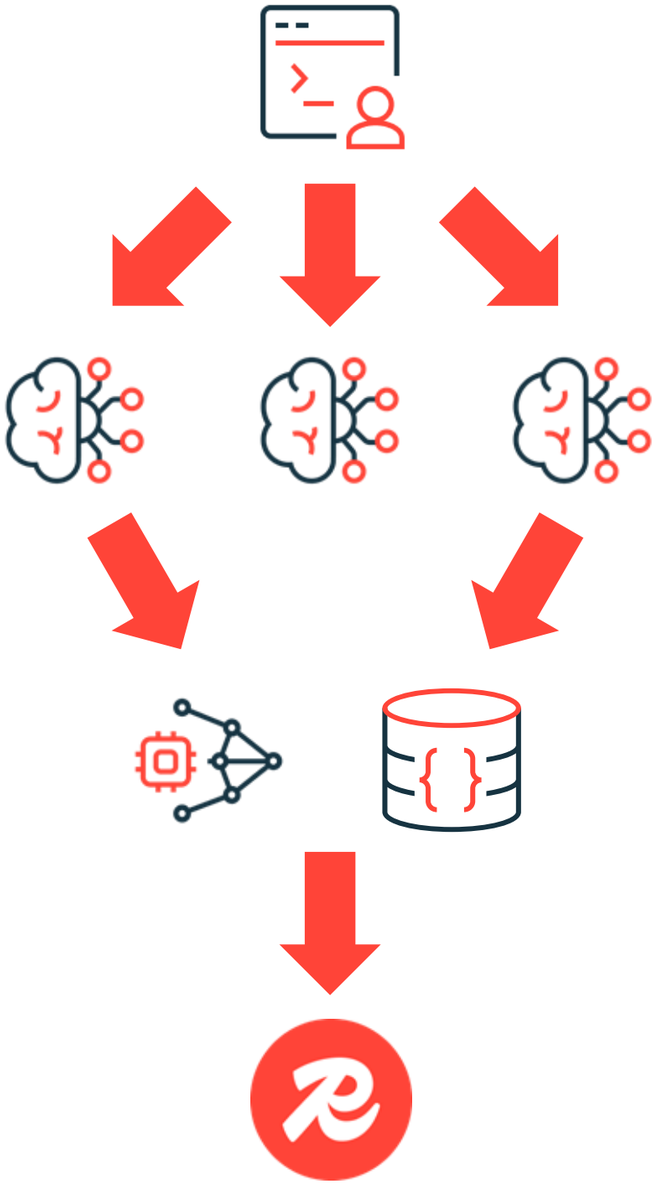
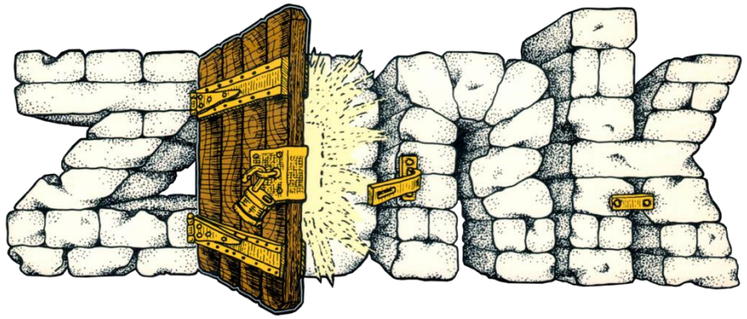
Multi-Agent AI Text Adventure Game with Agent Memory & Azure

Guy Royse
January 22nd, 2026

What is it?

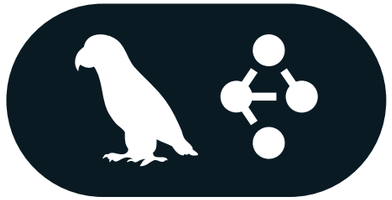
- It's a text-based AI-powered game inspired by the classic retro game Zork
- It takes user actions and delegates them to AI agents that represent stateful objects in the game world
- It remembers conversation history so you can speak to it naturally
- It stores and queries persistent game state using JSON
- It stores it all in Redis





Objectives

LangGraph.js
+
Agent Memory Service
+
Azure Managed Redis



LangGraph.js

- Accepts player input
- Routes player actions to correct agents
- Saves recommend changes from the agents to the game state
- Responds to the player

I open the mailbox and take what is in there.

I'm now open.

The player now has me.

JSON.SET

You open the mailbox and take out a neatly folder leaflet.

Agent Memory Server

- Stores short-term conversation history
- Summarizes older conversation history
- Extracts long-term memories for use in future conversations
- Generates prompts for your LLM to use

I look at the house.

The house is white. The front door is boarded shut.

