Testing Video Games
from a Software Engineering Perspective

Cristiano Politowski - Ph.D. Student
Concordia University - Montreal, Canada

Wuji: Automatic Online Combat Game Testing Using Evolutionary Deep Reinforcement Learning

Yan Zheng[1,*], Xiaofei Xie[2,*], Ting Su[2], Lei Ma[3], Jianye Hao[1,✉], Zhaopeng Meng[1],
Yang Liu[2], Ruimin Shen[4], Yinfeng Chen[4], Changjie Fan[4]
[1] *College of Intelligence and Computing, Tianjin University, China.*
[2] *Nanyang Technological University, Singapore.*
[3] *Kyushu University, Japan.*
[4] *Fuxi AI Lab, Netease, Inc., Hangzhou, China.*

Summary of the results

We perform an empirical study to characterize game bugs by **analyzing 1,349 real bugs** from four industrial games. We propose four **test oracles** for four types of bugs.

We propose an **on-the-fly automated testing framework** based on evolutionary deep reinforcement learning.

The approach consists in using a Evolutionary Deep Reinforcement Learning technique called actor-critic (A2C) for the agent accomplish the mission and Evolutionary Algorithms with Multi-Objective Optimization explore the game space.

Due to the complexity and heavy user interactions of games, currently, game testing is mainly ***dependent on human*** *testers.*

Traditional software is also hard to test, but game development additional concerns: Fun Factor, Balance, Game Level/World, AI Testing, Audio Testing, Physics, Realism.

Game testers are cheaper than engineers.

Game testers have poor work conditions.

*Most game companies adopt some **ad-hoc manual testing** without using systematic and automated testing solutions.*

Testing different types of games is a challenge:

- 2D vs 3D
- Platformer vs First-Person Shooter vs Real Time Strategy

Different logic means different assertions.

*The ad-hoc manual testing is costly and is **inefficient** in discovering bugs for large games.*

*As a result, many **bugs** are still discovered long after the official **release**.*

Lacking a source but we can agree if it is the same for Traditional Software. Is it?

Plus, it is a fact that from the top games on Steam, 80% needs critical updates

However:

1. It is still a subset of specific games on a specific platform.
2. There will be always bugs, specially with games as a service.

ABOUT THE BOOT.INI ISSUE

2007-12-11 - BY CCP EXPLORER - EVE DEV-BLOGS

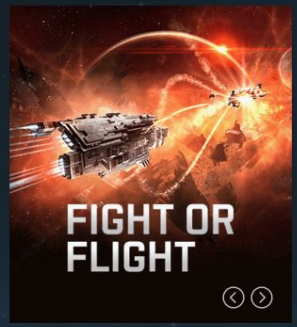
My name is Dr. Erlendur S. Thorsteinsson and I have been directing the EVE Online Software Group for the better part of this year. From my previous work experiences in the antivirus industry and following CCP for quite some time now, I have come to appreciate the need for full disclosure when things don't go according to plan.

Shortly after releasing EVE Online: Trinity at 22:04 GMT on Wednesday, 5 December, we started receiving reports that the Classic to Premium graphics content upgrade was causing problems to players by deleting the file C:\boot.ini, which is a Windows system startup file. In some cases the computer was not able to recover on the next startup and would not start until the file had been fixed. In this dev blog I want to tell you how this happened.

In the weeks leading up to the release of Trinity, one of our concerns was how to deliver this update to our players in a very short amount of time and to players that might not all have a good internet connection. Significant effort was therefore put into making the Classic to Premium graphics content upgrade as small as possible at various stages in the process.

Initially we had planned to use our third-party patching technology to create the graphics content upgrade file; however, we realized late in the development cycle that while it was suitable for creating small update patches for files that already existed on the computer, it did poorly in compressing new files. Since the content upgrade only had to replace two small text files - boot.ini and manifest.dat - and then copy 1.43 GB of new files, resDX9*.stuff, we decided to switch to our third-party installer technology that had superior compression, LZMA, enabling us to shrink the download from about 866 MB to 584 MB. The script to create the graphics content upgrade installer file was added to our source code management system on 30 November at 14:59 GMT. The first graphics content upgrade to use the installer technology was released on Singularity on Sunday, 2 December, and then a version for the final build, 45017, was released on Tranquility at 22:04 GMT on 5 December when the server was opened.

It might appear from the installer log that we made the mistake of explicitly deleting \boot.ini instead of



PATCH NOTES READ MORE >

2020-01-16 - BY CCP CONVICT
Patch Notes for January 2020 Release

2019-12-10 - BY CCP CONVICT
Patch Notes for December 2019 Release

THE SCOPE READ MORE >

*The direct **loss** caused by the bugs in this game [big massive online game on Netease] is about $2 million each year.*

~30 human testers

For example, balancing problems cause loss in microtransactions, meaning: I don't need to pay for this magical item if I can get easily.

