Intelligent Agents - Prior Authorization Model

Kevin Bradt, Carson DeSotel, Ryan Harvey, Kailash Kalyanasundaram

Summer 2021

Contents

---

1 Product Description

Prior Authorization

Prior Authorization is a complicated process where healthcare providers must attain advance approval for a specific service based on a patient's health plan. The process involves extensive negotiations back and forth between the healthcare provider and the patient. The negotiations often include tedious paperwork and extremely particular information, which cost patients, nurses, and physicians valuable time. The patient needs to constantly work with nurses, physicians, and their healthcare provider to make sure every single step of the prior authorization process is completed and approved before the patient can finally obtain their required service or treatment.

Motivations

The current prior authorization process is a lengthy, costly, and inefficient process. The steps involved in prior auth. are mostly manual and involve filling out forms and data by hand. According to the *2020 AMA Prior Authorization (PA) Physician Survey*, physicians and staff spend 16 hours/week completing PAs on average. This time takes away from their abilities to be serving their patients to the fullest extent. Additionally, prior auth. delays can affect patients as the same survey found 30% of physicians report PA has led to an adverse event for a patient in their care.

This product is being created to save time and allow physicians and others involved in the laborious task known as prior auth. the ability to do their **life's best work**™.

Final Product

The final product for this project will be a local demo that demonstrates how Intelligent Agents can be applied to automating the prior authorization process. The goals for this demo are:

1. to **argue the plausibility** of a distributed intelligent solution to prior auth.
2. to **showcase the capabilities** of automated negotiations in a multi-agent system.

The demo will consist of interactions between a manager agent and worker agents that represent decisions in the prior auth. process, namely eligibility, provider, level of care, medical necessity, and service decisions. The information flows between these agents have been outlined in Fig. 1 on page X. Additionally, we will represent the medical facility as an agent, however, this solution doesn't rely on that agent's intelligence, and can instead be thought of as a data stream to the manager agent.

The product will lack scalability and application outside of demonstration purposes. The intricacies of prior auth. have been simplified and will not be processing real prior auth. requests. The demo will exist within one container, which will make it portable, but a true system would require further containerization.

2 Innovation

Current Systems

Current prior auth. solution attempts have failed, or been inconclusive to solving the process due to attempting a **monolithic approach to a distributed problem**. Searching for "prior auth automation" will result in either software that doesn't denote its approach to automation, or lists that reveal what you *should do* and not what it can do for you. This demo aims to provide evidence that this solution is *viable*, *novel*, and *worth it*.

Distributed Approach: Intelligent Agents

The prior auth. system is, by nature, a distributed process. Forms are filled out by different individuals, authorized by others and so on. By using a distributed approach, we can mimic the current, manual prior auth. process through intelligent agents. The key to the distributed approach is mimicking a team structure, or organizational hierarchy which allows one agent to coordinate several other agents' interactions. Each worker agent under the manager is devoted to a single task. This is an **expandable** approach as each worker can focus on a custom microservice, and the manager can add another rules layer to interface with any additional agents.

3 Tech Stack

Java Agent Development Framework (JADE)

JADE is a framework used to create multi-agent systems in Java. These agents are by default, unintelligent. The power of using JADE comes in the way that the backend server handles interactions between the agents. It provides a way of creating agents, which can be thought of as threads with asynchronous event handling, in a way that minimizes the way the developer has to reason about the system. In this system, JADE is used to define the behavior of agents and handle coordination between the agents. The reasoning capabilities are a result of the integration with the Drools Rule Engine.

Drools Rule Engine

The Drools Rule Engine provides an expert system in the multi-agent model. The model takes advantage of Drools' forward chaining capabilities to control agent actions based on facts in the system. Drools enhances JADE agents by giving them complex reasoning abilities. The agents use this when it comes to message handling and data processing. These are necessary steps for the agents to negotiate and form conclusions regarding the present information.

Microservices

Through the use of microservices as endpoints in our agents, we can promote an expandable and modern approach to the development of a full application. For developers and development teams, this means a more focused and more testable approach to developing the jobs an agent can do. This helps to enforce the object-oriented development approach to be done throughout the project. This will also allow new systems to be created and installed without needing to interact with the agent framework itself. This would be impactful on a team, as developers could focus on creating Java microservices without needing to fully understand the implementation of the Drools Rule Engine or JADE.

