Chapter 1

Empirical Study

This chapter presents the experiments done with the different Multi-Agent systems designed in **??** and implemented using the framework explained in **??**. It outlines the experimental setup and design decisions that were taken, to best answer the research question **??** with all its sub-questions.

1.1 Research Questions

The main objective of research question **??** is to assess the different Multi-Agent Ssstems proposed in **??**. Each of the three different sub-questions focuses on a different dimension of the system.

Impact of Role Specialization (??)

"How does role specialization within an LLM-based Multi-Agent system influence its ability to discover business processes through conversations with process participants?"

The sub-question investigates the influence of role specialization within the MAS on the system's ability to reconstruct business processes from user interviews. For this purpose, the discovered and constructed business process in the form of a BPMN is qualitatively assessed similar to the work of Kourani et al. [12].

Impact of System Complexity (??)

"How does the number of agents in an LLM-based Multi-Agent system affect the efficiency of knowledge gathering and process model construction?"

This sub-question explores how increasing the number of different agents affects the overall system efficiency. Efficiency is quantified by the total number of tokens processed, encompassing both input and output, as this directly reflects resource usage and, therefore, correlates with costs for LLM agents. Qualitative analysis of intra-agent communication further examines the effectiveness of the communication. The hypothesis is that increasing specialization will result in more communication and, consequently, higher resource consumption [13].

Impact on Robustness (??)

"How robust are more complex Multi Agent Systems to deal with internal Failures?"

This sub-question considers the failures according to the taxonomy as explained in **??** by Cemri et al. [4]. The failure distribution can be compared to the benchmark also discussed by the authors and can help indicate potential problems and points of improvement. Further, it can be used

to analyze how more complex Multi-Agent systems, systems consisting of more agents, behave compared to simpler ones, and whether they are robust.

1.2 Simulation of Business Process Participants

To maintain experimental control and consistency, all experiments are conducted within a simulated environment, where the interviewed humans are role played by an LLM based agent. Park et al. [16] showed that agents can be used to accurately simulate even complex human social behavior, however in this study a way simpler approach can be used. Throughout this section, these simulated participants are referred to as "human"-agents, indicating that their responses are generated by LLMs rather than real people.

Each "human"-agent is assigned a specific role within the predefined business process and receives detailed instructions outlining their responsibilities, perspectives, and the scope of information they possess. These instructions are tailored to mimic the partial and sometimes subjective viewpoint of individual process participants within a real organization. During the interviews, "human"-agents provide narrative-style responses in the first person, describing tasks, decisions, and contextual details relevant to their part of the process. This design ensures that the interaction reflects typical real-world interviews, where no single participant holds the complete process knowledge.

Simulations play a central role in the study of Multi-Agent systems as Jin et al. [10] point out. They provide a controlled environment where experiments can be repeated under identical conditions, ensuring reproducibility and transparency. This approach allows for systematic variation not only in the configuration of the Multi-Agent system, but also in the choice of language models or other factors under investigation. They also offer a reliable way to observe how agents interact and adapt to their environment, which deepens the understanding of coordination and performance. Ultimately, these environments support the development of more robust and effective Multi-Agent systems for real-world use.