Also, the phenomenon I call **zero-day Inquisition** or **zero-day witch hunt:** non-polished games are massacred on reviews on released date causing financial loss and company credibility.

Warcraft 3: Reforged launch was 'a hard week,' Blizzard president says

By Andy Chalk 4 days ago

J. Allen Brack acknowledged that it hasn't gone well, but said that Blizzard isn't giving up.

COMMENTS

WARCRAFT III: REFORGED PC

Release Date: Jan 28, 2020

Summary | Critic Reviews | User Reviews | Details & Credits | Trailers & Videos

60 Metascore
Mixed or average reviews based on **32 Critic Reviews**

What's this?

0.5 User Score
Overwhelming dislike based on **28189 Ratings**

Your Score 0

Summary: A Classic Favorite, Reforged. Warcraft III: Reforged is a reimagining of the real-time strategy game that laid the foundation for Azeroth's most epic stories. It is a remake featuring a thorough visual overhaul, a suite of contemporary social and matchmaking features, and more. Command the... **Expand**

Developer: Blizzard Entertainment
Genre(s): Strategy, Real-Time, General
# of players: Online Multiplayer
Cheats: On GameFAQs
Rating: T
More Details and Credits »

How BioWare's *Anthem* Went Wrong

Jason Schreier
4/02/19 11:00AM • Filed to: ANTHEM

1.8M 951 123

ANTHEM PlayStation 4

Electronic Arts | **Release Date:** Feb 22, 2019 | **Also On:** PC, Xbox One

Summary | Critic Reviews | User Reviews | Details & Credits | Trailers & Videos

54 Metascore
Mixed or average reviews based on **28 Critic Reviews**

What's this?

3.4 User Score
Generally unfavorable reviews based on **1239 Ratings**

Your Score 0

Summary: Anthem is a shared-world action-RPG in which players delve into a vast world teeming with amazing technology and forgotten treasures. The world is also filled with savage beasts and ruthless marauders where Freelancers are called to defeat the forces plotting to conquer humanity. In Anthem,... **Expand**

Developer: BioWare
Genre(s): Role-Playing, Action RPG
# of players: Up to 4
Cheats: On GameFAQs
Rating: T
More Details and Credits »

Testing techniques

Gameplay testing with humans.

Ad-hoc solutions: many!

