GDC 12

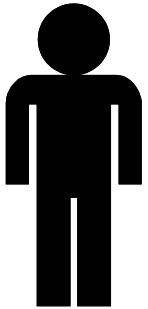
Automated Audio Testing

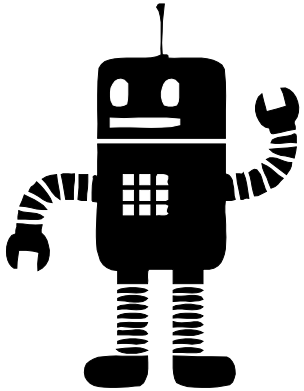
Bernard Rodrigue
Software Developer, Audiokinetic

GAME DEVELOPERS CONFERENCE
SAN FRANCISCO, CA
MARCH 5-9, 2012
EXPO DATES: MARCH 7-9
2012

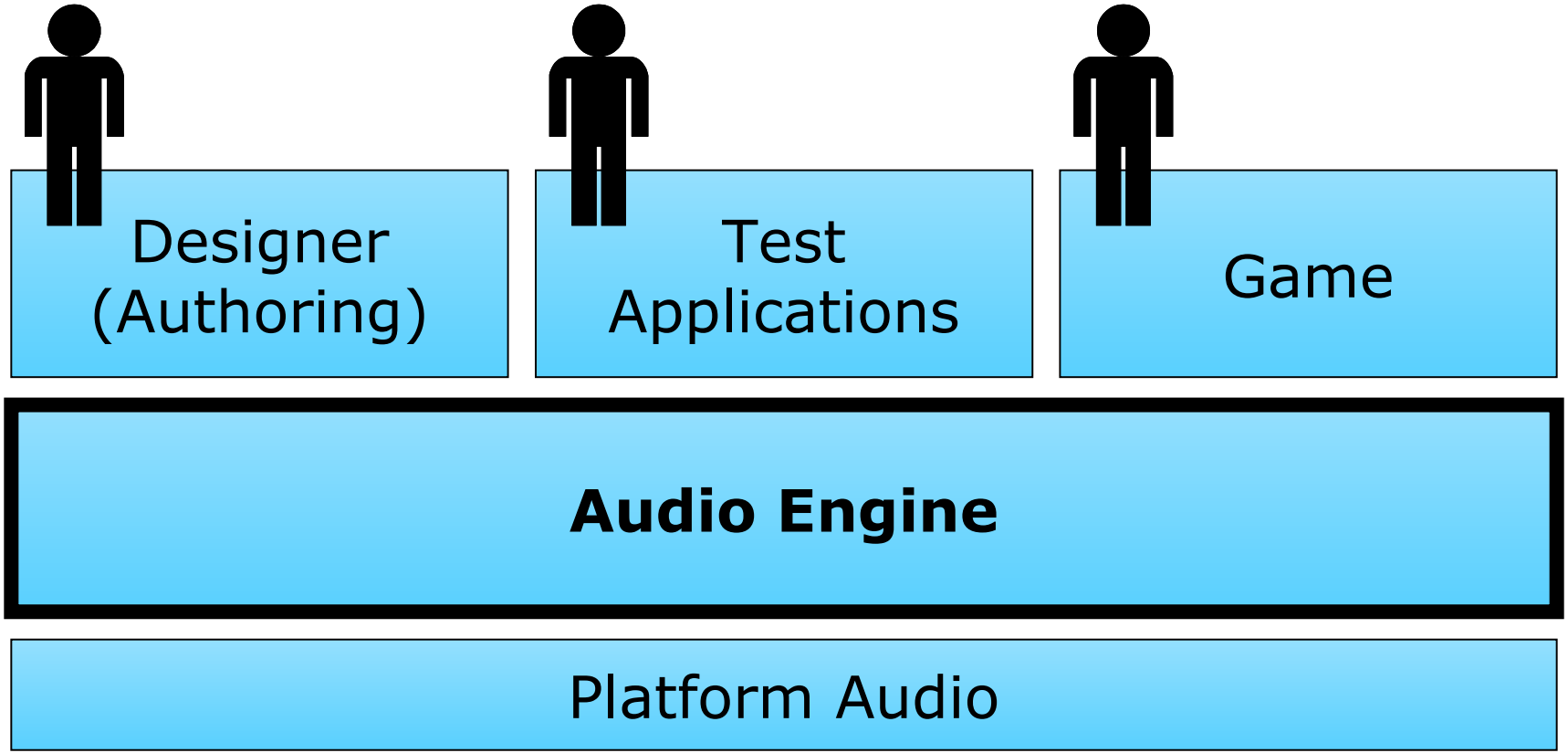
Agenda

- History of Audio Testing
- Optimizing human interventions
- RoboQA in details
- Other forms of testing
- Your game and tools



