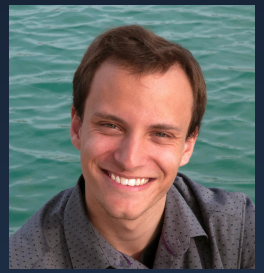
GENTEST

Silu Shen Tristan Miller Zane Fink Joshua Johnson
Mentored by Fabio Santos
Client: Associate Professor Alex Groce, NAU

Team Members



Joshua Johnson
Team Lead

Silu Shen
Recorder

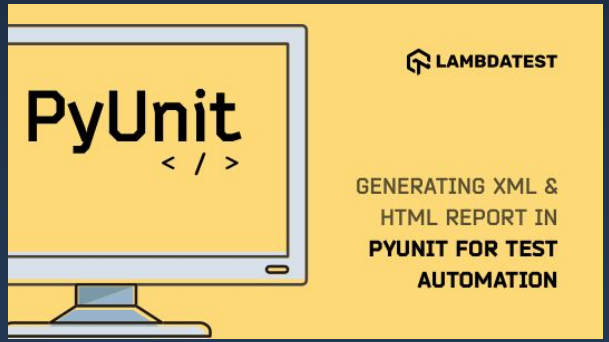
Zane Fink
Coder

Tristan Miller
Architect


Lion

About Unit Testing

- Unit testing - A test on a small, discrete piece of code in a single code module.
- There are hundreds of unit testing frameworks.
- Automatic Unit Testing can decrease software defects by 62 ~92%



About DeepState

Associate Professor Alex Groce

- Developed by Alex Groce and Trail of Bits

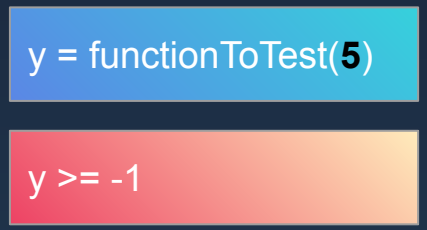
1. A unit testing library for C, C++ languages

2. Provides built-in automatic testing tools open-source

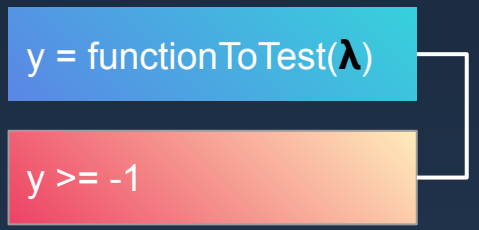
Workflow

functionToTest(int x) : int

Typical manual workflow



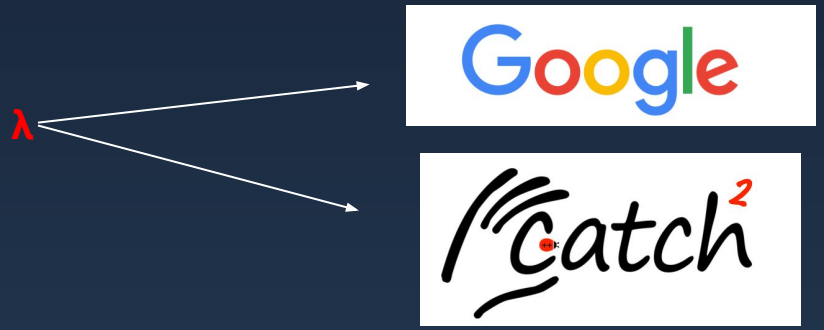
DeepState workflow



λ = 5, λ = 5023
λ = 1600382, ...

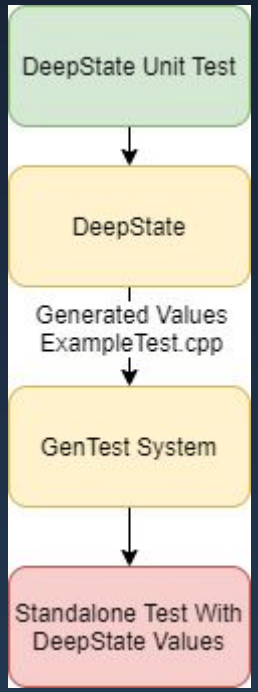
Problem Statement

- The current workflow is too coupled with the Deepstate library.