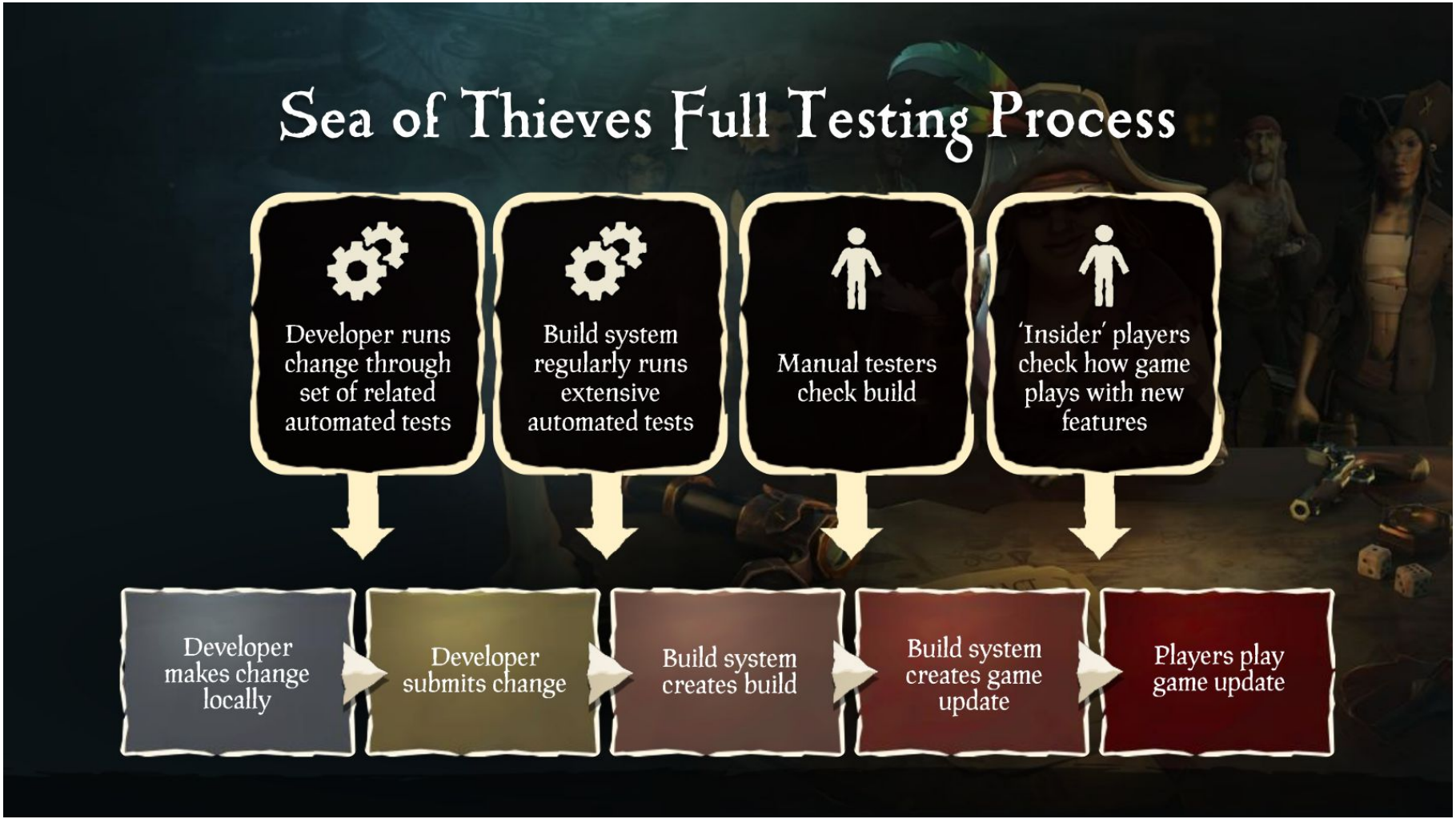
Variation of unit tests

Not enough: UI/UX testing is not game testing.

Case study:
RARE’ Sea of Thieves

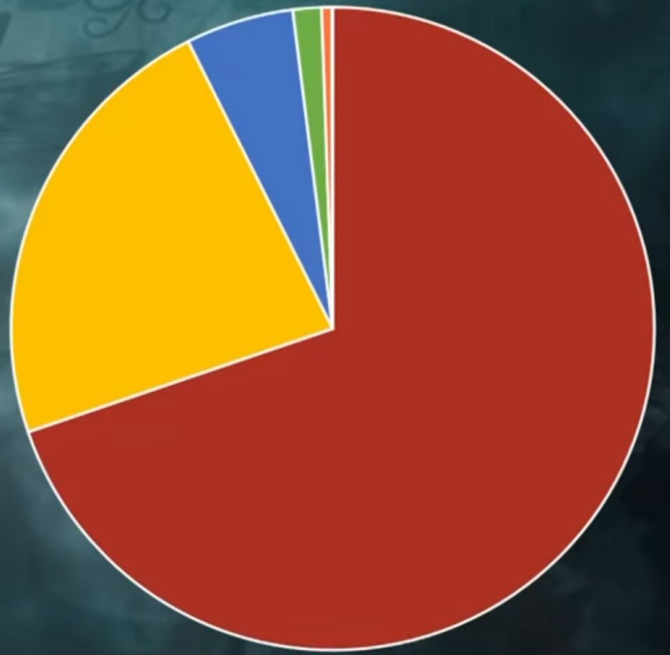
(game as a service)

Sea of Thieves Full Testing Process



Automated Tests Snapshot

Actor	16200 (70.2%)
Unit	5290 (22.8%)
Map	1260 (5.4%)
Screenshot	334 (1.4%)
Performance	119 (0.5%)
Bootflow	9 (0.005%)

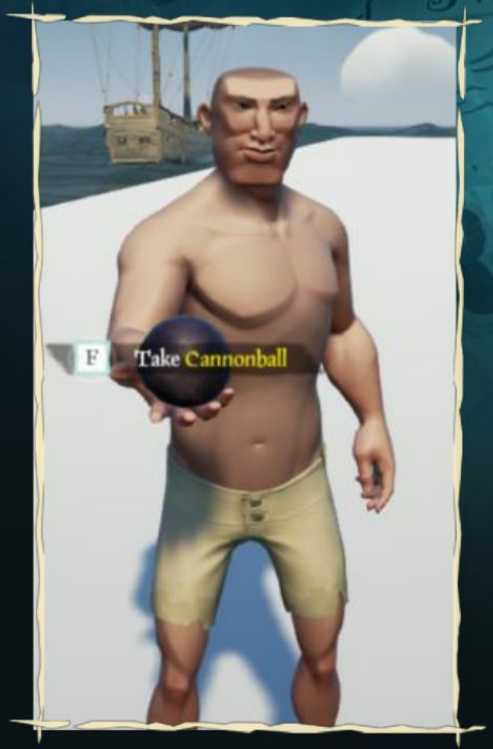


TOTAL = **23,212 TESTS**

\+ Asset Audit checks (81,700) = **104,912 TESTS!**


Integration Tests
Sea of Thieves
Actor Tests
UNREAL ENGINE
Unit Tests
Slower
Less Reliable

