
GUNS OF ICARUS
ONLINE

DEVELOPING *GUNS OF ICARUS ONLINE* THE UNITY WAY

BIG VISION, SMALL TEAM

BRIAN KEHRER, FORMER PROJECT DIRECTOR, CO-FOUNDER, MUSE GAMES
CREATIVE TECHNOLOGIST, PSYOP
FOUNDER, ICOSAHEDRA

THE VIEWS CONTAINED HEREIN ARE SOLELY MINE, AND DO NOT REPRESENT THOSE OF MUSE GAMES, OR ANYONE ELSE.

PART I

WHAT IS GUNS OF ICARUS ONLINE?

WHAT IS GUNS OF ICARUS ONLINE?

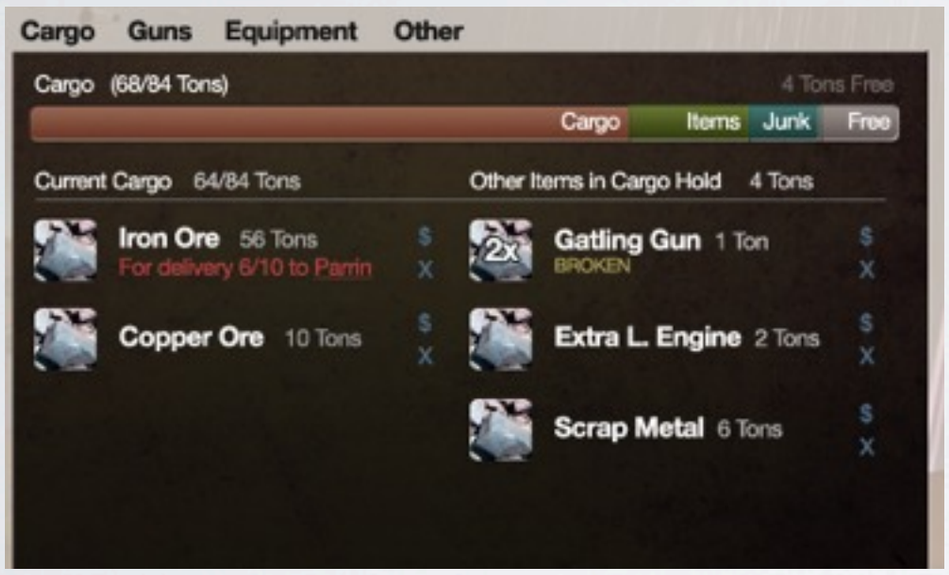
THE SPIRITUAL SUCCESSOR TO ***GUNS OF ICARUS***, RELEASED IN 2009

