OrangeSpot

Designing A Framework for Process Modelling using Large Language Models in a Multi-Agent Architecture

W.H. Monsma

Master's Thesis

Business Information Technology

UNIVERSITY OF TWENTE.

First Supervisor
Dr.ir. J.M. Moonen
Faculty of Behavioural, Management and Social sciences

Second supervisor
Dr. F.A. Bukhsh
Faculty of Electrical Engineering Mathematics and Computer Science

Company supervisor
M. Verburg Msc.
OrangeSpot B.V.

Master's Thesis

Designing A Framework for Process Modelling using Large Language Models in a Multi-Agent Architecture

Author

Name	W. H. Monsma
Study	Master Business & Information Technology
Specialization	Enterprise architecture & IT Management
Student number	S2381818
E-mail	Wytse.monsma@gmail.com

Graduation Committee

First supervisor	Dr.ir. J.M. Moonen *Faculty of Behavioural, Management and Social sciences*
Second supervisor	Dr. F.A. Bukhsh *Faculty of Electrical Engineering Mathematics and Computer Science*
Company supervisor	M. Verburg Msc. *OrangeSpot B.V.*

Preface

This master's thesis marks the completion of the master's programme in Business Information Technology at the University of Twente, and was carried out in collaboration with OrangeSpot B.V.

The project sits perfectly at the intersection of the things I enjoy most about our field: applied AI that helps people, designing interactions that feel intuitive rather than magical, and automation that removes friction without taking away control. In this thesis, I explore how large language models can be used to support business process modelling. Specifically, by designing and evaluating a guided process-modelling assistant that uses a multi-agent architecture. It was truly amazing working on this project and gaining valuable knowledge on using AI in practice.

I would like to thank my university supervisors, Hans Moonen and Faiza Bukhsh, for their guidance throughout the project. Their feedback consistently helped me sharpen both the academic reasoning and the practical relevance of the work.

I am also grateful to OrangeSpot for offering a challenging and highly relevant context for this research. I want to thank my company supervisor, Marlène Verburg, and colleague David Huistra. Marlène's practical insights helped steer research in a more useful direction, and David's technical expertise helped me overcome hurdles during development.

Finally, I would like to thank all participants who contributed to the evaluation. Whether by piloting prototypes, taking part in user tests, or reviewing outcomes as experts. Your time, critical comments, and questions materially improved the quality of this thesis and the artifact it describes.

I invite you to read this work, and I hope it offers useful insights. Both into process modelling support, and into what it takes to make AI assistance feel intuitive and simple.

Wytse Monsma

Abstract

This thesis addresses a recurring challenge in business process management: domain experts often have strong operational knowledge but struggle to translate it into process models without consultancy support. In OrangeSpot's onboarding context, this creates a bottleneck between scalable self-service and high-quality consultant-led modelling.

Following a design science approach combining stakeholder interviews, a literature review on process generation and evaluation methods, and four iterative design cycles, a prototype was developed for a process modelling assistant. Early prototypes explored different levels of automation and guidance, gradually converging on a hybrid approach that combines an agent-based architecture with deliberate interaction design. The final assistant employs multiple specialised agents and a human-in-the-loop workflow, where users approve suggested changes, control the modelling scope, and switch between process and entity perspectives.

A between-subjects evaluation compared AI-assisted modelling with manual modelling using a standard onboarding scenario. While expert-rated model quality was on average comparable between conditions, the AI-assisted results showed a wider range of outcomes. Participants reported higher perceived efficiency and ease of use, suggesting the assistant effectively reduces the modelling barrier. However, the limited sample size restricts statistical confidence, and consistency remains a key challenge. Overall, the results highlight clear potential for LLM-based modelling support, with propositions for future work focusing on more consistent outcomes.

Executive summary

This thesis set out to address a practical and increasingly relevant challenge in business process management: how domain experts, who possess rich knowledge of their own work but limited modelling expertise, can be supported in creating useful and accurate process models. In the context of OrangeSpot, this challenge manifests as a structural bottleneck. Consultancy-led onboarding produces high-quality models but is not always available, while self-service often leaves users struggling with a blank canvas and inconsistent results. The overarching goal of this research was therefore to design, implement, and evaluate a framework that enables guided, autonomous process modelling by domain experts.