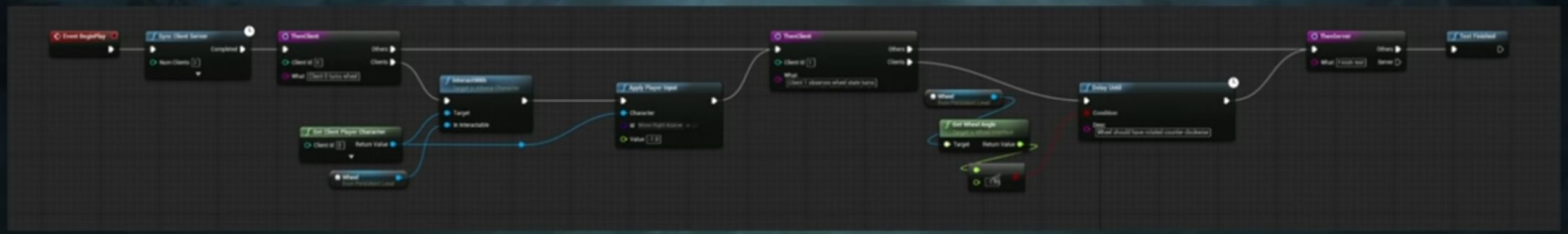
'Golden Path' Integration Testing



Actor	• *Player can't give item due to having no item* • *Player can't give that type of item* • *Player not close enough to give item*
Integration	• *Player successfully gives item to another player.*

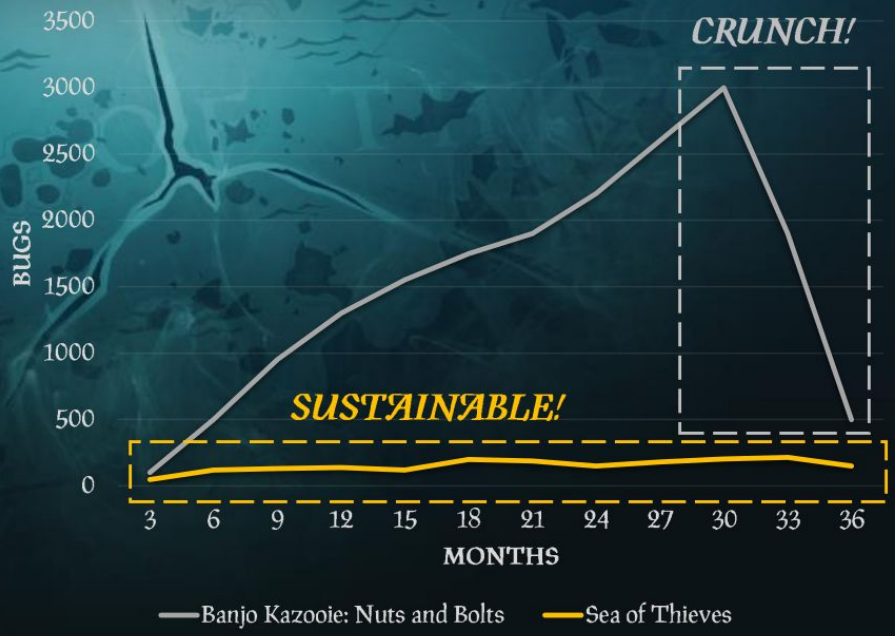
Ratio of actor tests to integration tests is around 12 : 1.

Map Tests- Capabilities



Benefits of Extra Build Confidence

- Reduced time to verify build
- Reduced manual testing
- Very low bug count
- Reduced crunch



Case study:
RIOT’ League of Legends

(game as a service)