1.3 Business Processes

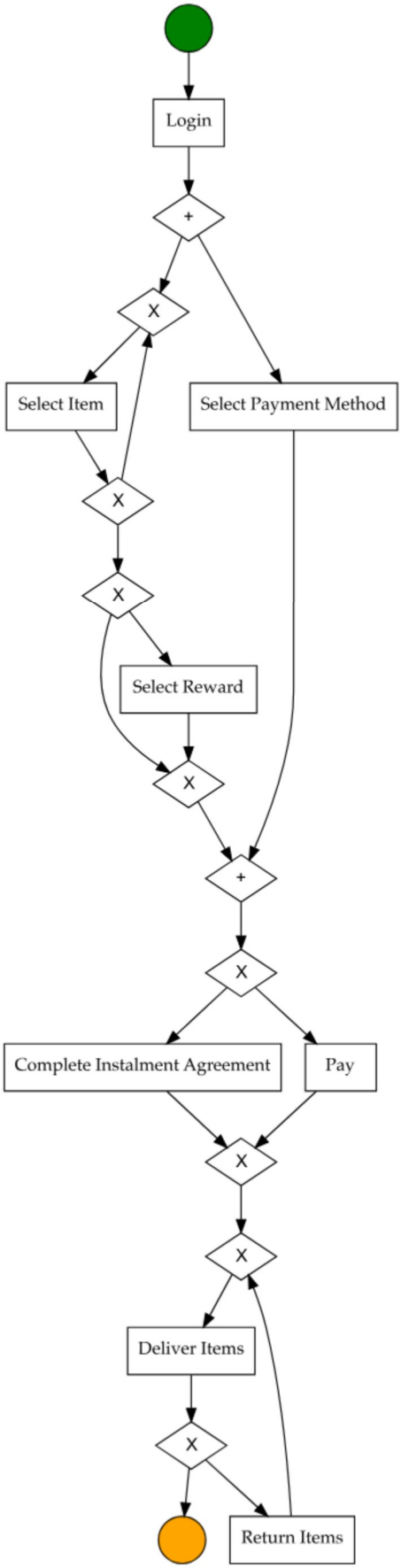
To make sure that the Multi-Agent system is not just performing well in one specific condition, three different scenarios are compared. These scenarios vary in the number of process workers that need to be interviewed and in key features that make them suitable for testing. All scenarios are briefly introduced here; more information on them can be found in **??**.

Online Shop purchase

The shop process, adapted from Kourani et al. [12], describes the flow of a customer through an online shop. Figure 1.1 shows the BPMN for this process. This scenario has six different process workers; the role description for all of them can be found in **??**. The distinctive feature of this business process is that it has a few repeated activities, combined with parallel paths and optional activities.

Reimbursement process

The reimbursement process, adapted from Dumas et al. [7], describes how a company employee can request reimbursement and how the request is evaluated. Figure 1.2 shows the BPMN for this process. This scenario has four different process workers; the role description for all of them can be found in **??**. This process is fairly linear, with only a few decision points that lead to different paths within the process.