The main research question asked: *How can state-of-the-art process-modelling techniques be leveraged to guide domain experts in modelling business processes?* This thesis demonstrates that such guidance can be realised by combining large language models with a multi-agent, human-in-the-loop architecture that supports iterative interaction. Instead of attempting to replace human judgement or fully automate modelling, the proposed framework leverages LLMs as collaborative assistants that help users overcome the initial blank-canvas problem, suggest structure and terminology, and support stepwise refinement within the constraints of an existing modelling language.

The research followed a Design Science Methodology, in which each phase produced outcomes that directly informed the next. Interviews with OrangeSpot stakeholders, combined with stakeholder analysis and the ISO/IEC 25010 quality model, resulted in a concrete set of functional and non-functional requirements for a process modelling assistant. These requirements articulated what the assistant must do (generate initial models, support iterative refinement, and integrate with the existing tool) as well as how well it must perform in terms of speed, usability, perceived usefulness, and accuracy. This phase transformed an initially broad onboarding problem into a clearly bounded design space.

Building on these requirements, a semi-systematic literature study explored the technical landscape of process model generation and evaluation. The review showed that large language models emerged as particularly suitable because of their ability to work with incomplete input, engage in dialogue, and adapt outputs based on feedback. At the same time, the literature review informed how model quality could be assessed in a way that matches practical use, leading to the selection of expert evaluations, behavioural metrics, and user perceptions as complementary measurement instruments. Together, the requirements from the problem investigation and the insights from the literature formed the basis for initial design choices.

These combined insights were translated into an initial prototype of a process modelling assistant. The first versions explored how LLM-based support could be embedded into the OrangeSpot modelling workflow. Through internal test iterations with expert users, the framework was progressively refined. Each iteration addressed shortcomings observed in earlier versions, such as limited user control, insufficient context selection, or unclear division of responsibilities between system components. This iterative validation phase resulted in a multi-agent, human-in-the-loop architecture that balances automation and user agency, and that fits the practical constraints of the existing platform.

The final version of the assistant was evaluated in a controlled experiment that compared AI-assisted modelling with a manual baseline. The results show that the assistant can meaningfully support domain experts, but not in a uniform way. Many participants experienced

the assistant as helpful, efficient, and easier than starting from a blank canvas, and response times were generally acceptable for an experimental setting. At the same time, a substantial share of users did not reach a first usable model within the targeted 30 minutes, and expert evaluations revealed mixed completeness and usability. The central outcome is that the assistant acts less as a uniform performance boost and more as an amplifier of user behaviour: expert-rated model quality becomes more variable, while perceived ease and efficiency improve consistently. This also surfaces a usability-expertise trade-off: shared responsibility between user and assistant can make modelling feel easier yet does not guarantee robust outcomes. Finally, the observed gap between user- and expert-rated quality suggests AI-assisted modelling should be treated partly as an interaction issue, not only a generation problem.

Although the evaluation provides useful first evidence for the applicability of a process modelling assistant, its conclusions are bounded by several limitations. Firstly, the user group was small and homogeneous, which limits how far the results can be generalised. Secondly, the study used only one supermarket use case, and many participants potentially lacked familiarity and intrinsic motivation to properly conduct the exercise. Thirdly, the 30-minute, between-subject design prevents within-person comparison. Fourthly, limited completeness makes other quality scores harder to interpret. Finally, the assistant's open-ended chat interaction made outcomes strongly dependent on user strategy, so tool capability and user behaviour are difficult to separate.

Future work should build on these findings in three directions. Firstly, the empirical evidence should be strengthened through larger and more diverse samples, multiple realistic use cases, and stronger study designs such as within-subject or longitudinal evaluations, ideally complemented with field pilots in real OrangeSpot onboarding projects. Secondly, alternative technical options should be explored, especially approaches that ground the assistant in OrangeSpot's growing corpus of high-quality blueprints (e.g., retrieval-based grounding or fine-tuning), as well as systematic comparisons of newer foundation models and agent configurations on quality, latency, controllability, and cost. Finally, an inverse prototyping strategy is proposed: start with a small set of well-defined, high-value modelling actions discovered from usage data, implement them as explicit UI features rather than free-form chat, and only expand capabilities once reliability is demonstrated.