Solution Overview

- Output code will be similar to that of a manual workflow
- Outputted code will contain no references to the DeepState library
- Replace value generation with hardcoded values
- Outputted code should be compilable
- Outputted code should have identical behavior



Solution Overview

- Utilize multiple data files
- Allows developers to view what values are causing issues
- Used by DeepState to output standalone tests for every test

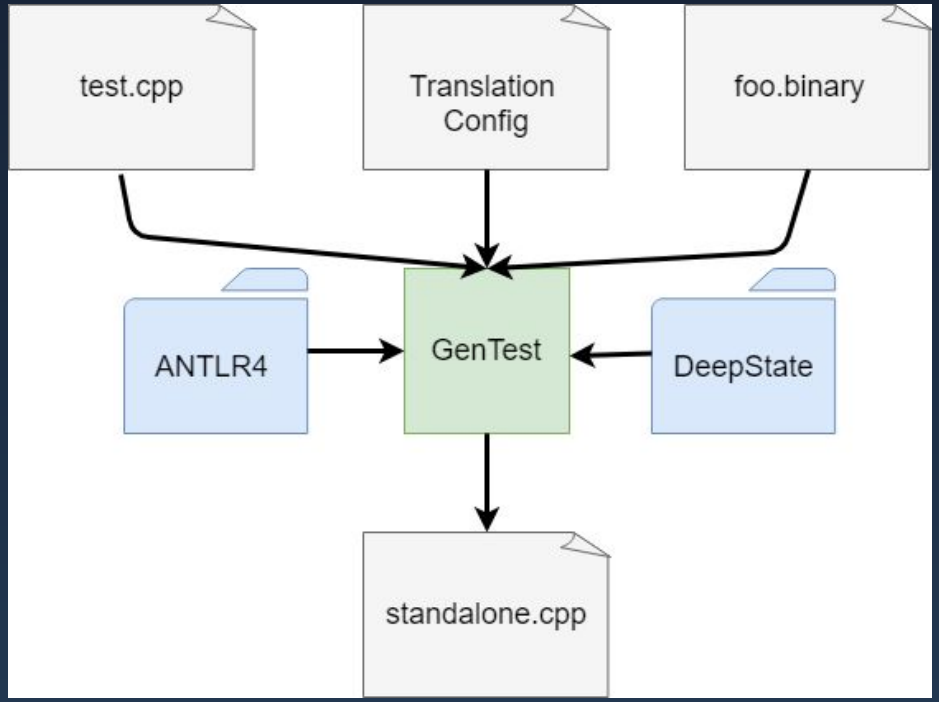
Implementation Overview: Requirements

Key Requirements from conversations with client:

- Extension to DeepState that allows the creation of standalone tests.
- These standalone files must contain the values generated by DeepState.
- Generated values can be inserted variables and loops.
- Support for user-specified target testing framework syntax (Google Test, Catch2, etc.)

Implementation Overview: Solution

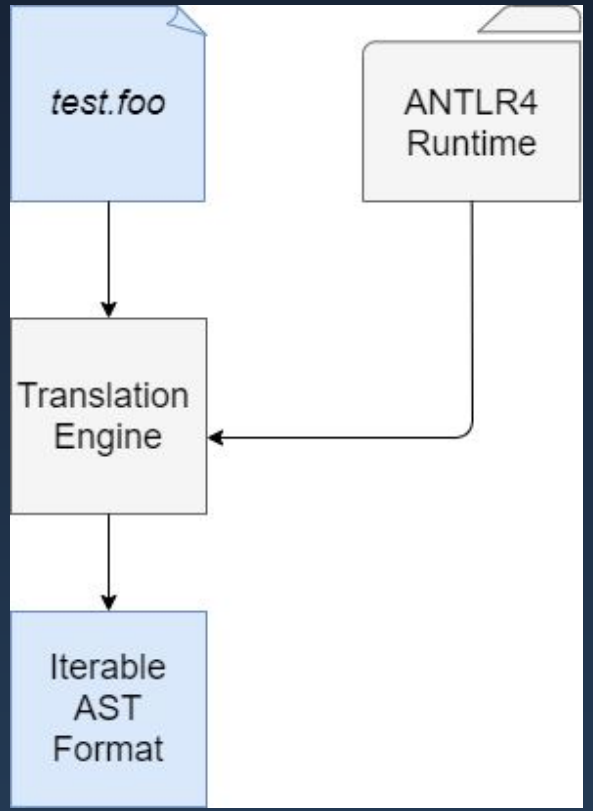
- Inputs:
  - Unit test written in DeepState Syntax
  - Configuration specifying translation rules
  - Binary Data Generated by DeepState
- Outputs:
  - Standalone unit test in syntax of target framework