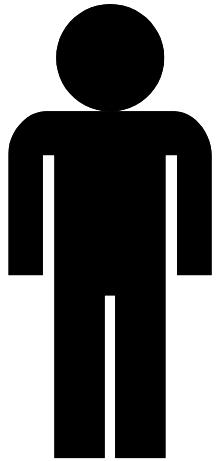


History


Designer
(Authoring)
Test
Applications
Game
Audio Engine
Platform Audio

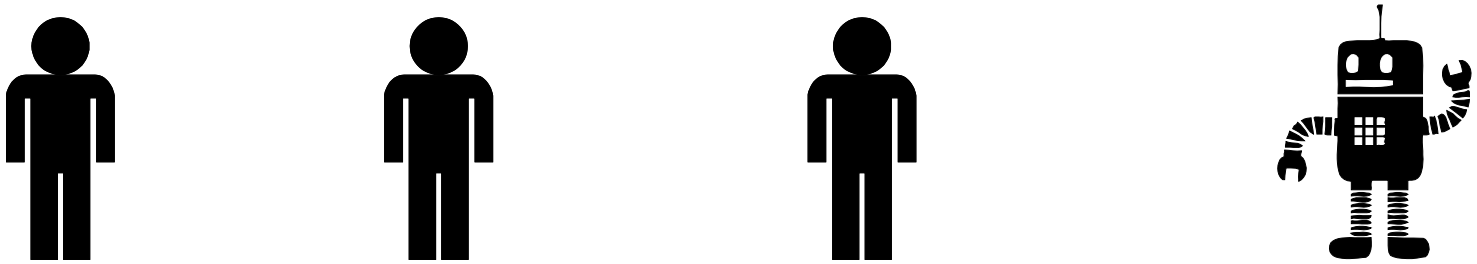
Manual Testing

- Audio Engine code needs to be tested
- Interactive testing
  - In game
  - In test applications
  - In designer tools
- Human ears

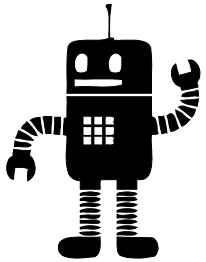


Issues with this Strategy

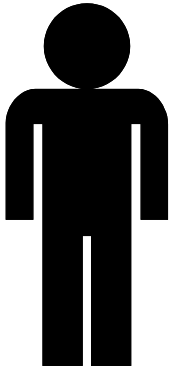
- Hard to structure
- Depends on human ears
- Repetitive and error prone
- Very low test coverage
- No quality metrics


Designer (Wwise)
Test Applications
Game
Script Engine
Audio Engine
Platform Audio

Defining Test Scenarios



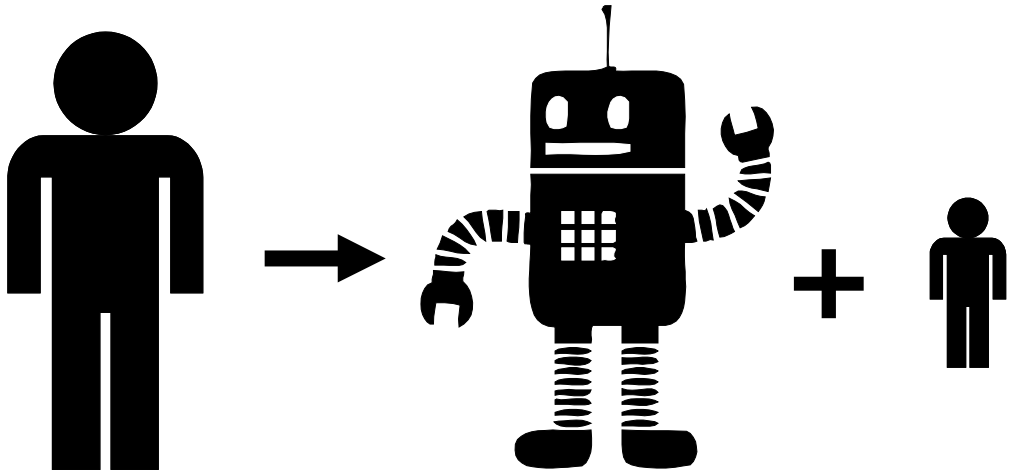
- Scripts library
- First quality metrics
- Listen to the script output
  - Have expectations of what to hear
  - Approve the audio or not
- Run on a many platforms