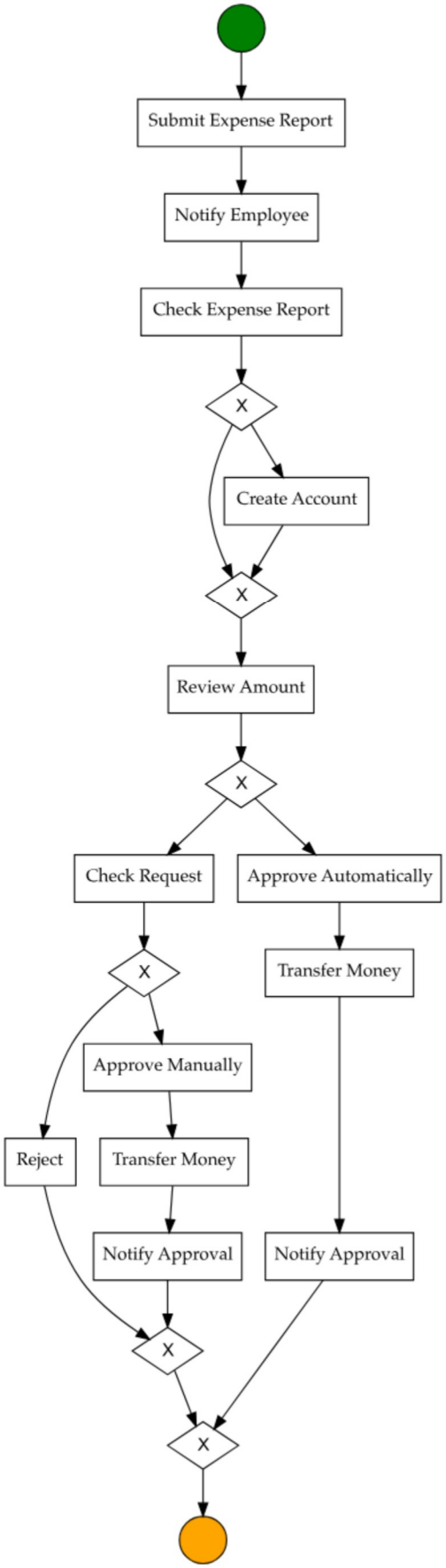


Figure 1.1: (left) ”Purchase in Shop” Process in BPMN

Figure 1.2: (right) ”Reimbursement” Process in BPMN

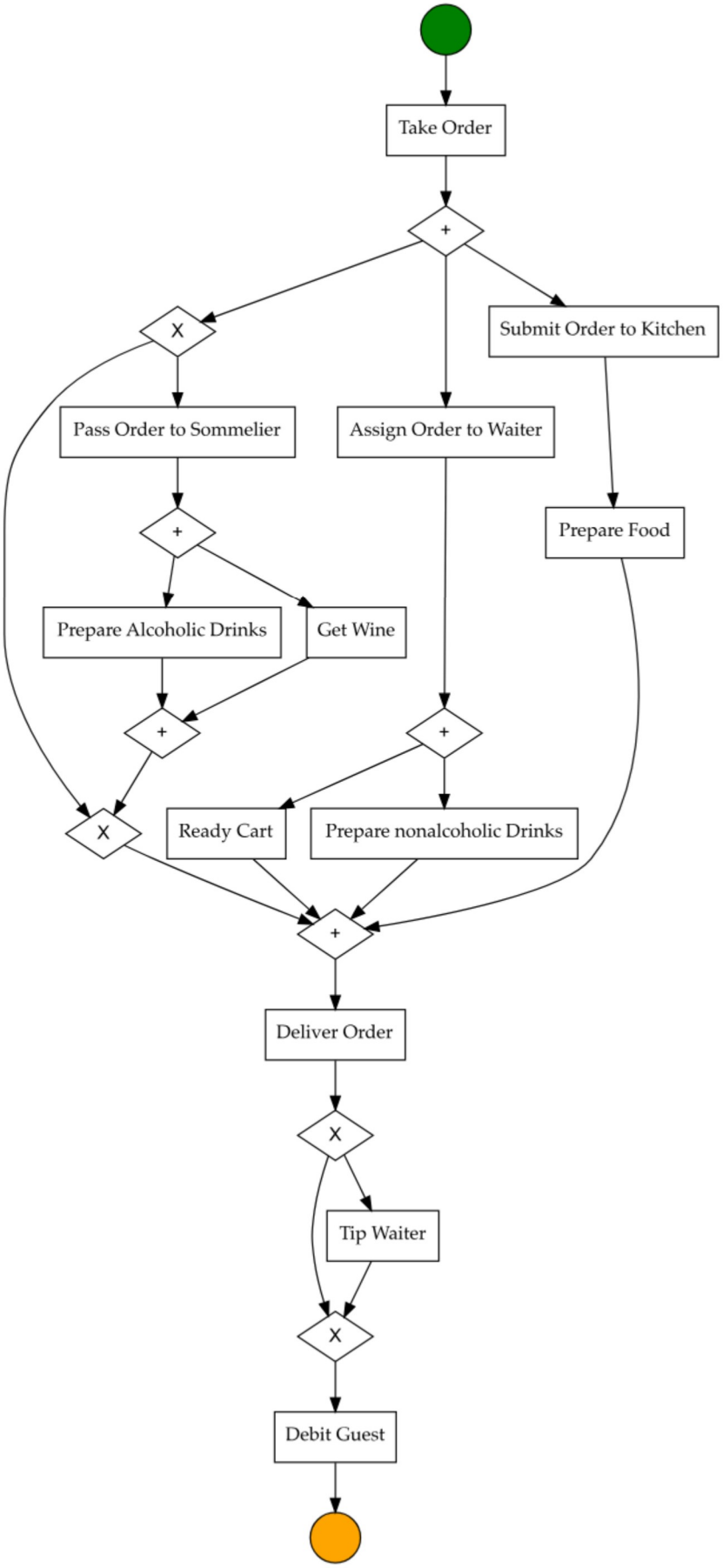


Figure 1.3: ”Order at Hotel” Process in BPMN

Hotel Room Service process

The hotel process, adapted from Kourani et al. [12], describes how a guest at a hotel can order dinner and drinks as room service. Figure 1.3 shows the BPMN for this process. This scenario has five different process workers; the role description for all of them can be found in **??**. The key feature of this process is that it has quite some activities happening in parallel to each other, where the roles themselves are also not fully aware of all the other activities and mainly know how their work joins the main process again.