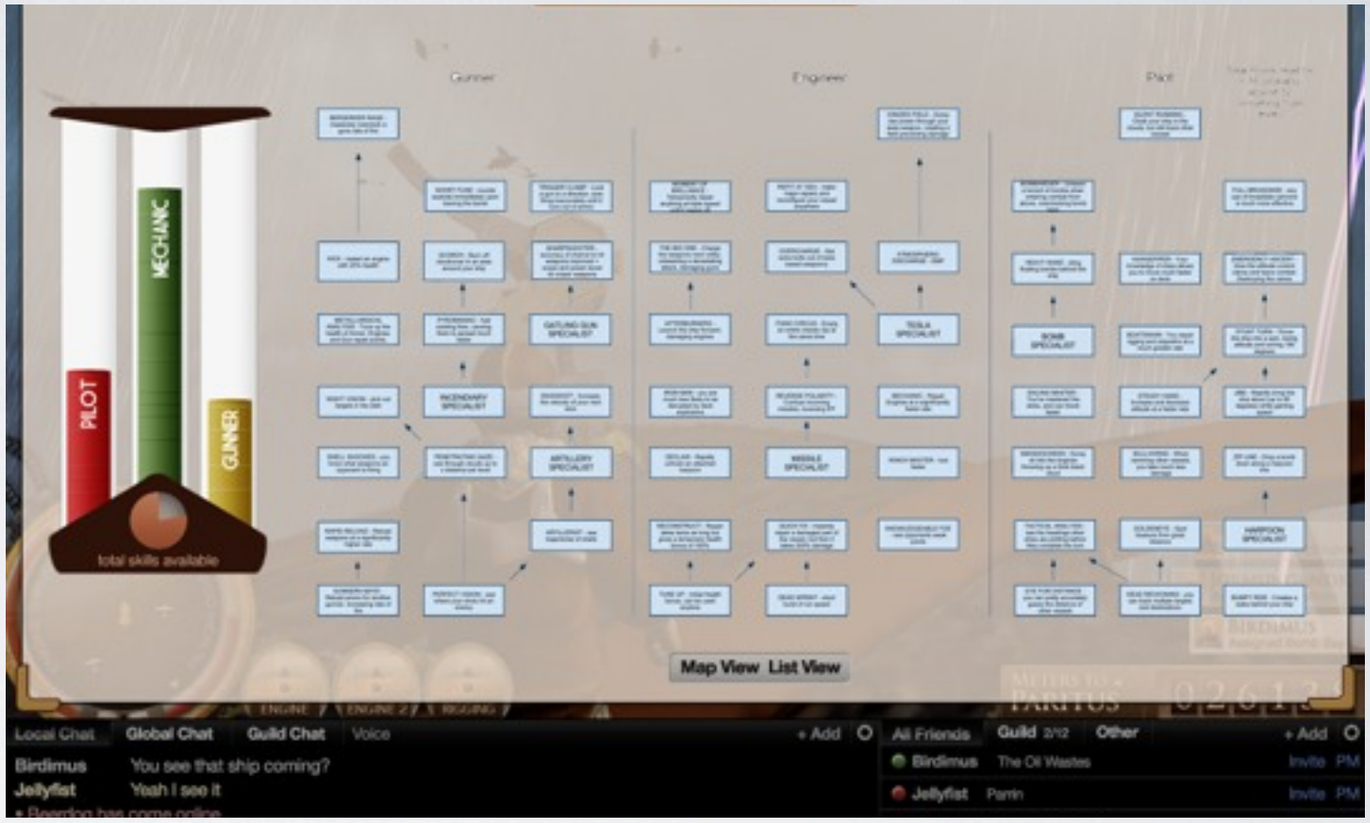

GUNS OF ICARUS
ONLINE

GOI ONLINE GDD - TABLE OF CONTENTS

OUR PUBLISHERS WANTED A BIG GAME. IN MAY, 2010, WE PITCHED A **130 PAGE** DESIGN DOC FOR ***GUNS OF ICARUS ONLINE***.

WHAT IS GUNS OF ICARUS ONLINE?





COOPERATIVE REALTIME MULTIPLAYER OPEN WORLD COMBAT, **CUSTOMIZABLE** SHIPS AND PLAYERS, SKILLS, **TRADING**, AUCTIONS, **PLAYER GENERATED TOWNS**, PROCEDURAL MISSION CONTENT, **PVE** AND **PVP**, AND A **MARKET ECONOMY**

WHAT IS GUNS OF ICARUS ONLINE?

AND BOSS BATTLES...

FOR THIS, OUR PUBLISHER OFFERED TO INVEST A GRAND TOTAL OF:

$300,000

TEAM MUSE - CIRCA 2009

WHAT IS GUNS OF ICARUS ONLINE?

DON'T CALL IT AN MMO!

DEFINITELY NOT AN MMO...

COOPERATIVE REALTIME MULTIPLAYER OPEN WORLD COMBAT, **CUSTOMIZABLE** SHIPS AND PLAYERS, SKILLS, **TRADING**, AUCTIONS, **PLAYER GENERATED TOWNS**, PROCEDURAL MISSION CONTENT, **PVE** AND **PVP**, AND A **MARKET ECONOMY**

GUNS OF ICARUS ONLINE

SHIPPED, OCTOBER, 2012

WHAT IS GUNS OF ICARUS ONLINE?

WHY DIDN'T WE FAIL?

YES, THERE WERE LONG HOURS, HARD WORK, EXCESSIVE QUANTITIES OF BEER, AND MEAGER LIVING CONDITIONS - BUT WE ALSO CHANGED OUR STRATEGY

DIGGING *GUNS OF ICARUS ONLINE* **OUT OF A VERY LARGE HOLE**

AN AGILE APPROACH TO GAME DEVELOPMENT

BRIAN KEHRER, FORMER PROJECT DIRECTOR, CO-FOUNDER, MUSE GAMES
CREATIVE TECHNOLOGIST, PSYOP
FOUNDER, ICOSAHEDRA

PART 2

NOT THE FUTURE WE EXPECTED

WE WERE **9 MONTHS INTO DEVELOPMENT,** BOUND TO A **PUBLISHER SCHEDULE** BEFORE WE REALIZED THE DEPTH OF OUR MISTAKE.

WE'D AMASSED NEARLY **TWO WORK-YEARS OF 'ENGINEERING DEBT'**, JUST TO MEET THE MILESTONES.

WE **WEREN'T ALLOWED TO CUT FEATURES** MADE OBSOLETE BY DESIGN CHANGES.

OH, AND **THE GAME WASN'T FUN.**

ON THE BRINK OF COLLAPSE, WE WERE ABLE TO FIND PRIVATE INVESTMENT TO FINISH THE GAME, THE WAY WE WANTED.

WE HAD A HARD DEADLINE OF **18 MONTHS**. A **PILE OF NEARLY USELESS CODE,** AN **UNFINISHED DESIGN**, AND A **BURNT OUT** TEAM.

NOT THE FUTURE WE EXPECTED

WE STILL HAD A VISION
“NO ONE SURVIVES ALONE”

REGROUP!