4 Model of Prior Authorization Process

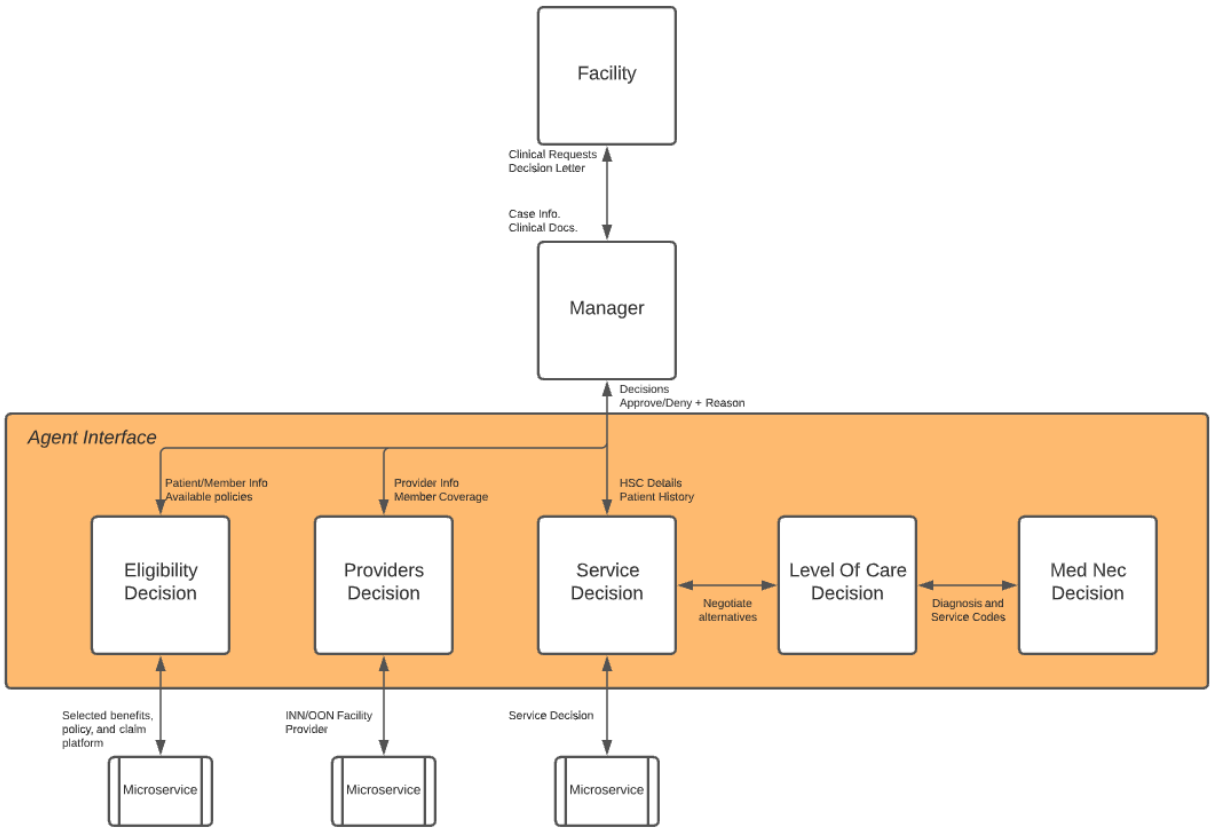

Figure 1: Diagram of the Multi-Agent Prior Auth. Process

Agent Interactions

The model relies on the work of a manager agent that handles incoming information from the medical facility. This information may include the initial prior auth. notification, case information, and clinical documents. It is the manager agent's task to determine what information is necessary for each agent and distribute the information accordingly. The three agents the manager can pass on information to are the eligibility, provider, and service decision agents. These agents will negotiate with the manager to obtain missing information. Additionally, these agents will report back the result of their decision for the manager to send an aggregate decision back to the facility.

In this demo, the worker agents are individual agents, however, a final application would likely have a network of micro-services for each decision process. The model provides opportunity for this expansion where the current agents can be added to without severing the communication link with the manager agent and the rest of the decision network. It is beyond the scope of this project to implement these complete networks that would handle the complex processing needed to perform all the steps of the prior auth. request process.

Use Case for Demo

The example case will demo a prior auth. request for an OP infusion-injection service for a member on a Medicare plan. This example has been chosen specifically since it is a common case that represents a significant amount of prior auth. requests.

Agent I/O

Eligibility decision will gather patient information and service description from the manager. This will be used to lookup available policies for the member and determine if the member is covered for this service. The agent will return to the manager with the selected policy used for the prior auth.

Providers decision will gather patient information, location, and health plan from the manager. The agent will select the physician and facility/org that will perform the service along with whether that provider is in network or out of network.

Service decision will gather patient history, ICD-10-CM diagnosis codes, and CPT service codes from the manager. The agent will approve the patient's required service based on the **Level of Care** and the **Med Nec decision**. The approved service codes will be returned to the manager agent for final decision.