In conclusion, this thesis shows that state-of-the-art LLM-based techniques can be leveraged to guide domain experts through process modelling when embedded in a multi-agent, human-in-the-loop framework. The assistant lowers the blank-canvas barrier and can help users progress faster, but consistency remains a critical challenge. Addressing the evaluation and interaction limitations above is the most direct path toward a dependable assistant that supports autonomous modelling at scale.

Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Full term
AI	Artificial Intelligence
API	Application Programming Interface
BART	Bidirectional and Auto-Regressive Transformers
BPM	Business Process Management
BPMN	Business Process Model and Notation
CEO	Chief Executive Officer
CRM	Customer Relationship Management
CRUD	Create, Read, Update, Delete
CTO	Chief Technology Officer
DPR	Dense Passage Retrieval
DSM	Design Science Methodology
ERP	Enterprise Resource Planning
GAI	Generative Artificial Intelligence
GED	Graph Edit Distance
GPT	Generative Pre-trained Transformer
GraphRAG	Graph-based Retrieval-Augmented Generation
HBO	Higher Professional Education (Hoger Beroepsonderwijs)
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
IT	Information Technology
IQR	Interquartile Range
JSON	JavaScript Object Notation
LLM	Large Language Model
MAO	Multi-Agent Orchestration
MAS	Multi-Agent System
MBO	Secondary Vocational Education (Middelbaar Beroepsonderwijs)
NER	Named Entity Recognition
NLP	Natural Language Processing
NP	Noun Phrase
ORM	Object-Relational Mapping
PET	Process Extraction from Text (also PET dataset)
PM	Process Mining
POS	Part-of-Speech
POWL	Partially Ordered Workflow Language
PP	Prepositional Phrase
RAG	Retrieval-Augmented Generation
RAGA	Retrieval-Augmented Generation Assessment
RQ	Research Question
SaaS	Software as a Service
SAP	Systems, Applications, and Products in Data Processing
SQ	Sub-question
UI	User Interface
URL	Uniform Resource Locator
UTAUT	Unified Theory of Acceptance and Use of Technology
UX	User Experience
VP	Verb Phrase
WO	University Education (Wetenschappelijk Onderwijs)
XML	Extensible Markup Language
XPDL	XML Process Definition Language

1 INTRODUCTION

Change management projects are a critical yet challenging aspect of organisational development. Drivers such as organisational growth, mergers, regulatory shifts, or sustainability initiatives can trigger the need for structured change efforts. Despite clear intentions and careful planning, many initiatives struggle to deliver lasting results due to misalignment between stakeholders, unclear ownership of tasks, siloed decision-making, or scattered documentation. As organisations grow, these challenges often intensify because complexity increases and strategic objectives become harder to connect to day-to-day operations.

Achieving alignment in change projects requires an explicit and shared understanding of how work is performed: the organisation's business processes. A business process can be defined as "a set of activities that are performed in coordination in an organisational and technical environment" (Weske, 2024). In early stages of organisational development, processes are often informal: adaptability and fast feedback cycles help young organisations find their place in the market. However, as organisations mature, informal processes can lead to inefficiencies, duplicated work, and inconsistent execution (Weske, 2024). As noted by Greiner (1998), the creative activities that enable early growth can later become a source of coordination problems. Sustained growth therefore typically requires a gradual shift from flexibility toward standardisation and explicit coordination.

Managing and improving this coordination is central to Business Process Management (BPM), which encompasses "concepts, methods, and techniques to support the design, administration, configuration, enactment, and analysis of business processes" (Weske, 2024). A key practice within BPM is business process modelling: creating conceptual representations of processes, often using graphical notations to outline tasks, decisions, and their underlying logic (Reijers & Mendling, 2011). Process models can support discussion and documentation, but also analysis and execution (Van der Aalst, 2011). As such, they serve as a practical bridge between strategic intentions and operational reality.

In many organisations, producing useful process models requires involving a diverse range of stakeholders such as employees, managers, customers, or domain experts (Rosemann, 2006). These stakeholders contribute essential domain knowledge, yet they often lack expertise in formal modelling techniques. This creates a tension between inclusivity and modelling quality: broader participation can improve relevance and correctness, but limited modelling skill can reduce consistency and understandability. Prior work indicates that modeller experience significantly affects model understandability, both objectively and subjectively (Dikici et al., 2018).