WE'RE **TOO SMALL** TO COMPETE WITH TRADITIONAL GAME STUDIOS OF >100 PEOPLE

WHY DID WE **COPY THEIR STRUCTURE?**

FUN
NOT
FEATURES

DEVELOPING GAMES IS **SCIENTIFIC RESEARCH**, NOT MANUFACTURING. **FAILURE IS VALUABLE. FAIL FAST.**

ITERATION -> FUN

HOW CAN WE **ITERATE MORE**, WITH THE **SAME PEOPLE?**

AGILE

RAPID DEVELOPMENT

FREQUENT ITERATIONS

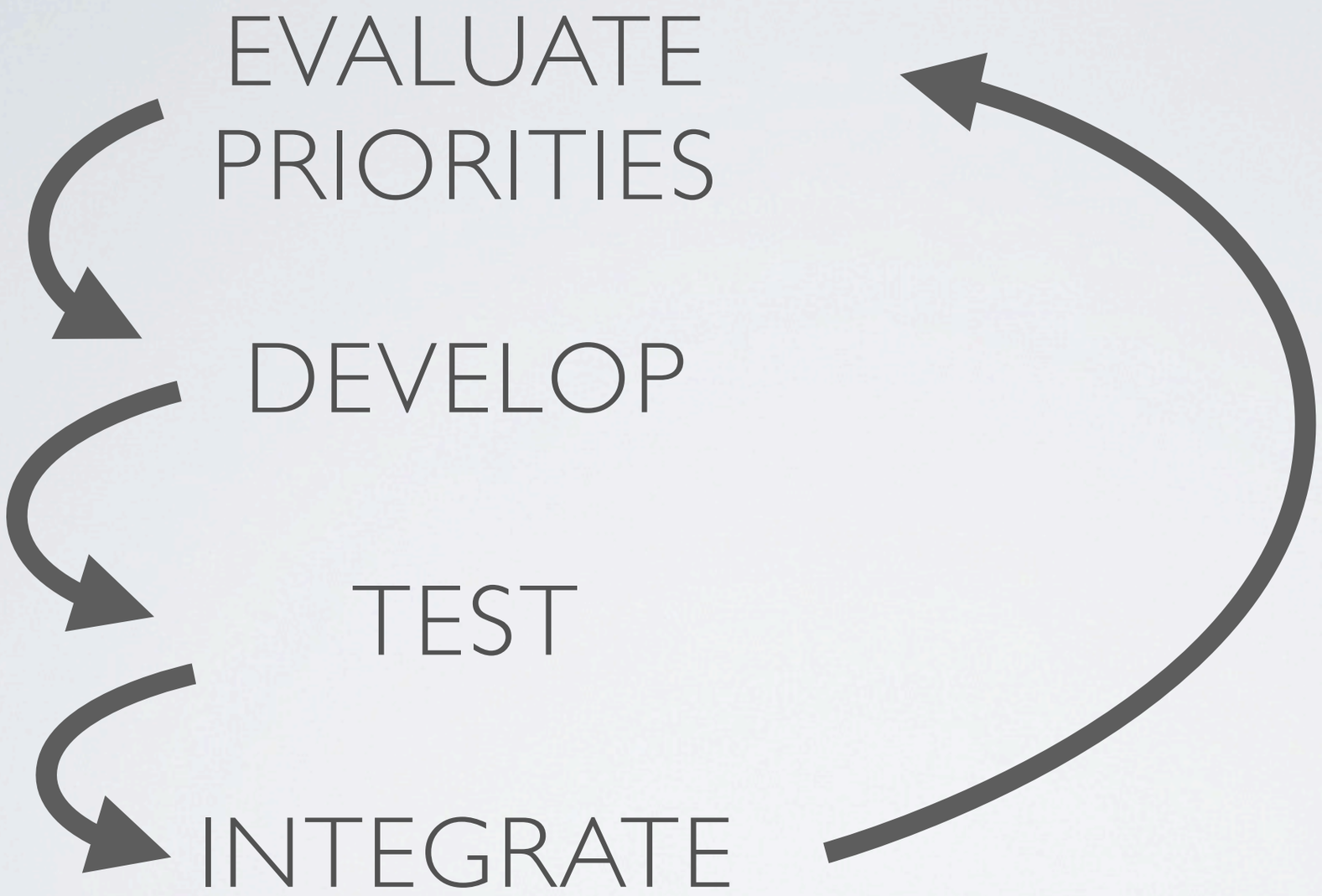
RELEASE READY

CHANGING REQUIREMENTS

SUSTAINABLE PACE

SIMPLICITY

CO-LOCATION



