Caitie McCaffrey
Distributed Systems Engineer

@Caitie
CaitieM.com
• Halo Services Overview
• Architectural Challenges
• Orleans Basics
• Tales From Production
Presence
Statistics
Title Files
Cheat Detection
User Generated Content
Halo:CE - 6.43 million
Halo 2 - 8.49 million
Halo 3 - 11.87 million
Halo 3: ODST - 6.22 million
Halo Reach - 9.52 million
Day One

$220 million in sales

1 million players online
Week One
$300 million in sales

4 million players online

31.4 million hours
Overall
11.6 million players
1.5 billion games
270 million hours
Architectural Challenges
Load Patterns
Azure Worker Roles
Azure Table
Azure Blob
Azure Service Bus
Always Available
Low Latency & High Concurrency
Stateless 3 Tier Architecture
Latency Issues
Add A Cache
Concurrency Issues
Data Locality
The Actor Model

A framework & basis for reasoning about concurrency

A Universal Modular Actor Formalism for Artificial Intelligence
Carl Hewitt, Peter Bishop, Richard Steiger (1973)
Send A Message
Create a New Actor
Change Internal
State-full Services
Orleans: Distributed Virtual Actors for Programmability and Scalability

Philip A. Bernstein, Sergey Bykov, Alan Geller, Gabriel Kliot, Jorgen Thelin

eXtreme Computing Group MSR
“Orleans is a runtime and programming model for building distributed systems, based on the actor model”
Virtual Actors

“An Orleans actor always exists, virtually. It cannot be explicitly created or destroyed”
Virtual Actors

• Perpetual Existence
• Automatic Instantiation
• Location Transparency
• Automatic Scale out
Runtime

- Messaging
- Hosting
- Execution
namespace HelloWorldInterfaces
{
    /// <summary>
    /// Orleans grain communication interface IHello
    /// </summary>
    public interface IHello : Orleans.IGrain
    {
        Task<string> SayHello();
        Task<string> SayGoodbye();
    }
}
Reliability

“Orleans manages all aspects of reliability automatically”
Performance & Scalability
“Orleans applications run at very high CPU Utilization. We have run load tests with full saturation of 25 servers for many days at 90%+ CPU utilization without any instability”
Figure 6: Throughput of Halo 4 Presence service. Linear scalability as number of server increases.
Load Patterns
Orleans is AP
• Statefull Services
• Virtual Actor Abstraction
• Self Healing Frameworks
Orleans & Halo
Get Orleans

https://github.com/dotnet/orleans
Tales From Production
DevOps

noun

1. The Decisions You Make Now Will Affect the Quality of Sleep You Get Later
Load Patterns
Story: No Data Like Prod Data

aka Halo 4 launch night was not the first time Azure & Orleans saw Production Data
New Technology

• Orleans: MSR Technology
• Azure
• Dispatcher
Halo Reach:
Presence Service
Memory Leak
Practice DevOps
Story: Validate Dependencies

aka the time we broke Azure Service Bus
STOP WHAT YOU’RE DOING!!!!
WHAT WERE YOU DOING???
"Who owns my availability" as a service.

whoownsmyavailability.com

(You're a dumdum if you put all the blame for an outage on your provider.)
YOU

Recommended reading from selected authors:

Human Error, by James Reason

This reminder is brought to you by @jmhodges and @tmn.
Backup the Backup
Story: Clients are Jerks

aka remember that time the game DOS’d us at Launch
Different Priorities
Release Valves
Back Pressure
Protect Your Services
Let’s Wrap it Up
Distributed Systems is hard
CAP Theorem

aka why we can’t have nice things
Know You’re Tradeoffs

hint: you are making one whether you know it or not
Consistency or Availability
Questions

@Caitie