Architecture Overview: Translation Engine

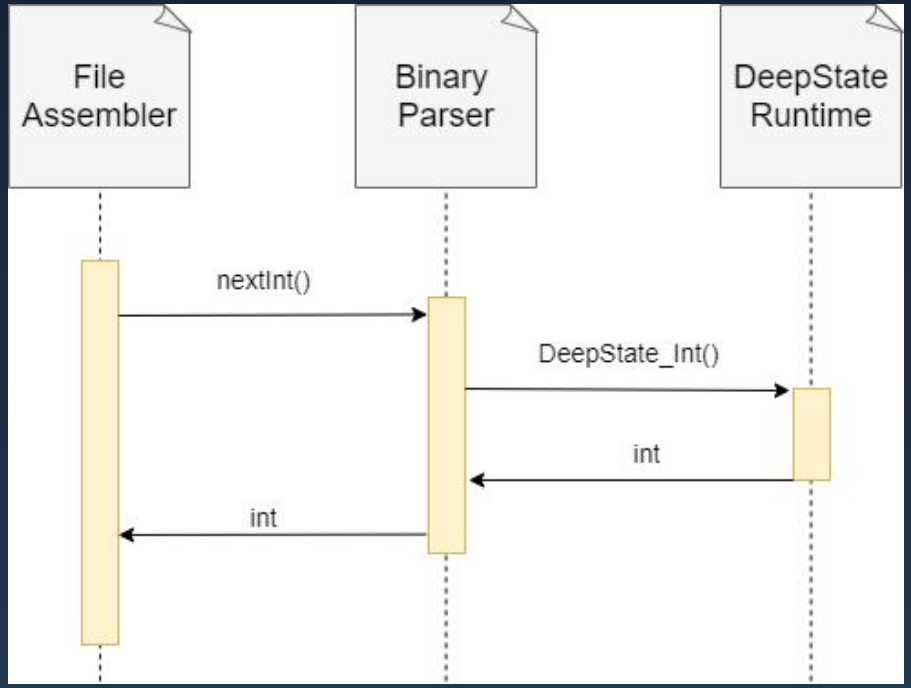
- Breaks a test file down into an Abstract Syntax Tree (AST)

- Provides an interface for traversal of the AST.



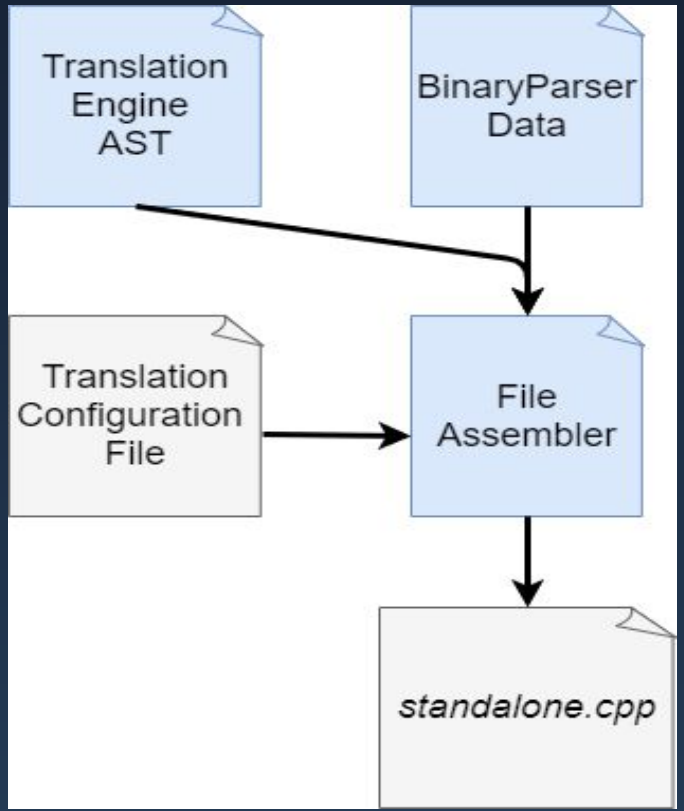
Architecture Overview: Binary Parser

- Provides an object-oriented interface to the DeepState runtime.
- For complete information on supported types, see Software Design Document.
- Used by the File Assembler to make symbolic values concrete.