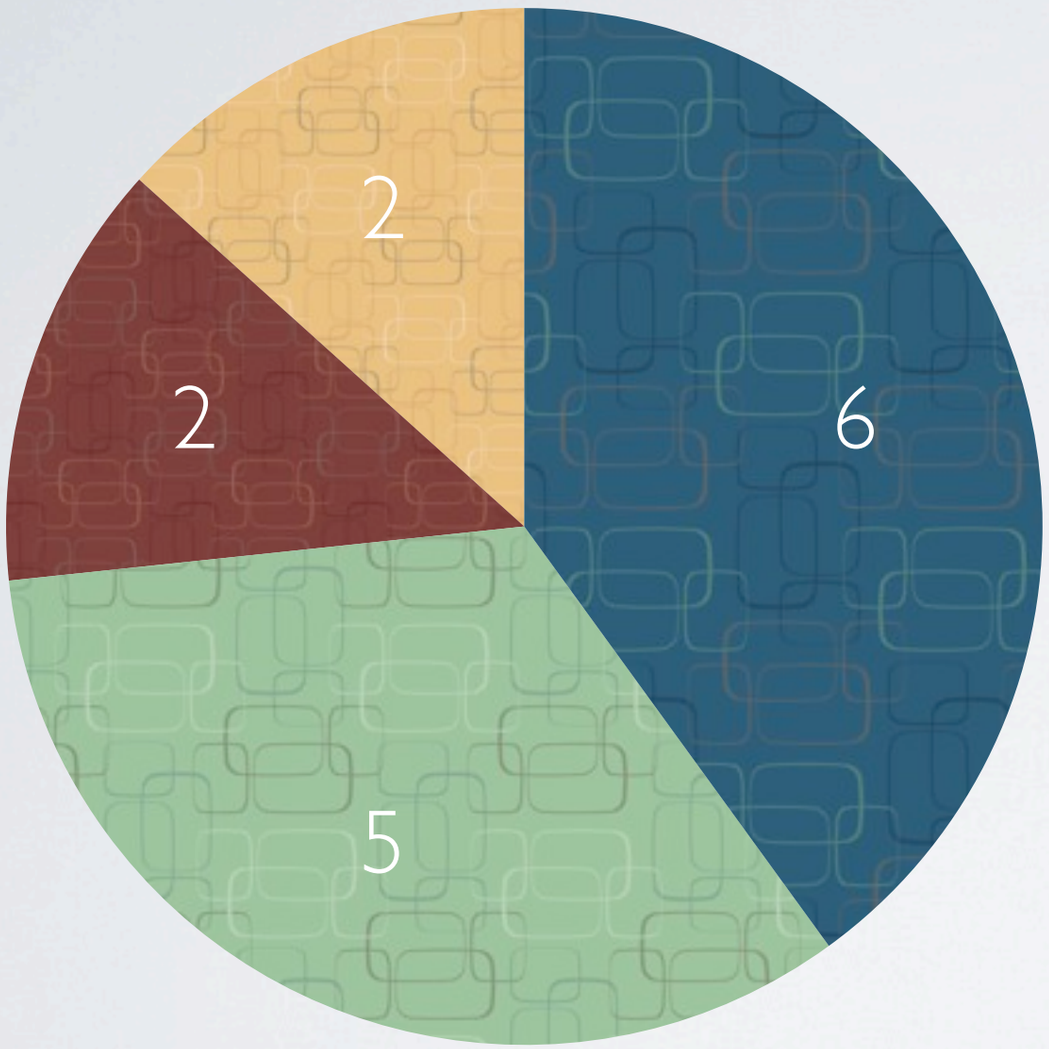
AGILE 2 WEEK *SPRINT* CYCLE

HOW MUCH CAN YOU ACCOMPLISH IN **2 WEEKS?**

PART 3

TEAM & TECHNOLOGY

TEAM COMPOSITION



- Artists
- Engineers
- Designers
- Business

SHIPPED ~3 UNITY GAMES

BEEN WORKING TOGETHER, IN UNITY SINCE 2006, UNITY VERSION 1.6

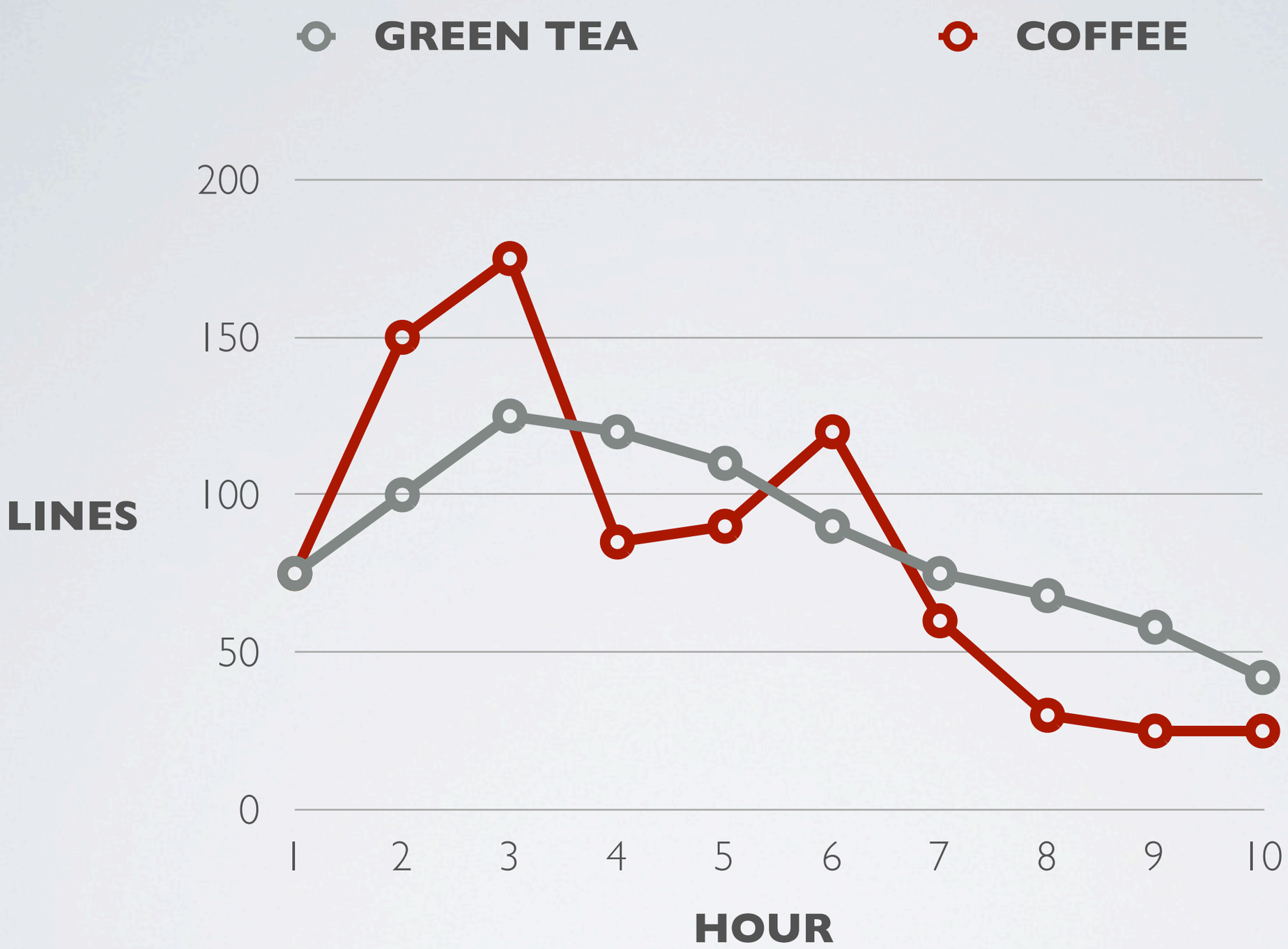
EFFICIENCY

"EFFICIENCY IS A RATIO OF AN EMPLOYEE'S **ACTUAL TIME** TO PERFORM EACH **UNIT OF SERVICE** AGAINST THE **THEORETICAL TIME** NEEDED TO COMPLETE IT."