Issues with this Strategy

- Depends on human ears
- Can not detect subtle issues
- Long and repetitive process (weeks!)
- Easy to get distracted and overlook issues
- The tests are not run often enough
- The tester becomes insane

A New Era



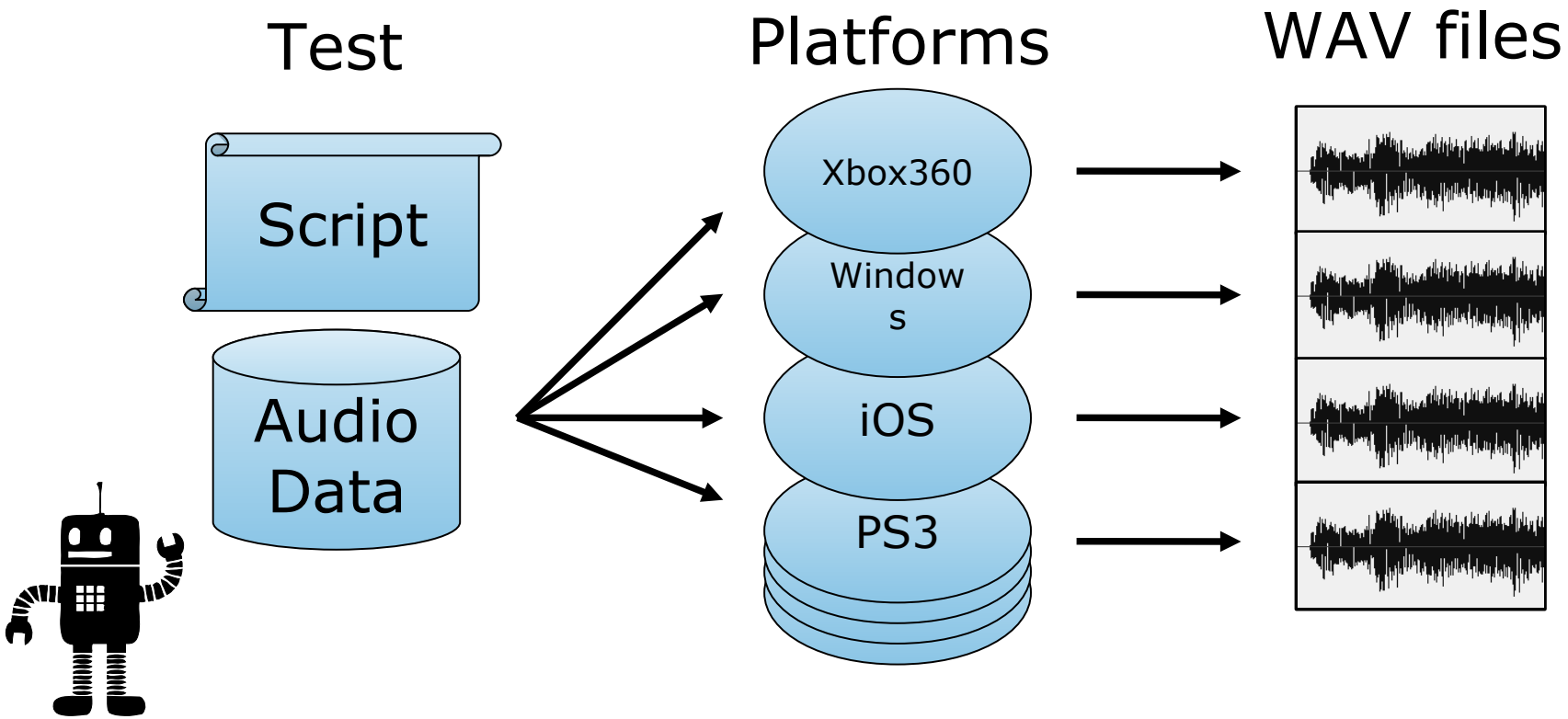
Audio Testing Goals

- Minimize dependency on human ears
- Minimize human interventions
- Catch regression issues early
- Increase test coverage
- Maintain high quality builds