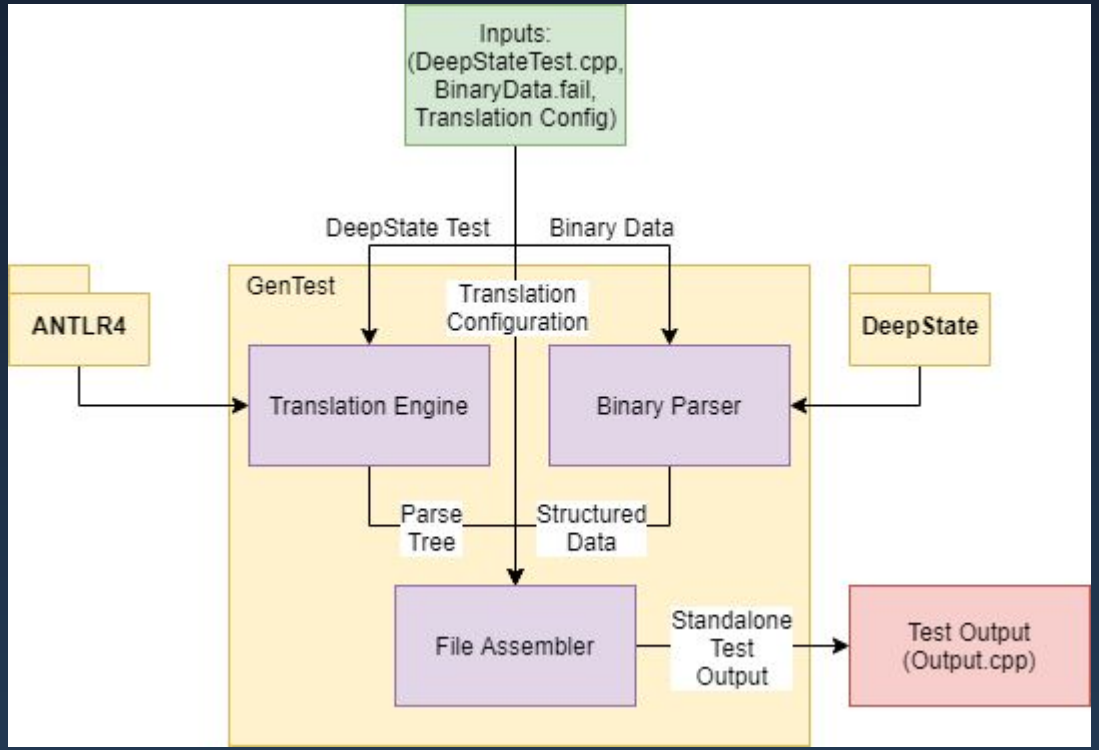
Architecture Overview: File Assembler

- Traverses the Abstract Syntax Tree (AST) created by the Translation Engine.
- Using the BinaryParser, replaces symbolic values with concrete ones.
- Produces standalone output file in user-specified testing framework.



Architecture Overview: Integration

- The modules, with ANTLR4 and DeepState, work together at runtime to translate and insert concrete values into a standalone C++ test.