ARTISTS MAKE **PAINTINGS.**

DESIGNERS MAKE **GOOGLE DOCS.**

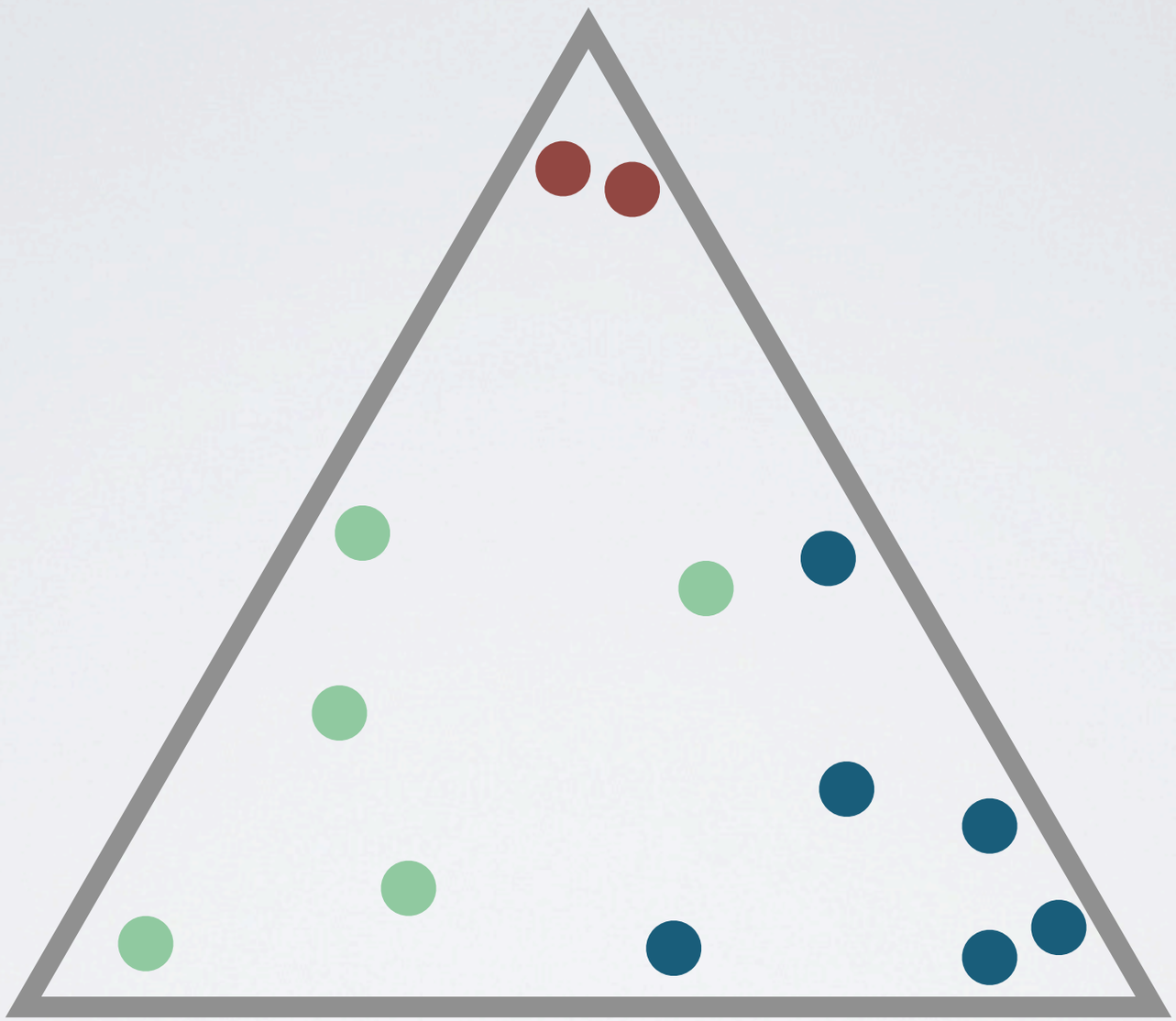
WHAT DO ENGINEERS **MAKE?**



ENGINEERS MAKE CODE

WRONG!

THE NEW **UNIT OF SERVICE** IS A TESTABLE ITERATION.



