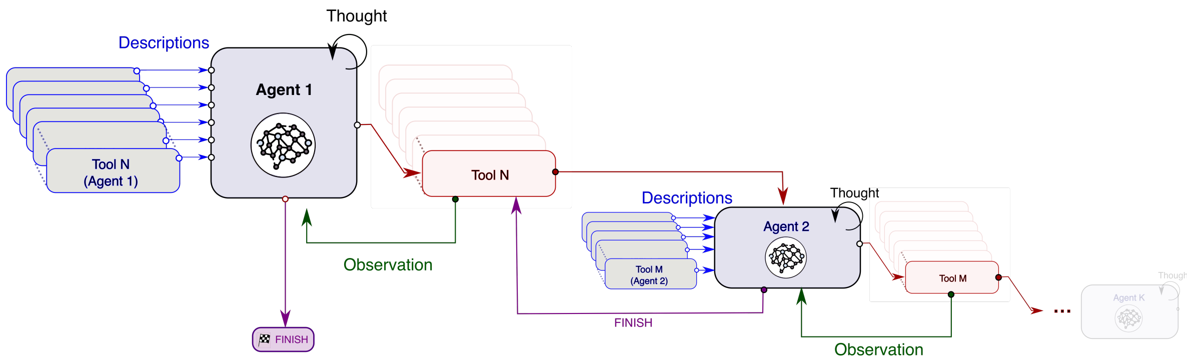

Figure 2: **LLM-based agent architecture.** Each agent is comprised of an LLM prompt template, a collection of tools, a variable that summarizes the results of previous reasoning steps of the agent, and a parser capable of interpreting the outputs of the LLM. Each tool is comprised of a single function where the inputs and outputs are both strings and an accompanying textual description that describes to the agent how to use the tool. The hierarchical nature of the system arises from the fact that some of the tools are implemented as more specialized autonomous agents themselves.

complex commands [6]. They are not capable of inferring the users' goals and acting accordingly.

In an attempt to overcome some of these challenges, recent work proposed to leverage the reasoning capabilities of LLMs to better understand and carry out user commands. In particular, Sasha [7] introduced the use of LLMs in smart home environments and showcased that the LLMs can be used to produce reasonable behaviors in response to complex or vague commands. Sasha implements a decision making pipeline where each step (such as selecting the device to use, or checking if a routine already exists) is implemented using an LLM. However, unlike SAGE, the stages of the Sasha pipeline are manually defined and fixed, limiting its flexibility. Today's smart home users also have the option of defining IFTTT-style routines, which connect trigger conditions to actions [8], in order to create complex behaviors. This approach is inconvenient because it must be implemented by users manually through an app or web interface. It is also limited by the fact that trigger conditions must be defined by device manufacturers. Users can also write their own apps to manage device states, such as SmartThings' SmartApps [9], but this requires a level of technical sophistication beyond the ability of most users, as well as a significant time investment. Web-based services such as IFTTT[1], Zapier[2], and Home Assistant [3] enable the user to create rules to control their smart devices. The advantage of these services is that they simplify the process of connecting various services and smart devices without the need for extensive programming knowledge. However, these solutions also lack reasoning and context-awareness offered by LLMs. Recently, IFTTT has begun to leverage the power of LLMs through the creation of ChatGPT plugin[4], which provide a more user-friendly and accessible interface to interact with the automation platform conversationally. However, this plugin does not allow GPT to generate new routines, but rather trigger existing ones. Zapier also offers an LLM-based offering, which allows users to describe action in a more natural way, but currently lacks significant reasoning sophistication. [7] created an LLM-based pipeline which could output trigger-action pairs to create simple IFTTT routines, but logical complexity of these routines, and well as the flexbility of the triggers, was limited. Earlier academic work investigated training

[1]https://ifttt.com/
[2]https://zapier.com/
[3]https://www.home-assistant.io/
[4]https://ifttt.com/explore/business/ifttt-ai