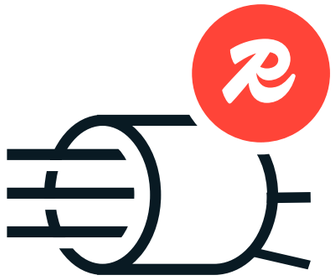
I walk around the house and look for an open window. I like climbing in windows.

Player looked at the house and walked around to look for an open window to climb into.

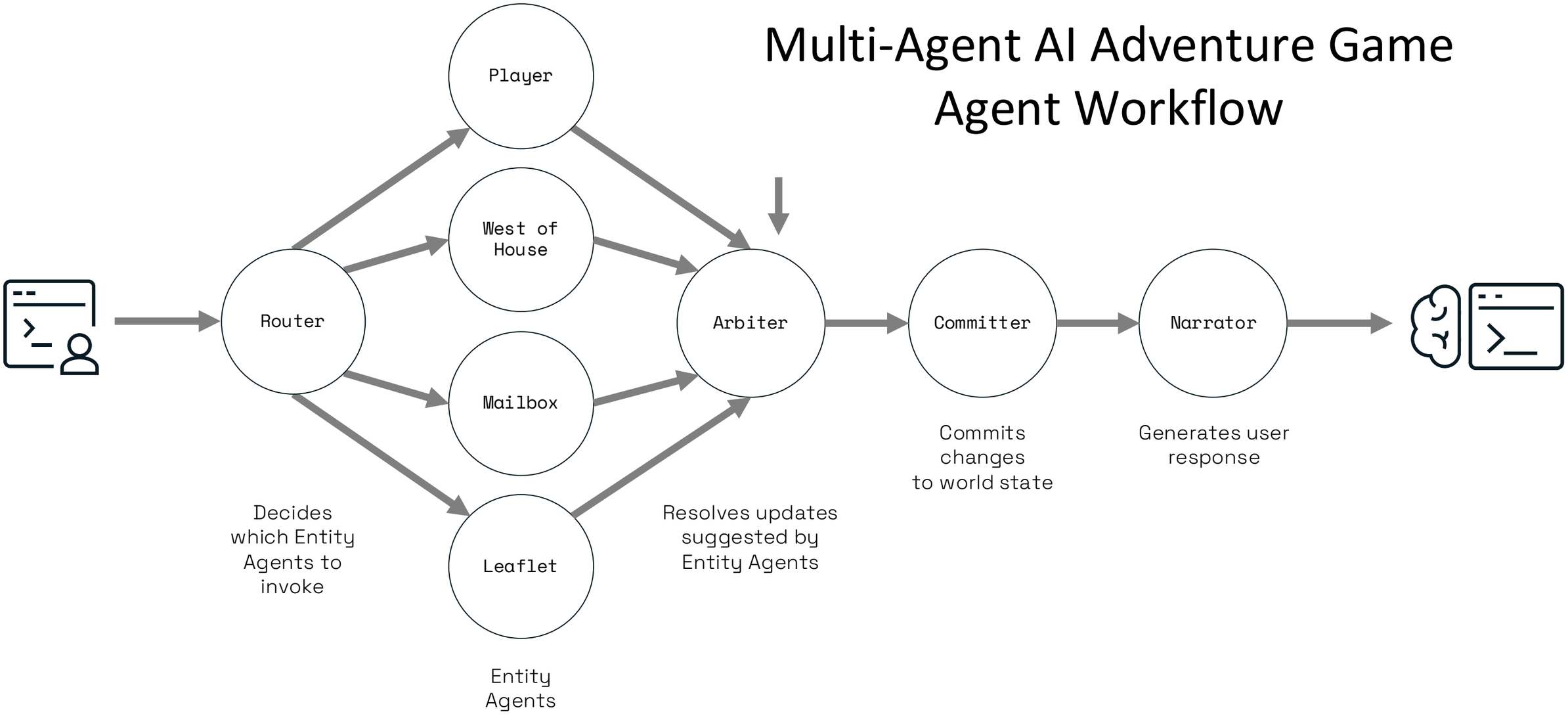
Player likes climbing in windows.

Azure Managed Redis

- Stores memories for Agent Memory Server
- Stores game state
- Stores game history



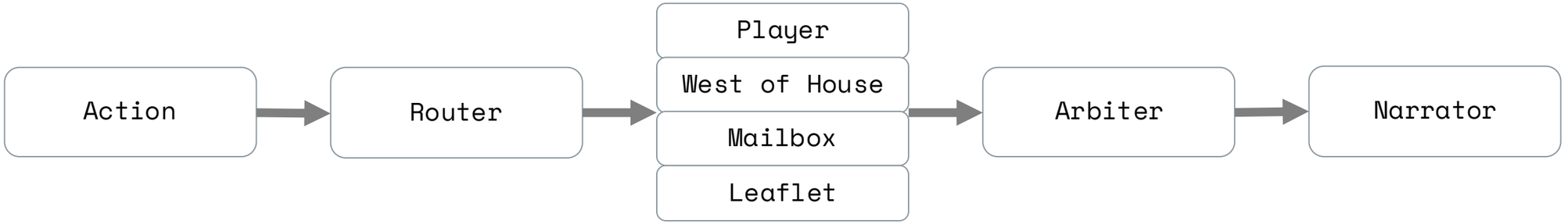
Multi-Agent AI Adventure Game
Agent Workflow