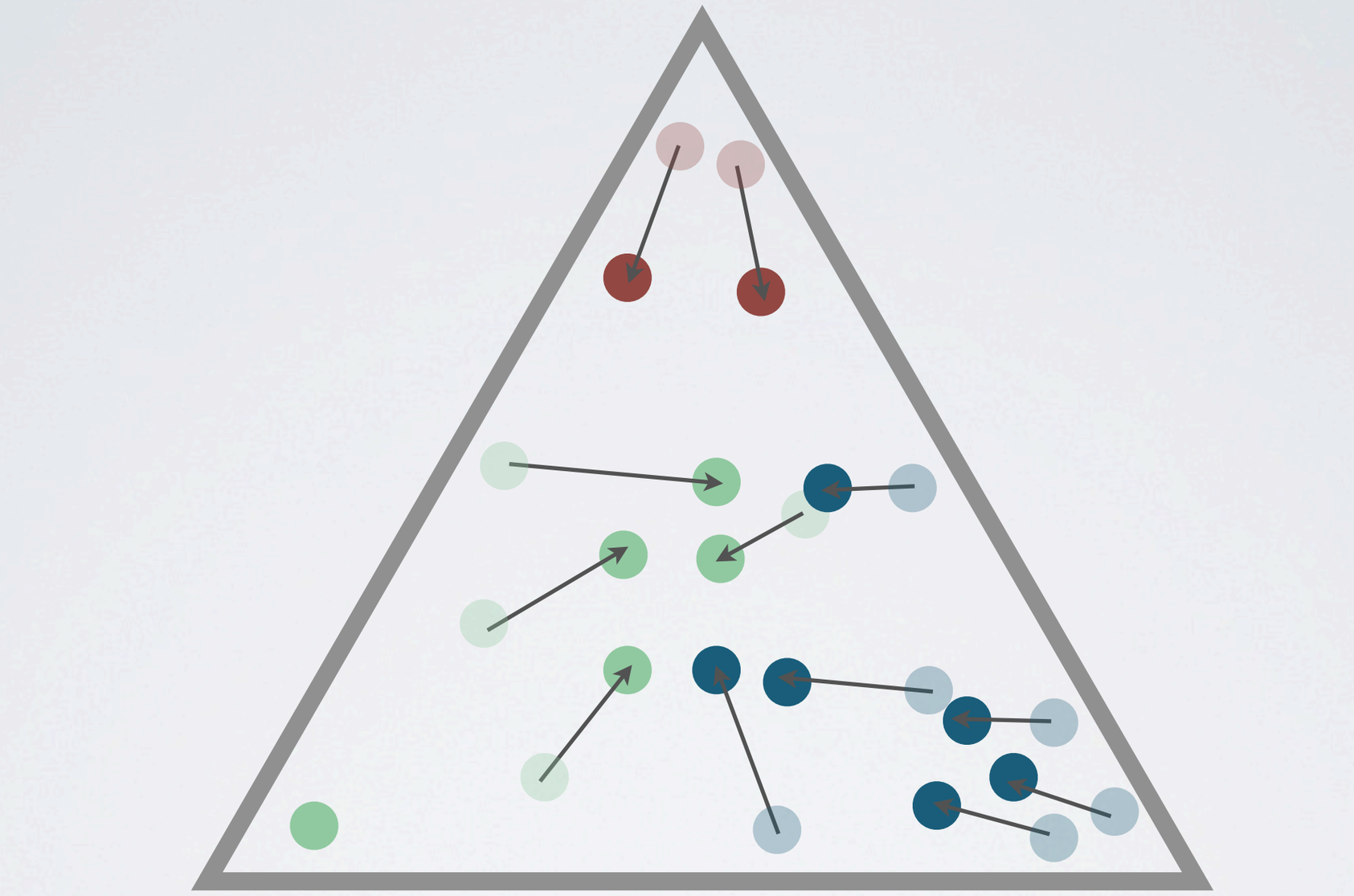
HIGHLY CROSS-FUNCTIONAL

UNITY ENABLES MINGLING OF SKILLS, MAKES TECHNOLOGY ACCESSIBLE

WHY?

SMALL, HIGHLY CROSS-FUNCTIONAL TEAMS
ARE LINEARLY LESS EFFICIENT
THAN HIGHLY SPECIALIZED TEAMS, BUT
THEY CAN **ITERATE**
EXPONENTIALLY FASTER

WHY? **RESOURCE BOTTLENECKING**.
IF ONE PERSON COULD HOLD UP AN ITERATION CYCLE, YOU HAVE A SPECIALIZATION PROBLEM.



CO-LOCATED IN NYC

THE BENEFITS OF SHARED LEARNING ARE WORTH THE COST

1. EVERYONE MUST USE VERSION CONTROL

THEN: SVN 1.7 + CORNERSTONE + TORTOISE

NOW: GIT + SOURCETREE

ARTISTS MAY DISCOVER HOW AWESOME IT IS, AND WANT TO USE IT FOR SOURCE FILES. THAT'S GREAT! MAKE ANOTHER REPO JUST FOR THEM.

2. ARTISTS INTEGRATE THEIR OWN ASSETS

SOUND, ANIMATION, MODELS, TEXTURES, PARTICLES

IN UNITY, EVERYONE COULD **RUN THE GAME FROM THE EDITOR**, AND WE COULD ALL MAKE BUILDS. ITERATION TIME IS NOW ONLY A FEW MINUTES.

USE `ScriptableObject` AND `CustomEditor` TO CREATE DATA DRIVEN WRAPPERS FOR ARTIST FRIENDLY CONFIGURATION AND VERSIONING

3. DAILY SCRUM

CAPPED 10 MINUTE STAND UP MEETING

- YESTERDAY I INTEGRATED THE CHARACTER JUMP ANIMATION
- TODAY I'M ADDING IK TO THE LAND ANIMATION
- BUT I'M BLOCKED UNTIL CHANGES ARE MADE TO THE RIG, WE'LL DISCUSS AFTER SCRUM

SCRUM KEEPS THE ENTIRE TEAM INFORMED AND ENABLES INDIVIDUAL TEAM MEMBERS TO SELF-ORGANIZE

4. DEDICATE TIME TO LEARNING

ALLOWING PEOPLE TO TRY NEW THINGS MAKES THEM BETTER AT WHAT THEY DO

- OUR ARTISTS BECAME PROFICIENT WITH SHURIKEN, AND LEARNED ABOUT PERFORMANCE.
- OUR ENGINEERS LEARNED FROM EACH OTHER - 7 YEARS AGO, I WAS A FILM MAJOR...