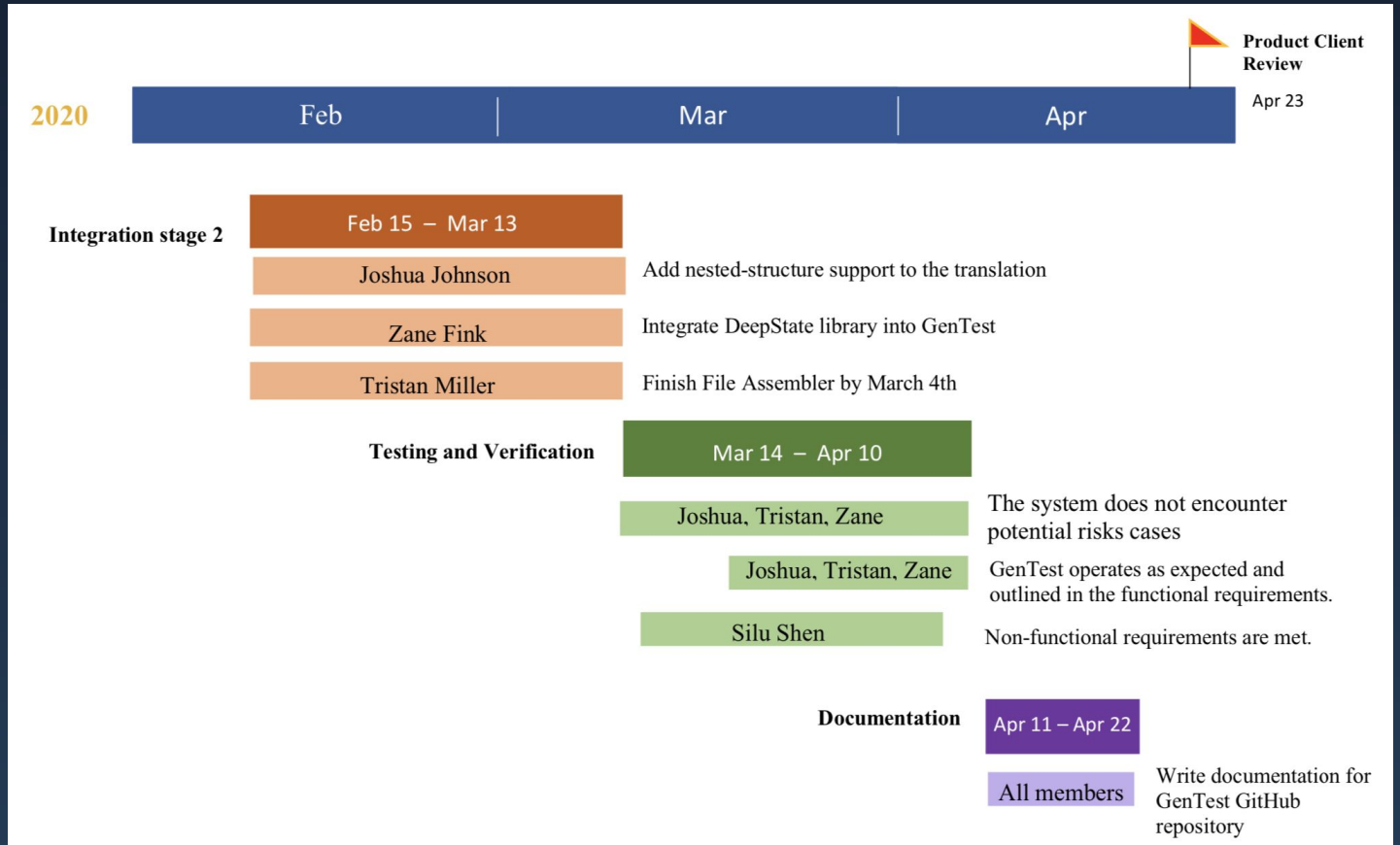
Challenges and Resolutions

- Complex syntax of C++ makes parsing it non-trivial.
  - Solution: The use of ANTLR4: a parser generator for C/C++

- Due to ANTLR4 C++ support being new, there are library integration problems.
  - Solution: Incorporating the ANTLR4 runtime into a build environment.

- Integration of GenTest software into DeepState source code is complex.
  - Solution: Discussions with DeepState team to ensure existing coding standards are maintained.

Schedule


Product Client Review
Apr 23
2020
Feb
Mar
Apr
Integration stage 2
Feb 15 – Mar 13
Joshua Johnson
Add nested-structure support to the translation
Zane Fink
Integrate DeepState library into GenTest
Tristan Miller
Finish File Assembler by March 4th
Testing and Verification
Mar 14 – Apr 10
Joshua, Tristan, Zane
The system does not encounter potential risks cases
Joshua, Tristan, Zane
GenTest operates as expected and outlined in the functional requirements.
Silu Shen
Non-functional requirements are met.
Documentation
Apr 11 – Apr 22
All members
Write documentation for GenTest GitHub repository

Conclusion

- The Problem
  - DeepState tests are highly coupled with DeepState itself
  - Need a way to increase portability
- Our Solution
  - Generates standalone tests without the need for DeepState
  - Standalone tests will utilize a testing framework of a users choice as well as the generated values
- Moving Forward
  - Finish development
  - Begin testing of the final product

Conclusion

- Thank you for listening

- Questions?