The project context

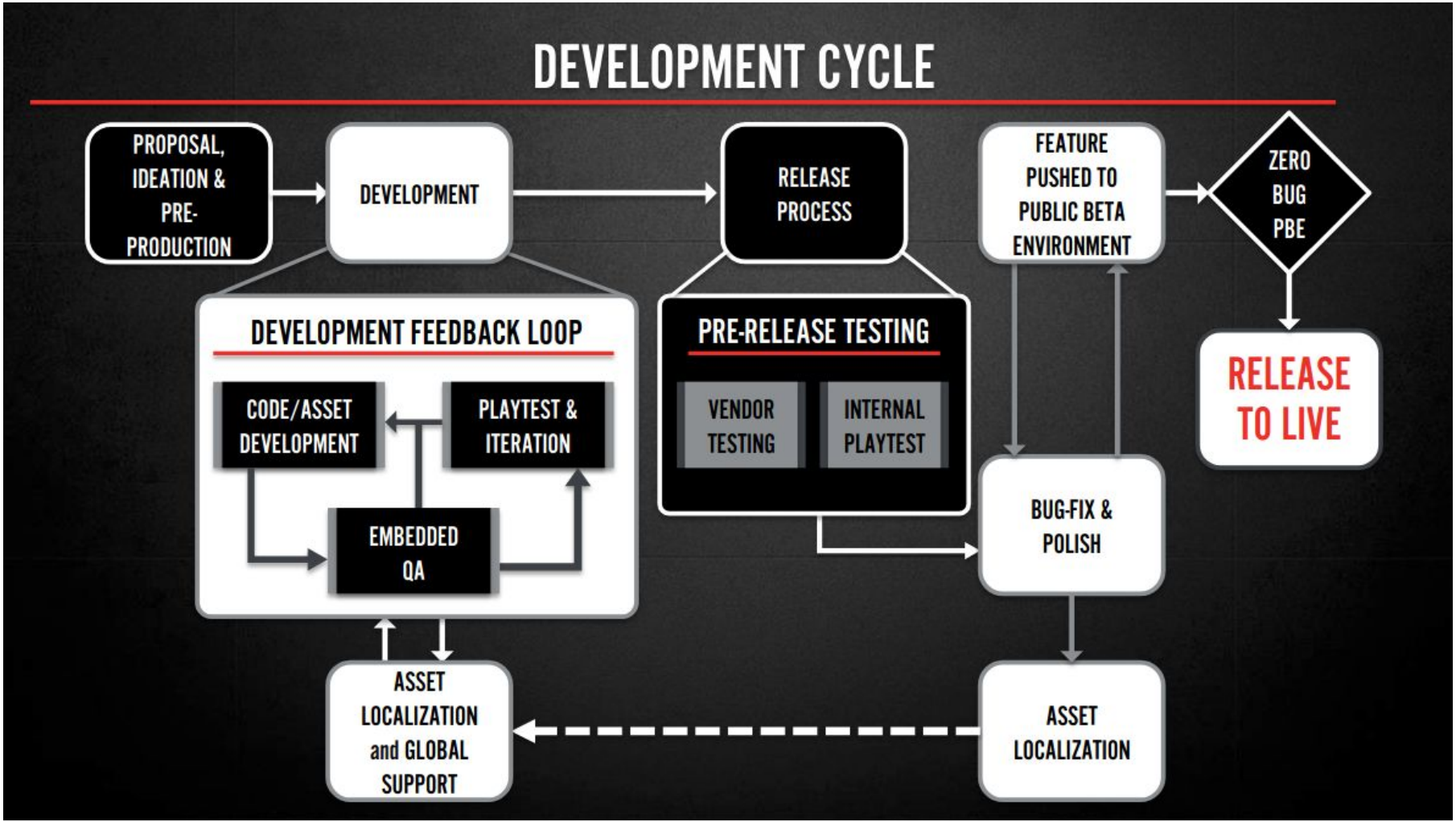
The game is a Live Service: 11M daily players with 5M concurrent

One bug affects millions

New content multiple times a month

Rapid iterations with daily playtests

Continuous delivery force to use days rather than weeks to test


DEVELOPMENT CYCLE
PROPOSAL, IDEATION & PRE-PRODUCTION
DEVELOPMENT
RELEASE PROCESS
FEATURE PUSHED TO PUBLIC BETA ENVIRONMENT
ZERO BUG PBE
RELEASE TO LIVE
DEVELOPMENT FEEDBACK LOOP
CODE/ASSET DEVELOPMENT
PLAYTEST & ITERATION
EMBEDDED QA
PRE-RELEASE TESTING
VENDOR TESTING
INTERNAL PLAYTEST
BUG-FIX & POLISH
ASSET LOCALIZATION and GLOBAL SUPPORT
ASSET LOCALIZATION

Case study:
EA’ Call of Duty

What?	When?	Parallel?	Description
CI	Every 30 min	Yes	Compile the game code, tools code; convert assets; run the game on 3-4 maps, on all platforms; device debug; unit tests.
All maps	Every hour	No	Boot tests every map in the game and check the resulting image/rendering.
Nightly test	Every day	No	Load the map of the game and put the character on a different locations keeping him there for around 30 secs and capture some performance metrics.
Release build	Longer but variable	No	Build a full package version of the game to send to QA or put on discs.
Branch maintenance	Variable	No	Automatic merge-down from stable version to less stable versions.