IF LEARNING SOUNDS LIKE HIPPY CRAP, YOU PROBABLY SHOULDN'T BE MAKING GAMES.

5. DID I MENTION CO-LOCATION

- COST OF AN ENGINEER IN TAIWAN **$20-30K**
- ITERATION TIME - **24 - 48 HOURS**

- COST OF AN ENGINEER IN NYC **$60-120K**
- ITERATION TIME - **HOURS OR MINUTES**

PROGRESS SHOULD BE MEASURED IN ITERATIONS. DO THE MATH.

BURST CAPACITY

WE'RE GOING FOR HIGH ACCELERATION, INSTEAD OF HIGH TOP SPEED.

PART 4

ARCHITECTS NOT PROGRAMMERS

FINDING A
BALANCE BETWEEN
GOOD **SOFTWARE**
ARCHITECTURE AND
RAPID
PROTOTYPING IS
DIFFICULT.

UI WAS CONSISTENTLY THE **LARGEST SOURCE OF ENGINEERING DEBT** WHILE WORKING IN UNITY.

WE RAN **GAME-JAMS** ON NEW MECHANICS AS GREY BOX PROTOTYPES IN **SEPARATE UNITY PROJECTS**.

QUICK AND DIRTY, AND THE CODE NEVER SAW PRODUCTION.

OUR UI WAS **COMPLEX**, CHANGED FREQUENTLY, RELIED ON **SERVER-SIDE** STATE, AND HAD TO BE **PERFORMANT**.

WE WROTE A COMPLETELY **CUSTOM UI LIBRARY**, AND IT HELPED, TO AN EXTENT.

STILL, SOME OF THE CLASSES IN PRODUCTION ARE **COMPLETELY UNMAINTAINABLE**.

IDENTIFY AREAS FOR **FREQUENT ITERATION** EARLY, AND BUILD **ARCHITECTURE** TO SUPPORT THAT.

WE SWITCHED OUR SERVER FROM JAVA TO C#, AND WROTE A **SCENEGRAPH**, **MATH LIBRARY**, AND **PHYSX WRAPPER** TO MIMIC UNITY'S API.

IT COST **3 MONTHS**.

BUT AFTERWARD, OUR CLIENT CODE **PORTED DIRECTLY** TO THE SERVER, AND OUR ENGINEERS COULD **STAY IN A SINGLE LANGUAGE**.

WE WERE ABLE TO PROTOTYPE SERVER-SIDE LOGIC INSIDE UNITY, AND SHARE LIBRARY CODE.

