)



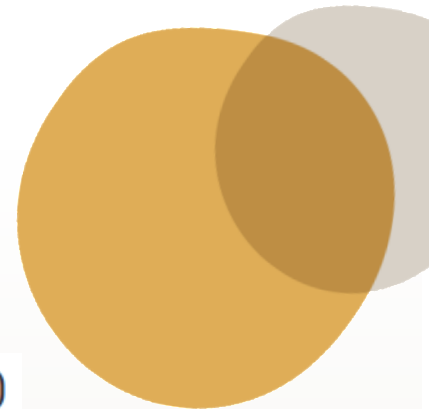
Challenges

Time-sensitive testing

- Phone actions are not instant
- But you can subscribe to call events

```
// Subscribe to the incoming call event  
userAgent.OnIncomingCall += IncomingCallHandler;
```

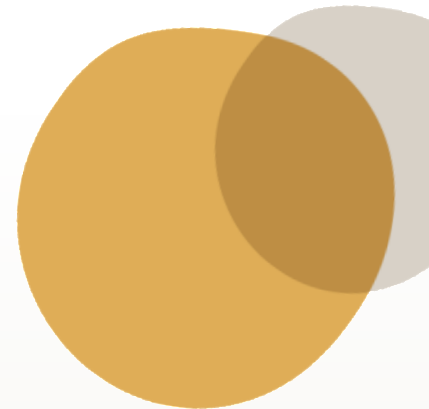
```
void IncomingCallHandler(SIPUserAgent userAgent, SIPRequest request)  
{  
    // Handle incoming call  
}
```





Time-sensitive testing

- Wait till a certain event happened
 - Timeout hit → test invalid
 - Event found → continue test
- Timeout lengths
- Timeout multiplier for slower target machines





Summary

- Using SIPSorcery to test a VoIP backend
- Challenges
 - Realistic phone behaviour
 - Audio correctness
 - Time-sensitive testing