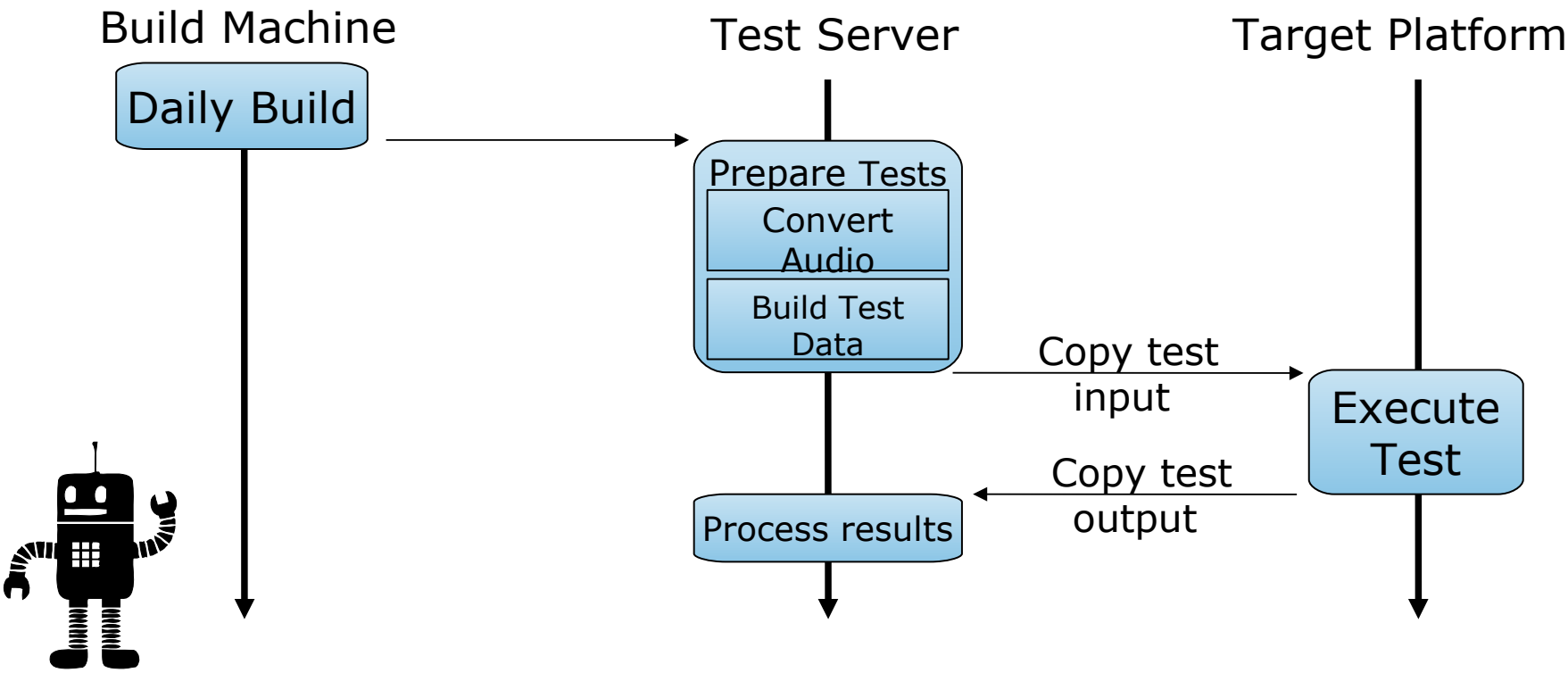
Key Elements

- Solid script library
- **Record** and **compare** the audio output
- Process the audio in offline vs. real-time
  - Record 1 minute in 5 seconds
- Multiple platforms/configurations
- Daily