DATA DRIVEN DESIGN

OUR **ARTISTS AND DESIGNERS** COULD **CONFIGURE EVERYTHING**
IN THE **UNITY EDITOR**, INCLUDING **DATABASE OBJECTS**

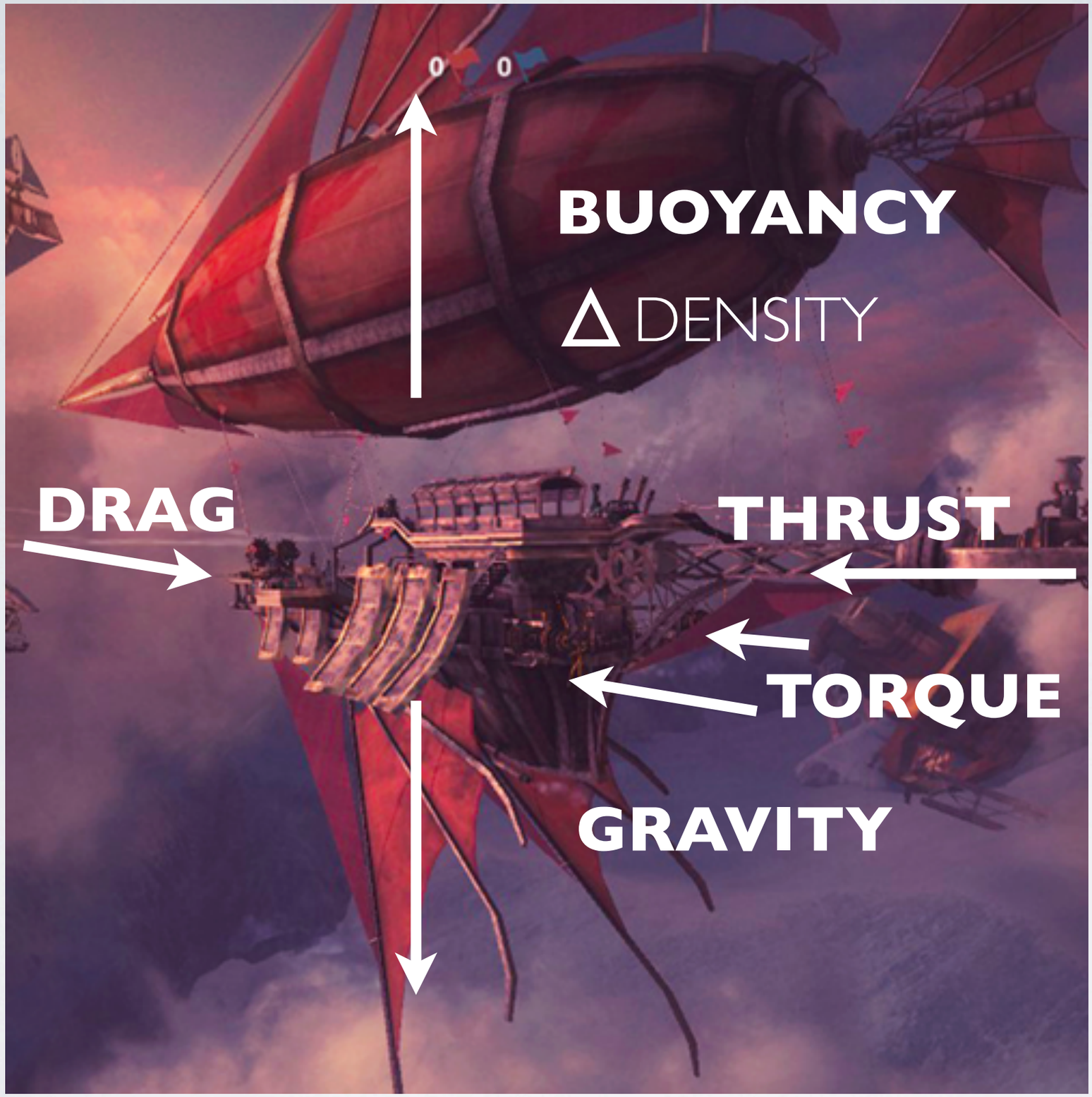
WEAPONS

DIRECT DAMAGE
AOE DAMAGE
DAMAGE TYPE(S)
COMBUSTION CHANCE
SHELL GRAVITY
MUZZLE VELOCITY
RATE OF FIRE
RELOAD TIME
CLIP SIZE
WEAPON SCOPE
WEAPON JITTER
IMPACT FORCE

OUR **CUSTOM EDITOR TOOLS** ALLOWED OUR DESIGNERS TO PUSH BALANCE CHANGES **IMMEDIATELY**, ON THEIR OWN, IN A SIMPLE UI.

PHYSICALLY BASED SYSTEMS

IT'S IMPORTANT TO DEFINE **MEANINGFUL VALUES**, AND AS FEW VARIABLES AS POSSIBLE. **MAGIC NUMBERS SLOW DOWN DESIGN**.



USING REAL PHYSICS ADDS **GAMEPLAY DEPTH** WITH **SIMPLE DESIGN**.
THE PLAYER CONTROLS **BALLONETS**, AND **ENGINE THROTTLE**

ENGINEERS ARE THE **SPECIAL UNIT** THAT MAKE THE REST OF THE SQUAD **FASTER.**

ARTISTS LOVE **CREATING CONTENT.** THEY PERFORM THE SAME TASK OVER AND OVER AGAIN.

UNITY FACILITATES THE **COMMUNICATION** BETWEEN ENGINEERS AND ARTISTS.

ONLY THE
LIBRARIES WE
NEED

WE KEPT LIBRARIES TO A MINIMUM. THEY EACH COME WITH SOME OVERHEAD AND GOTCHAS - BUT CAN **SAVE YEARS** OF DEV TIME.

PART 5

STRATEGY & TACTICS

WE REDEFINED OUR CORE EXPERIENCE AS A

COMPETITIVE MULTIPLAYER GAME, FEATURING **COOPERATIVE COLLABORATION**

WE CUT OVER **2/3** OF THE FEATURES FROM THE GAME INCREMENTALLY BY PRIORITIZING **CORE EXPERIENCE.**

THERE WAS **NO PANIC.**

OKAY, THERE WAS *LESS* PANIC...

WE UTILIZED **UNITY'S STRENGTHS** TO ADAPT QUICKLY - RAPID PROTOTYPING, ASSET INTEGRATION, AND EDITOR FRIENDLINESS.

BUT **WAS IT FUN?**

OUR PLAYERS SEEM TO THINK SO.

OPTIMIZE FOR ITERATION!

THANKS!

twitter: @birdimus

EPILOGUE

PAINFUL LESSONS MASHUP

FLOATING POINT PRECISION AS YOU LEAVE THE ORIGIN

+

SHURIKEN / LEGACY PARTICLE SYSTEMS

"OH I'LL JUST RECENTER THE ORIGIN..." F***

UDP NETWORKED DATA EVERY 20MS

+

THE **GARBAGE** COLLECTOR

F***! F***! F***!

NETWORKED & INTERPOLATED **PLAYER**,
FIRING A NETWORKED & INTERPOLATED
GUN, ON A NETWORKED & INTERPOLATED
SHIP

+

A NETWORKED & INTERPOLATED **TARGET**

"HOW DID WE ALL APPROVE THIS AT THE DESIGN PHASE?"

F***!

THE **TERRAIN** SYSTEM

+

USER CONTROLLED VARIABLE **ANGLE OF VIEW**

“WTF?!“
“OH, THEY OPTIMIZED IT *THAT* WAY”
“CAN WE LOCK THE TESSELLATION?”

...

...

F***!

HUNDREDS OF UNRESOLVED **EDITOR WARNINGS**

+

A DEEP **CLASS HIERARCHY**

"OH, THIS WASN'T SET TO **OVERRIDE**..."
"THAT CODE ISN'T BEING CALLED."
"WAIT, THAT CODE HAS ***NEVER*** BEEN CALLED..."
"HOW DID THIS ***EVER*** WORK?"

OH... F***...

OUR MOST COMMON BUG

```
public class Turret{

  void OnEnable{
    //startup stuff
    ResourceManager.OnResourceFound +=
LoadResource;
  }
  void LoadResource(){
    //load lots of large textures
  }
}
```

"OH, ALEX, LAST WEEK I ADDED THE TURRET CLASS TO THE **OBJECT POOL**"

F*!