1.4 Evaluation Metrics

Evaluating the performance of the Multi-Agent system relies on a set of direct and measurable criteria. This section discusses the main metrics used to assess how effectively the agents discover underlying processes through interactions with simulated human participants.

1.4.1 Core Task Success

The primary objective is to reconstruct the original business process, referred to as the ground truth. Since the actual modeling step is based on the work of Kourani et al. [12], assessing the quality of the BPMN itself is done in their work. Therefore the focus is put on evaluating if the different Multi-Agent systems captured the essence of the process in the respective BPMN and proof an understanding of the activities and how they relate to each other, by putting them in the right order or parallel to each other. This is assessed by checking if the process model has certain features that are described in the distributed process knowledge the process workers have. A list of all these features per process can be found in **??**.

The initial idea of also algorithmically rating the similarity of the business process by comparing its result on a token-based replay [1] to that of the ground truth was discarded as the first test experiments showed there were too many naming differences, which made it impossible. And adaptations of the experiments where the agents knew the exact names of the activities to be able to name them correctly skewed the whole results too much as it gave them too much initial understanding of the process[1]. Another problem was, that even if similar names were used, depending on the deviation, even a small one, could lead to the whole similarity being 0, for example if just the order of two activities was swapped. All in all, this metric was therefore considered not indicative of evaluating the performance of the model against the predefined business process. Also in their work, Kourani et al. [12], did not use an algorithmic approach but human judgment to evaluate the output models.

1.4.2 Token Cost

The amount of tokens is directly correlated to the costs of using an LLM-based Multi-Agent system. When using an LLM via an API, the cost depends on the tokens[2], or when hosting it locally, the tokens correlate to the computational load. Token usage is tracked separately for input and output tokens, because commercial LLM services typically differentiate between them; input tokens, or "prompt tokens," are typically cheaper than the generated output token. This research only considers the tokens as the actual price has no long-term relevance[3]. For each agent in the system, token consumption is monitored individually.

[1] *Delivering Items* and *Return Items* are probably in that order, and need no further investigation

[2] See overview on VertexAI for the prices of several different LLMs

[3] An overview analysis from the Venture Capitalist A16Z done by Appenzeller [3] showed the price droppings in recent years. One of the API providers used in this research even offered discounts of 50% when using their services during *off peak* hours

1.4.3 Failure Analysis

As discussed in **??**, Cemri et al. [4] proposed a taxonomy of failure modes in Multi-Agent systems. In their work, they also built an LLM-as-a-Judge that can be used to analyze the traces that are the result of the experiments and give more insights into the failures. As a side product, they also output a *Task Completed* assessment, that indicates if the agent has done what it was supposed to do on an abstract level. This is not the task *"reconstruct the business process"*; it can be more understood as the task that is implied due to the role specification and input from another agent. As this failure taxonomy is designed to be generalizable to all kinds of Multi-Agent setups, it also allows the comparison of these experiments with a benchmark that the authors made in their research as well.

1.5 Selection of LLM Models

The different Multi-Agent Systems should also be tested with different LLMs as the core model in each agent, making the decisions and planning. A comparison of three different chosen models is shown in Table 1.1. The temperature is an important parameter that controls the uncertainty of the output of an LLM [9]. This also influences the reproducibility and therefore some benchmarks like AgentBench use 0 as temperature [14] while MultiAgentBench uses 0.7 [21] to allow for more creativity in the response. Since both aspects are relevant in this context, a temperature of 0.5 was chosen as a compromise to balance reproducibility and creativity, with a slight emphasis on the latter.

	Gemini 2.5 Pro	Mistral Large 2	DeepSeek-V3
Vendor	Google	Mistral AI	DeepSeek-AI
Release Date [4]	25.3.2025	24.7.2024	27.12.2024
Architecture	sparse MoE	dense	sparse MoE
Parameters	not public	123B	671B
Context Tokens	1M	128K	64K
Output Tokens	64K		8K
Chatbot Arena Rank [5] [5]	1	70	8
License	Proprietary	Research	MIT

Table 1.1: Comparative overview of the different Large Language Models

1.5.1 GeminiPro

GeminiPro 2.5 is a large language model developed by Google [8]. The initial release of this generation was on March 25, 2025 [11]. Its architecture builds upon the sparse Mixture-of-Experts Transformer design used in previous Gemini models. A notable feature of GeminiPro 2.5 is its large context window, which supports up to one million tokens. The model demonstrates strong reasoning capabilities and currently holds the top position on the Chatbot Arena Leaderboard [5]. This model was selected for its state-of-the-art performance and demonstrated reliability in public evaluations.