Legend: Passed | Failed | Running | Exception | No data

	Bootstrap	Convert/Link Assets			Compass		Convert/Link Assets		Run Dedicated			Run Game									Screenshots (PS4)		
		all			Lint	Test	Convert Data	Compile Scripts Simple	PC	PS4	XB3	[redacted]			[redacted]			[redacted]			[redacted]	[redacted]	[redacted]
		PC	PS4	XB3								PC	PS4	XB3	PC	PS4	XB3	PC	PS4	XB3			
✓ #26765 alchapman																							
Change 1968696	[redacted]																						
✓ #26764 integrator																							
Change 1968688	[redacted]																						
✓ #26763 integrator																							
Change 1968684	[redacted]																						
✓ #26762 integrator																							
Change 1968681	[redacted]																						
✓ #26761 kmckisic																							
Change 1968674	[redacted]																						
✓ #26760 jminkoff																							
Change 1968661	[redacted]																						
✓ #26759 jminkoff																							
Change 1968653	[redacted]																						
✓ #26758 jminkoff																							
Change 1968643	[redacted]																						
✓ #26757 jminkoff																							
Change 1968642	[redacted]																						
✓ #26756 sgil																							
PreSubmit ✓ Change 1968637	[redacted]																						
✓ #26755 jminkoff																							
Change 1968634	[redacted]																						
✓ #26754 aroycewicz																							
Change 1968630	[redacted]																						
✓ #26753 jminkoff																							
Change 1968627	[redacted]																						
✓ #26752 dhorrocks																							
Change 1968623	[redacted]																						
✓ #26751 gsmith																							
Change 1968620	[redacted]																						
✓ #26750 gsmith																							
Change 1968617	[redacted]																						
✓ #26749 cpierro																							

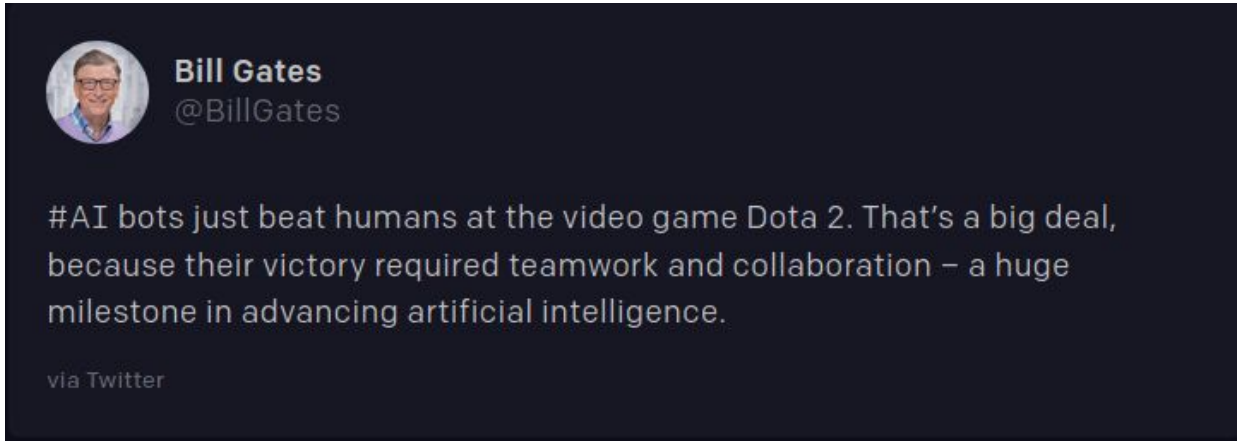
AI for testing != testing an AI

Deep Reinforcement Learning on playing games

2015 - OpenAI on playing Atari games using raw pixels as input

2016 - DeepMind' AlphaGo winning 4-1 GO grandmaster

2019 - OpenAI Five wins Dota 2 match against professionals



Challenges of game testing and AI

DLR focus on finish the game, not find bugs.

An Oracle is needed.

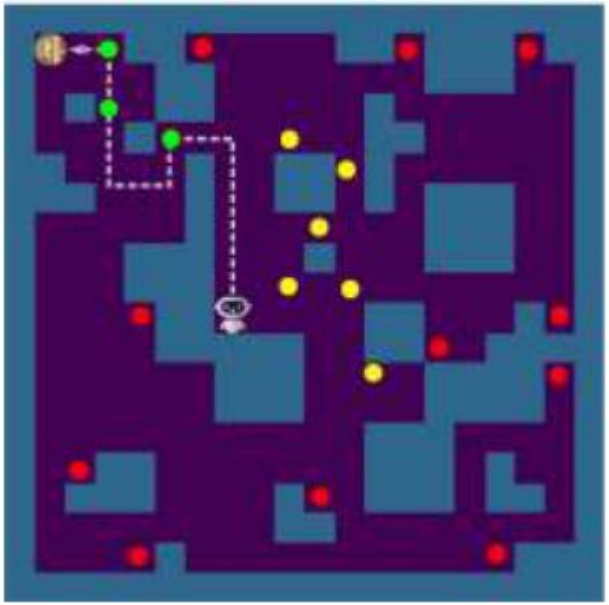
“(...) due to the uniqueness of game software, the test oracle also quite differs from other kinds of software, which is yet not well-studied.”