These challenges are particularly visible in software platforms that aim to support organisational change through process modelling. One such platform is OrangeSpot: a SaaS solution that supports organisations in documenting, structuring, and managing (digital) business processes, and that explicitly aims to make modelling accessible to a wide range of stakeholders through a simplified and visual modelling approach.

However, even when process models are easier to interpret, creating them from scratch remains difficult for many users. In OrangeSpot, users frequently experience a "blank canvas" effect: they possess rich knowledge of how work is done, but struggle to decide where to start, what structure to follow, and what level of detail constitutes a high-quality model. To reduce

this barrier, OrangeSpot currently offers an Excel-based template to collect process information and provides consultancy support to translate this information into structured models. While this approach can produce strong outcomes, it is time-consuming, difficult to scale, and can deter smaller clients who prefer a more self-guided onboarding experience.

Recent developments in AI-based systems create opportunities to support users more actively during modelling by enabling more natural, interactive elicitation of process knowledge. At the same time, the way modelling support is commonly studied and evaluated does not always reflect real onboarding contexts: onboarding typically involves iterative clarification and refinement rather than a single conversion step. This motivates the need to better understand and support the *iterative path* from domain knowledge to a usable process model in practice.

1.1 PROBLEM STATEMENT

Organisations using OrangeSpot possess rich domain knowledge about their own ways of working, yet many lack the modelling expertise needed to convert that knowledge into clear, executable business process models. Within the platform, users regularly confront a "blank canvas" effect: they know what happens in the business but are unsure where to start, which concepts to capture, or how to structure flows, decisions, and hand-offs in a consistent way. The consequence is a dependence on consultancy and manual templating, which yields quality but does not scale and can deter smaller clients.

In the academic landscape, current contributions emphasize one-shot process modelling. Textual descriptions are converted into a process model in a single attempt and compared to the works of an expert modeller or agreed upon ground truth. This largely misrepresents a real-world scenario, where a consultant would iterate together with a domain expert to come to a result. What is largely missing is an approach that combines the best of both worlds. An approach that actively guides domain experts through the modelling task.

1.2 RESEARCH GOAL

The goal of this research is to design, implement, and evaluate a framework that enables users or domain experts to design process models of sufficient quality autonomously. Therefore, this thesis first aims to discover requirements of such an artifact. Secondly, this research will aim to discover the state-of-the-art techniques that can be used to create such an artifact. Thirdly, this study aims to create an artifact based on the discovered match between requirements and technologies/methods. Finally, this research aims to evaluate the developed framework with end-users and industry experts.

1.3 RESEARCH QUESTIONS

The main research question is as follows:

Main RQ: How can state-of-the-art process-modelling techniques be leveraged to guide domain experts in modelling business processes?

To answer this question, several sub questions must be answered. Firstly, the problem context and requirements must be clearly established. To develop an understanding of the functional and non-functional requirements of a process modelling assistant, the first sub question is:

SQ1: What are the requirements of a process modelling assistant?

Additionally, the technical landscape of process modelling approaches should be discovered to understand which state-of-the-art solutions are applicable to the problem context and requirements. Therefore, the second sub question is defined as:

SQ2: What is the state-of-the-art in process model generation methods?

Consequently, we can then draw conclusions on which process model generation methods would be well-suited to form the backbone of a process modelling assistant framework. To assess well-suited methods and technologies, the third sub question is stated as:

SQ3: Which process model generation methods fit the requirements?

To be able to draw conclusions on the output quality of the developed framework, an overview of evaluation methods should be created to design the measurement instruments of our evaluation phase. Therefore, a fourth sub question is defined as:

SQ4: How can the quality of (generated) business process models be assessed?

With all relevant prior knowledge in place, design decisions can be made to come to a solution for a framework to realise a process modelling assistant. Therefore, the fifth sub question is:

SQ5: What would a framework for guided process modelling look like?

Finally, the designed framework should be evaluated to determine whether it satisfies the requirements. Therefore, the final sub question is stated as:

SQ6: To what extend does the designed framework satisfy the requirements?