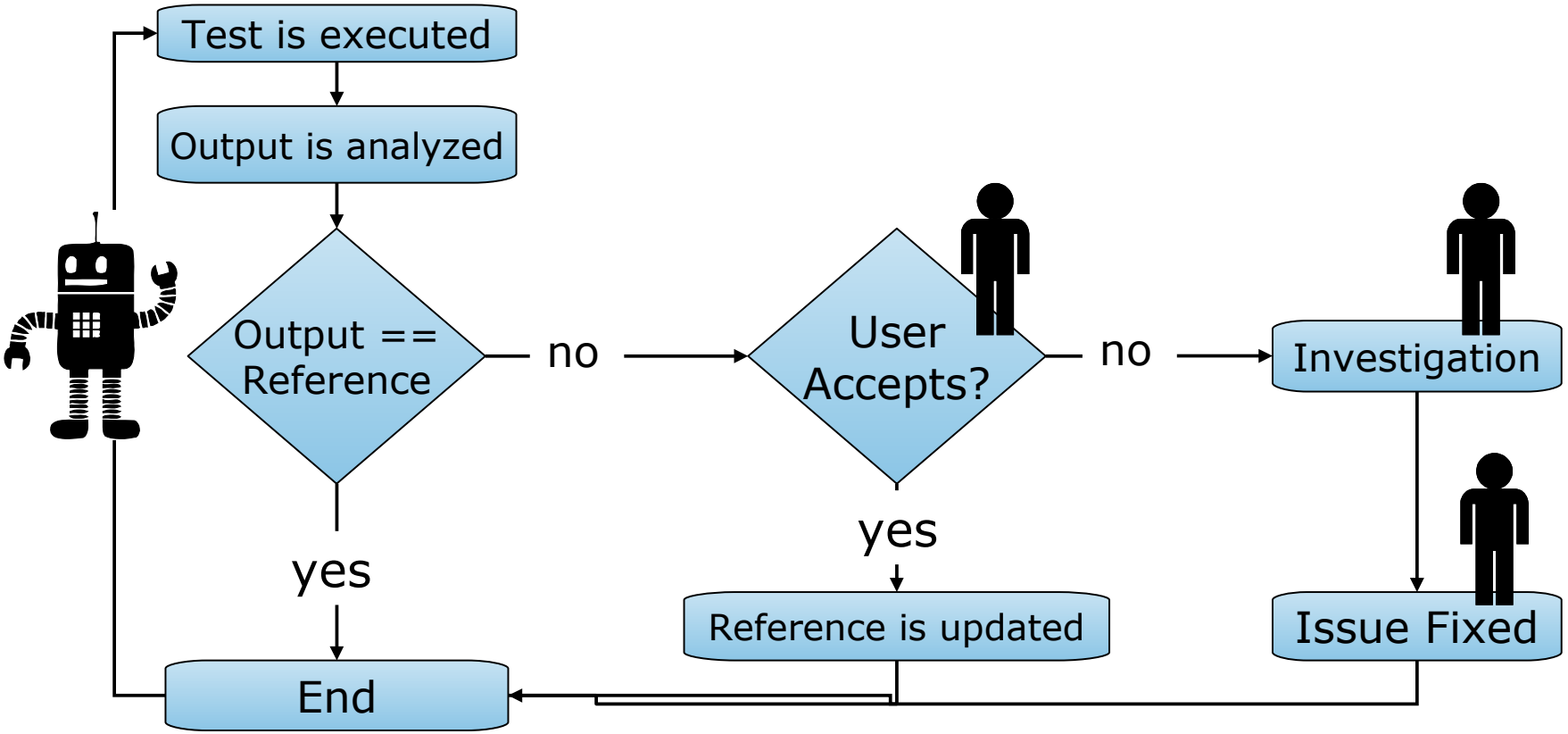
The Automated System



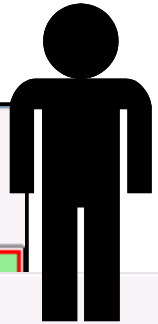
Daily Sequence



Evaluating the Test Results



Report - Front End



Script/Test	x86 vc90 Dbg 2.0	x86 vc90 Dbg 5.1	x86 vc90 Prof 2.0	x86 vc90 Prof 5.1	x64 vc100 Dbg 2.0	x64 vc100 Dbg 5.1	x64 vc100 Prof 2.0	x64 vc100 Prof 5.1	XBox vc90 Dbg 2.0	XBox vc90 Dbg 5.1	XBox vc90 Prof 2.0	XBox vc90 Prof 5.1	PS3 Dbg 2.0
AllActionBehavior-ParamTest.lua													
CoTest80_ Reset Pitch All Action Random													
CoTest98_ Stop Action Actor-Mixer													
BusHierarchy.lua													
EmbeddedContainers.lua													
AkPlayEventUntilDone_Play_Case_14_XM													
3DGameDefinedTest.lua													
Test 2- 3D Game-Defined- Spatialization													
Test 3- 3D Game-Defined- Spatialization													
Play mode Test.lua													
Multichannel virtual voices.lua													
PlaySurroundVirtual													
Vorbis_QualityControlTest_GenerateRefe													
AkLoadBankCoRoutine	0dB		0dB										
Vorbis01_Q_01_castanets-OggWwiseRef													
Vorbis02_Q_01_glock-OggWwiseRef													
Vorbis03_Q_01_Krall-OggWwiseRef													