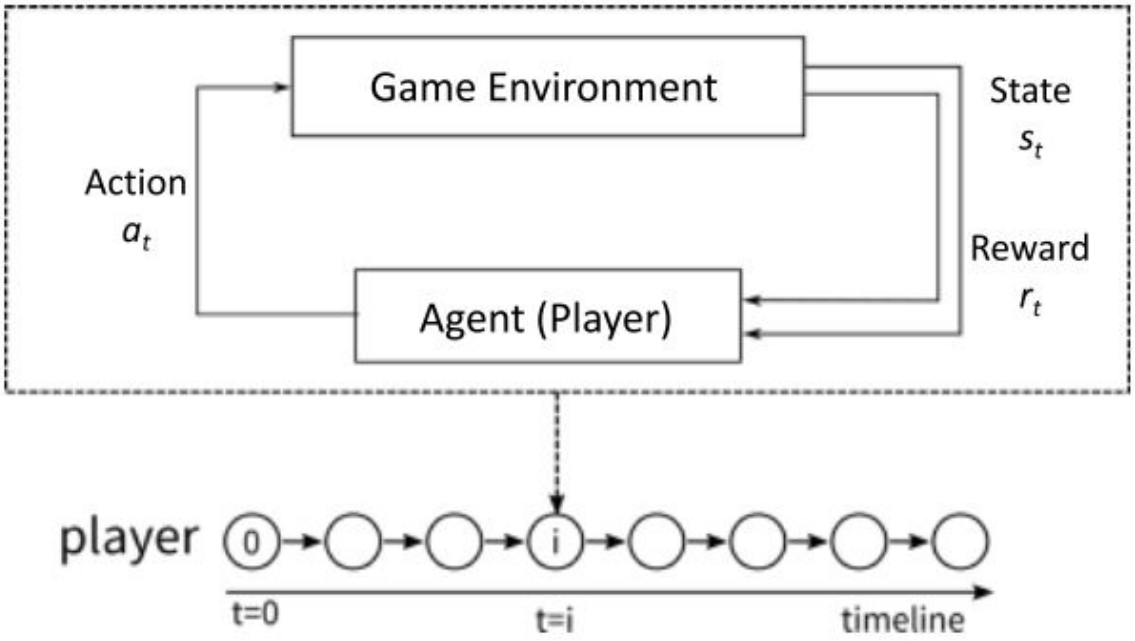
“The objective of Wuji is to detect game bugs by exploring states as much as possible.” (...) while trying to complete the game.

“Wuji performs an on-the-fly testing while constantly training the agent policies” (reasoning)

PS.: game must be deterministic!

Game Playing -> Markov Decision Process





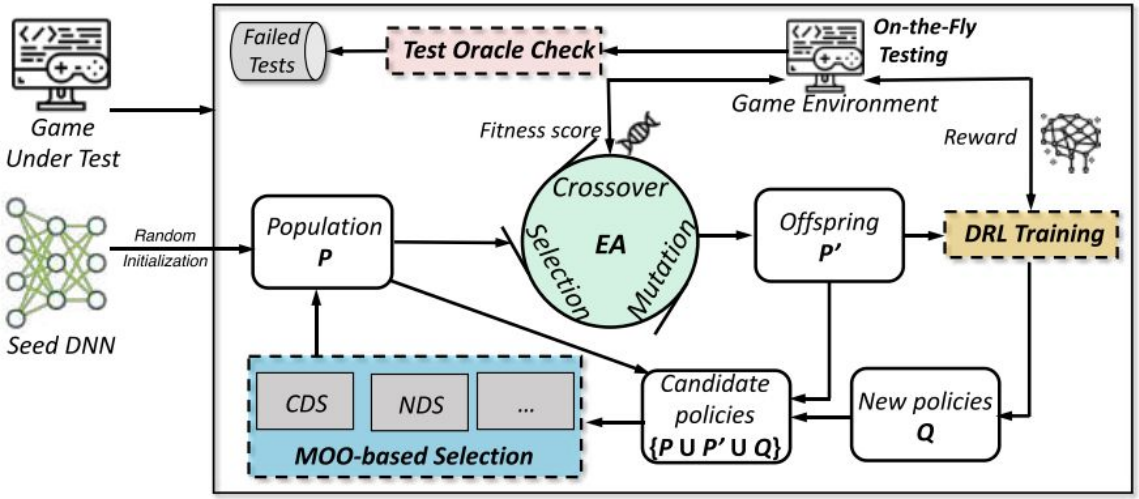
Oracles and Bugs

I'm no supposed to fly!