1.4 METHODS

This thesis addresses a *design* problem rather than an explanatory one. The goal is not to describe how organisations currently model processes, but to design, implement and evaluate an artifact that helps OrangeSpot clients construct business process models more independently. The artifact is a process-modelling assistant and accompanying framework that guide domain experts through iterative model construction within the OrangeSpot platform.

To structure the research around this artifact, the study follows Wieringa's Design Science Methodology (DSM) (Wieringa, 2014). DSM organises design research as an engineering cycle consisting of problem investigation, treatment design, treatment validation, treatment implementation and implementation evaluation. In this thesis, these activities are instantiated as four main phases:

- Problem investigation: understanding context, stakeholders and requirements.
- Treatment design: identifying and combining solution concepts into a prototype design.
- Treatment validation: iteratively refining the artifact in a controlled setting.
- Treatment evaluation: assessing the artifact's performance with users and experts.

Each phase is linked to one or more research sub-questions (SQ1-SQ6) and associated research methods, as visualised in Figure 1.

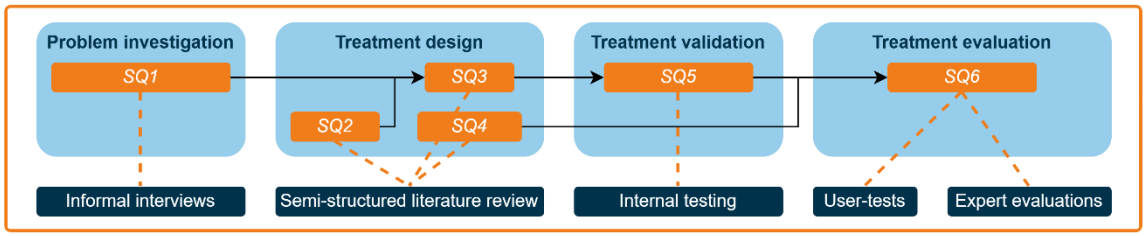


Figure 1 - Methods and Research questions mapped out against Wieringa's DSM

Problem investigation

The first phase investigates why onboarding is difficult in the OrangeSpot platform and what a process-modelling assistant should achieve. This phase focuses on SQ1: *"What are the requirements of a process modelling assistant?"*

The problem investigation combines:

- Informal and semi-structured interviews with key OrangeSpot stakeholders (management, consultants and support staff) to understand current onboarding practices, perceived problems and desired outcomes. The interview guides are based on the problem-investigation phase proposed in Wieringa's DSM.
- Stakeholder analysis using Alexander's (2007) stakeholder taxonomy to systematically identify normal operators, operational support, functional beneficiaries and sponsors, and to make explicit their (possibly conflicting) goals regarding the assistant.
- Discovery of both functional and non-functional requirements using the ISO/IEC 25010:2023 software product quality model as a checklist.

Insights from these activities are synthesised into a set of functional and non-functional requirements that define what the artifact must do and how well it must perform. These requirements form the reference point for the subsequent design, validation and evaluation phases.

Treatment design

The second phase investigates which technical approaches could realise a modelling assistant that fits OrangeSpot's context. This phase consists of a two-step literature study:

- An exploratory review to map key terminology, method families and evaluation approaches in business process model generation. Using broad search terms in scientific databases and grey literature, the review identifies major lines of work and commonly used concepts.
- A semi-systematic literature review to study these method families in more depth. Refined queries, inclusion/exclusion criteria and snowballing are used to build a focused corpus of papers and reports. For each method, the review analyses how it could support iterative, guided modelling in OrangeSpot's context.

In parallel, the review surveys evaluation techniques for process models, covering quantitative and qualitative approaches. The outcome of this phase is a set of design principles and evaluation dimensions that inform the prototype artifact and the later measurement instruments.

Treatment validation (SQ5)

The third phase focuses on shaping and validating the artifact itself and relates primarily to SQ5. Here, the research follows an iterative prototyping approach:

- A dedicated test environment is set up that mirrors OrangeSpot's modelling interface while remaining technically isolated from production.
- Within this environment, several design iterations of the assistant are implemented. Each iteration specifies an interaction design (how users collaborate with the assistant), and an architectural setup (e.g. division of responsibilities between components or agents).
- Each version is internally tested through pilot sessions with OrangeSpot staff and expert users. Participants perform short modelling tasks while thinking aloud, and their interactions are logged. Qualitative observations (e.g. points of confusion, perceived control) and basic behavioural data (e.g. response times, number of suggestions) are used to identify design issues and refine the next iteration.