1.5.2 Mistral

Mistral Large 2 is a language model developed by Mistral AI [18], first released on July 24, 2024 [17]. It employs a dense Transformer architecture optimized for cost-efficient and high-throughput inference, it features 123 billion parameters with a 128,000-token context window. Mistral Large 2 offers advanced function-calling capabilities that are beneficial for the framework

[4] Reported is the release date of that model "family", not necessarily the latest update, as this is more indicative of how new the model is.

[5] Latest version as of this writing was from 22.05.2025, the Rank (UB) is given

used in this research. Performance benchmarks place it competitively alongside other leading large language models. The model is available under a research license[6]. It was chosen for this study due to its public release for research purposes and its origin within a European company.

1.5.3 DeepSeek

DeepSeek-V3 is a model developed by DeepSeek AI [6] and was initially released on December 27, 2024. It uses a sparse Mixture-of-Experts Transformer architecture with a total of 671 billion parameters, of which 37 billion are active per token. DeepSeek-V3 features a 128,000-token context window and is noted for its cost-efficient training approach. According to the Chatbot Arena Leaderboard [5], DeepSeek-V3 is the highest-performing open-source model at the time of the writing of this thesis. This model was selected for its strong benchmark performance and open source availability.

1.6 Challenges

During initial tests with the setup, some problems arose that needed to be addressed.

1.6.1 Forced Tool Usage

During initial experiments and test runs, it became apparent that the agents were prematurely ending the discovery process due to incorrectly formatted tool calls. Instead of correctly outputting a tool call, they responded with a text message describing the function they intended to call. As a result, the libraries used to interface with these LLMs did not recognize the response as a valid tool call. During inference, the LLMs accept an additional flag indicating that they must respond with a tool call. This could resolve the issue; however, for the process to function correctly, the LLMs must also be able to indicate when they are finished using tools and wish to return a normal message. This is achieved by providing the LLMs with an additional tool that they can use to signal their intention to exit tool-call mode. When they respond using this tool, they are immediately recalled without the flag that enforces tool-call responses[7]. This seemed to improve the robustness of the system and is therefore used in all the agents.

1.6.2 Internal Thinking Step

While testing the setup, Anthropic published there findings on how the adding of a special "*think*" tool improved the performance of agents [2]. The idea is similar to the Chain of thought idea by Wei et al. [19] explained in **??**. The model can use this tool to note down thoughts and think about the next steps. Due to the promising results this tool was also added to the agents in these experiments. As Anthropic suggest this tool is mostly useful when the agent has a more complex task, as otherwise the usage of this tool, diminishes the performance. Therefore in these experiments, the agent who has the *Manager* role gets this tool, as well as the *Interviewer* agent in the *Manager* setup.

1.7 Different Multi-Agent Setups

The four different Multi-Agent systems resulting from the application of the Gaia methodology by Wooldridge et al. [20], as discussed in **??**, will be implemented using the agent framework described in **??**. All the system prompts that give the agents their role and task description can be found in **??**. Table 1.2 provides a quick overview of the four Multi-Agent systems that are

[6]The model is available on huggingface and licensed under the Mistral Research License.
[7]To be precise, they can still call a tool afterwards, but they don't have to.

compared, the agents involved, and the number of agent instances. How the agents are connected is best visualized in the Acquaintance Models of the Gaia methodology in **??**.

Multi-Agent Systems	Agents	Agent Instances
Monolithic	Process Consultant	1
Duo	Manager_KnowledgeGatherer, Process Modeler	2
Manager	Manager, KnowledgeGatherer, Process Modeler	3
Team	Manager, KnowledgeGatherer, Process Modeler, Interviewer	3 + #Process Worker

Table 1.2: Overview of the different Multi-Agent systems that are tested.

The three dimensions: 4 different Multi-Agent systems (Table 1.2), 3 different LLMs (Table 1.1), and 3 different process scenarios (Section 1.3); lead to 36 different runs that are conducted. All the results are tracked using MLflow [15] as explained in **??**.