- Compare
- Compare All Platforms
- Compare Selected
- Compare with master ref
- Open Script
- Open Project
- SVN log for script
- SVN log for project
- SVN log for source code
- Open Reference Folder
- Open Test Folder
- Open Project Folder
- Accept Changes
- Cancel Task
- Restart Task

Also in the report

- Error return values
- Crashes
- Asserts
- Execution issues
- Network and System failures

Focusing on Important Details

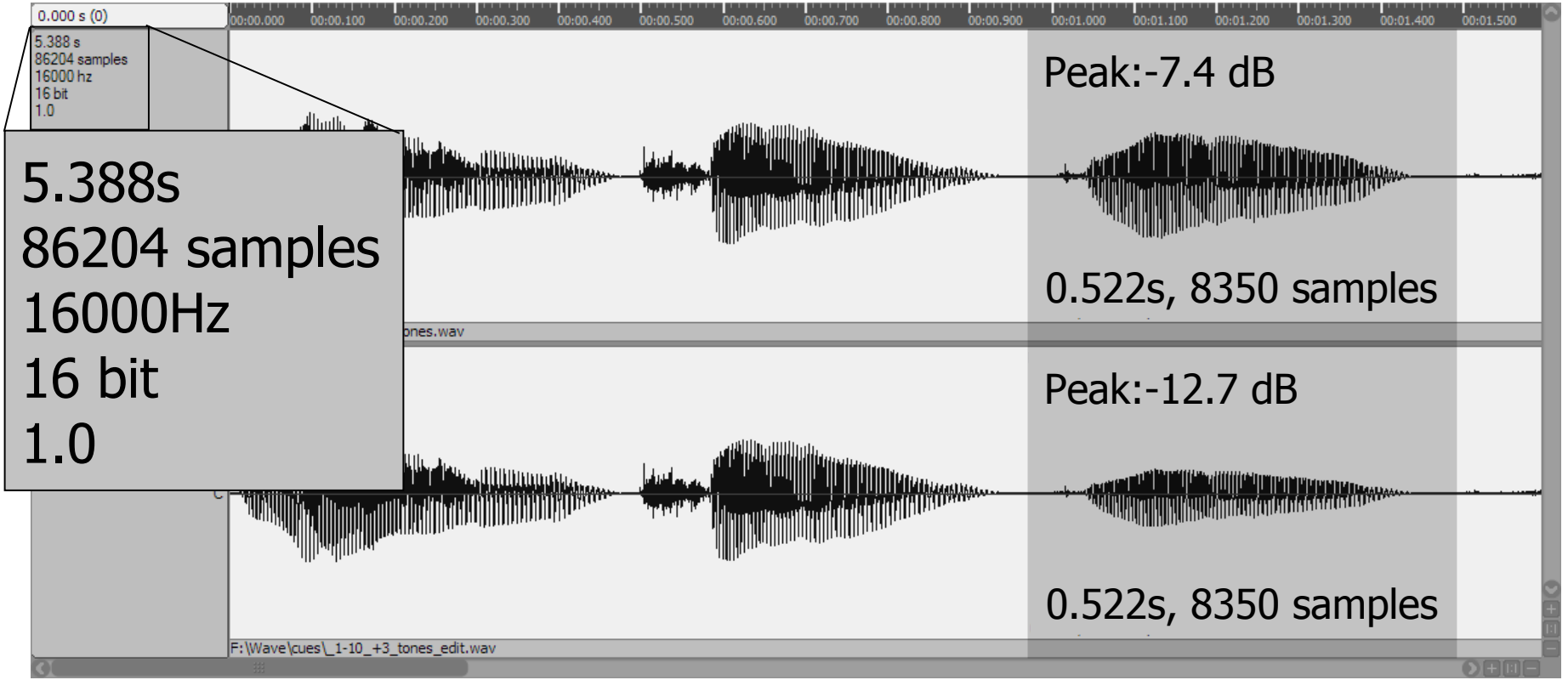
- Very important
  - New differences
  - New Asserts and Crashes
- Less important
  - System or execution failures
- Not important
  - Known issues
  - Success