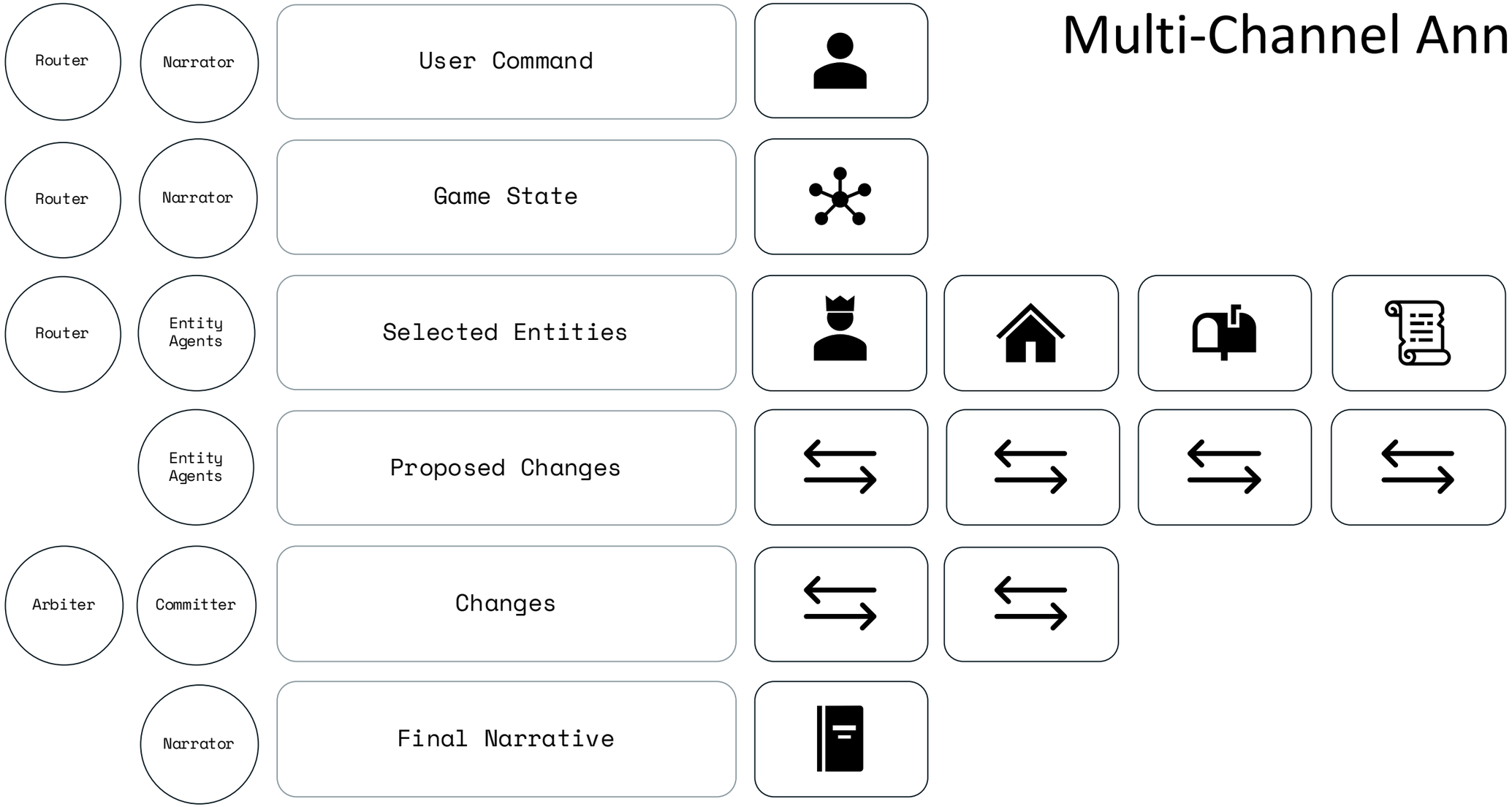
Redis

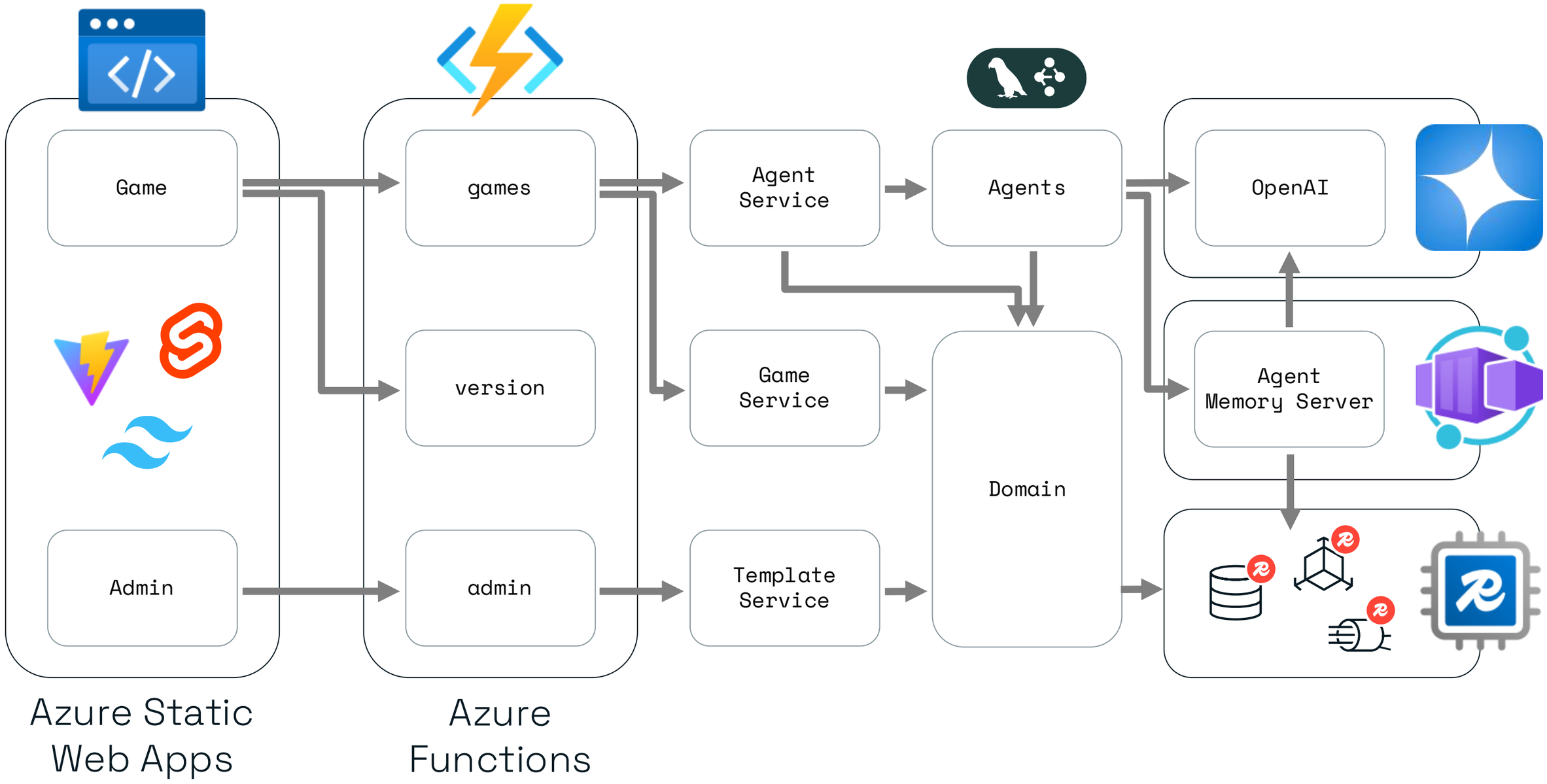
Typical Workflow State

Message Annotation



Multi-Channel Annotation





Multi-Agent AI Adventure Game Architecture

Code & Demo

What was learned along the way

- Context in multi-agent systems can grow rapidly
  - The default message annotation in LangGraph.js exacerbated it
  - If you want to manage context effectively with LangGraph.js you need to use multi-channel state
  - This pairing makes context engineering much easier
- There were a lot of system prompts for all the agents
  - Writing these to be specific is a challenge
  - Agents tended to be overly enthusiastic about changes
- Overall and excellent pattern to learn from that is applicable beyond just gaming

What we could change

- Agent Memory Server only gets invoked in the narrator
  - It should be invoked at the start of the workflow
  - Then all agents would have access to conversation history, memories, and summaries instead of just the final node

- More Entity Agents
  - We only implemented Player, Location, Fixture, and Exit
  - Could add Items which would force more interaction of game state
    - The key unlocks the door or the torch illuminates the room
  - Could add NPCs which would create crunchier game state
    - The sword does 1d6 damage or the ogre has 17 hit points

Thank you.