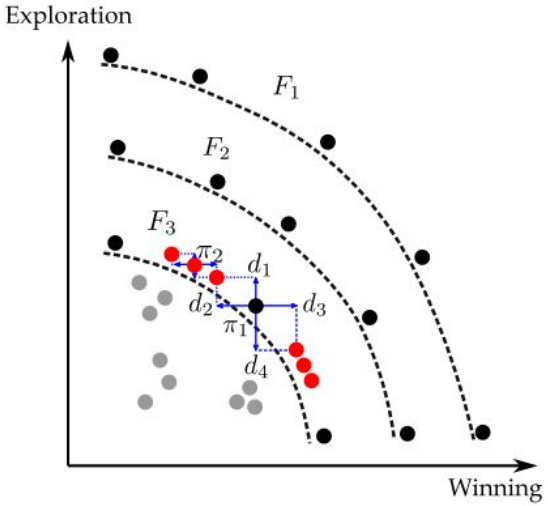
Games	L10	NSH	L12	LH01	Total
Crash	89	16	10	30	145(**10.75%**)
Stuck	22	7	9	27	65(**4.82%**)
Logical	379	25	22	62	488(**36.17%**)
Balance	7	2	2	1	12(**0.89%**)
Experience	476	88	20	55	639(**47.37%**)
Total	973	138	63	175	1349 (**100.00%**)

I'm no supposed to fly in this level!

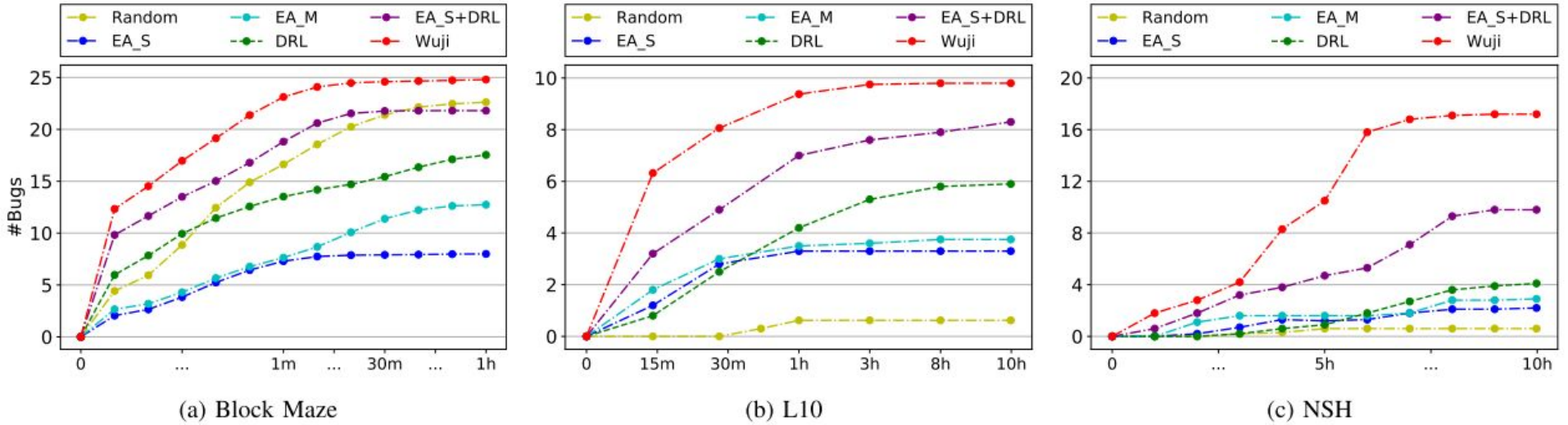
Framework



(a) The workflow of *Wuji*



(b) Visualization of MOO-based selection



It is about finding more bugs!

(my) Questions

The logical bug need specific assertions? This means that the agents/system need to know the game rules, items and all the variables of the game?

How would be the results if compared with human testers (performance)?

Which input and output were tested? Not clear

Does the framework needs to know the rules of the game?

How will you apply this in a real workflow scenario?

What is the input for the seed DNN?

What is an Action? One input?

Some thoughts

Game testing rely on (underpaid) human testers expertise

Automation can aid testers on repetitive tasks (regression tests)

AI can be used as a tool to aid developers testing their games quickly (AI for SE)

The paper uses games to validate an AI technique. There is no discussion on the SE side. How to implement/embed it on production? What are the drawbacks? What do I need to know about the game to use the tool? How different is the implementation for each game?

Oracles: how to identify bugs on systems?

Future

Game engines are the core of game development. Now, there is no workflow focused on testing the games as the developer use trial-and-error approach.

Putting the capabilities of the AI on a Game Engine offering a different way to work. Think TDD with steroids.

Notes from discussion

Game logic on games are spread or concentrated on a rule engines in business rules.

Extract the rules from games.

Analise Mega Man (or other franchise) games and see how the bugs spread in other versions.