Diff Application

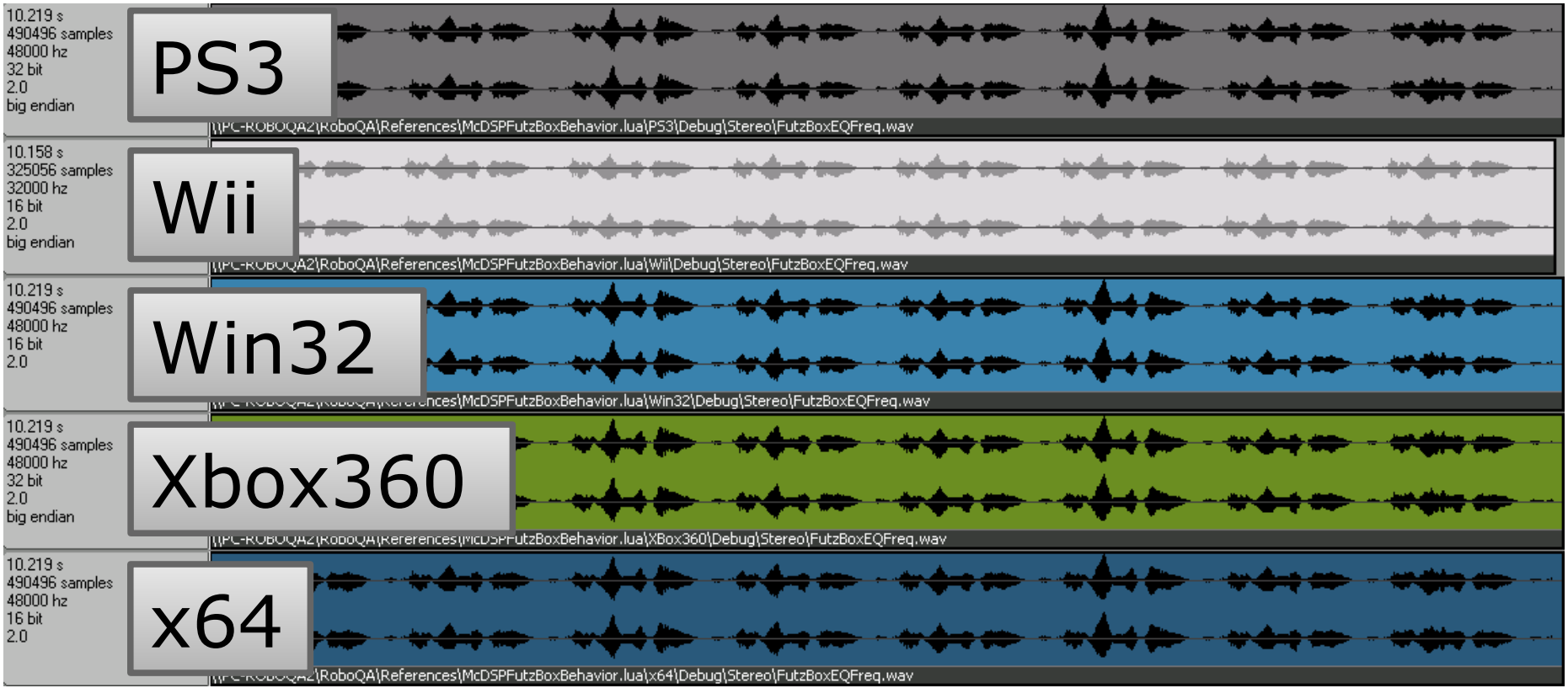
Diff Tool


0.000 s (0)
00:00.000
00:00.500
00:01.000
00:01.500
00:02.000
00:02.500
00:03.000
00:03.500
00:04.000
00:04.500
00:05.000
00:05.500
00:06.000
00:06.500
00:07.000
7.419 s
327168 samples
44100 hz
16 bit
2.0
Reference
FR
C:\temp\wav\A.wav
7.419 s
327168 samples
44100 hz
16 bit
2.0
Daily Test Output
FR
C:\temp\wav\B.wav
7.419 s
327168 samples
44100 hz
16 bit
2.0
Peak: -3.3 dB
Diff
FR
C:\temp\wav\A.wav.diff.tmp

Measure Tools



Compare Across Platforms



Diff Algorithm

```
[] Diff( a[], b[] )
{
    for i = 0; i < min(a.count, b.count)
        diff[i] = a[i] – b[i];
    return diff;
}
```

Lua Scripts

Lua Programming Language

- Lua
  - Simple – Light - Powerful
  - Easy type conversion (Coercion)
  - Garbage collection
- Advanced Lua
  - Use closures to define dynamic functions
  - Use tables to define test data

Script Example

```
function Test1()
    -- Show expectations
    LogMsg( "Hear noisy sound for 1 sec, then clean." )

    -- Set game variable
    AK.SoundEngine.SetRTPCValue( "Intensity", 100, g_AkDefaultGameObject )

    -- Play the sound and wait
    AK.SoundEngine.PostEvent( "FutzBox_Preset2", g_AkDefaultGameObject )
    Wait( 1000 )

    -- Set game variable again
    AK.SoundEngine.SetRTPCValue( "Intensity", 0, g_AkDefaultGameObject )
    Wait( 1000 )

    AK.SoundEngine.StopAll()
end
```

Script Wrapping

```
--RoboQA +All[Stereo,51] -Wii

-- Define the tests here
-- ...

-- Start the script
AkDefaultScriptMain("Bank", {Test1, Test2, Test3})
```

A Great Deal of Data

Multiplication of data

- Over 200 test scripts
- Over 5000 test functions
- 10 platforms
- Debug, Profile, Release
- 5.1 and Stereo
- 300 000 combinations!

x86/x64/vc9/vc10
Xbox360
PS3
Wii
WiiU
3DS
Vita
Android
Mac
iOS

Disk space

- 350 GB of wav files
- 150 hours of audio (6 days+)
- Select configuration/platform per test
- 90 000 reference wav files
- Keep tests short!

Output Hashing

- Do not transfer 350 GB on network!
- Avoid sending redundant information
- While executing a test:
  - Hash the wav data
- Only transfer if different from reference
- Save bandwidth
- Save time

Other issues

Handling constant randomness

- Tests must always sound the same
- Set the seed to a constant value
- Same random sequence every time

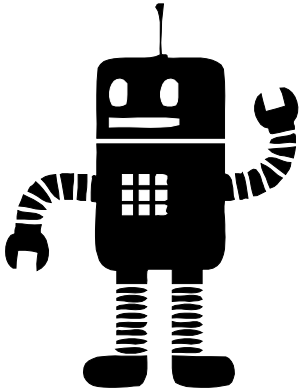
Non-audible differences

- Some differences are non audible
- Automatically accept -90dB
- Run peak level analysis on the difference
- Focus on serious issues

Other forms of testing

Performance Testing

- Run suite of benchmark scenarios
- Use the profiling services to calculate CPU usage
- Run in real-time (not offline)
- Run on all platforms
- Compare performance over time

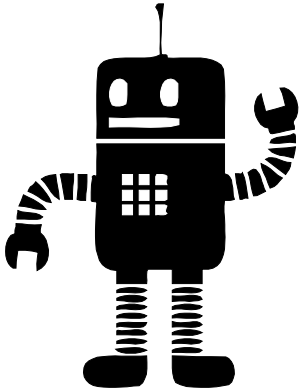


Performance scenarios

- Examples:
  - Playing 32 vorbis voices from memory
  - Running 64 EQ plugin simultaneously
- Run the scenario a couple of times
- Store the mean and variance

Stress Testing

- Cover the 10% non tested
- Test the memory allocation failures
- Random null pointer on alloc()
- Run at repetition
- No output recording
- No constant random seed
- Catch any crash



What is Next?

Code coverage

- Improve test metrics
- How much code has been tested?
- Use a code coverage tool
- Identify non-tested code

Future improvements

- Better integration with bug tracker
- Collect crash dumps
- Detect audio glitches
- Save Profiler output

Applying this to your game

- Audio tools
- Effects or sources plug-in
- Complex audio structures
  - Guns
  - Cars
- Any audio code

Game walk-through recording

- Build a script while playing the game
- Replay the script every day
- Record the audio
- Seed the random with constant value

Game Simulator

- Check out:
  - Wwise Game Simulator
  - AkLuaFramework.lua
  - StartOutputCapture()
- Test!

Question?