Treatment validation thus serves to check whether the proposed design is technically feasible and usable in a realistic setting before exposing it to a larger user group in the evaluation phase.

Treatment evaluation (SQ6)

The final phase evaluates the most mature version of the artifact in a more controlled and systematic way, addressing SQ6. The evaluation takes the form of a between-subjects user test compares a manual modelling condition to an assisted condition using the developed assistant. Several measurement instruments are used in combination:

- a pre-task questionnaire capturing demographics and prior experience.
- System logs monitoring user behavioural metrics
- post-task questionnaires assessing technology acceptance constructs based on the UTAUT framework.
- expert evaluations in which experts assess the resulting process models

This mixed-methods setup allows the study to relate the assistant's use to both objective indicators (e.g. model quality, efficiency) and subjective user perceptions, while directly linking outcomes back to the requirements defined in the problem investigation.

In summary, the thesis applies Wieringa's Design Science Methodology to structure the research from understanding the problem context, through designing and refining an artifact, to evaluating its effects. Problem investigation relies on interviews, stakeholder analysis and quality modelling to derive requirements (SQ1). Treatment design uses exploratory and semi-systematic literature reviews to identify suitable techniques and evaluation dimensions (SQ2-SQ4). Treatment validation employs iterative prototyping and internal testing in a safe test environment to converge on a promising assistant design (SQ5). Treatment evaluation implements a controlled user study complemented by expert assessments to determine how well the assistant meets the requirements (SQ6). Figure 1 summarises how these phases, sub-questions and methods connect in the overall research design.

1.5 Outline

The outline of this research is as follows: Chapter 2 covers the problem investigation phase and answers SQ1. Chapter 3 addresses treatment design and answers SQ2-SQ4. Chapter 4 focuses on treatment validation and answers SQ5. Chapter 5 covers treatment evaluation and answers SQ6. Chapter 6 synthesises and discusses the outcomes across DSM phases, reflecting on SQ1-SQ6 and discussing the thesis contributions, limitations, and areas for future work.

Table 1 - Research outline

Chapter no.	Method(s) used	Question(s) answered
2	Informal interviews	SQ1
3	Semi-structured literature review	SQ2-SQ4
4	Iterative prototyping; internal testing	SQ5
5	User tests; expert evaluations	SQ6
6	Synthesis / discussion	SQ1-SQ6

2 ORANGESPOT B.V.

This chapter addresses sub-question 1 (SQ1), “What are the requirements of a process modelling assistant?”, by analysing OrangeSpot B.V. as the organisational and technical context in which the artifact will be deployed. It follows the problem investigation phase of Wieringa (2014)’s Design Science Methodology (DSM) through a semi-structured interview with OrangeSpot, aimed at understanding how processes are currently modelled and used in practice. The findings of this investigation are synthesised into a structured overview of OrangeSpot’s business, application and technical landscape, as well as the challenges and stakeholder needs that motivate the development of a process modelling assistant.

To elicit non-functional requirements in a systematic way, the interview questions were further structured using the ISO/IEC 25010:2023 software product quality model. ISO/IEC 25010:2023 defines a product quality model consisting of a set of characteristics (e.g. functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability and portability) and associated sub-characteristics for ICT and software products. (ISO & IEC, 2023) These characteristics provide a common vocabulary for discussing quality properties and serve as a checklist to ensure that non-functional aspects of the modelling assistant are not overlooked.

The remainder of this chapter is structured as follows. Section 2.1 introduces the business landscape of OrangeSpot: mission, value proposition, services and customer segments. Section 2.2 describes the application landscape, with particular attention to the OrangeSpot modelling language and the functional modules of the platform. Section 2.3 outlines the technical architecture, including the development stack and important data communication standards. Across these sections, the core issues and stakeholder needs that emerged from the problem investigation are highlighted. These issues are then translated into functional and non-functional requirements for a process modelling assistant, which later form the basis for selecting appropriate methods and techniques in the design and evaluation of the artifact.

2.1 BUSINESS LANDSCAPE

OrangeSpot B.V. is a Dutch software company based in Enschede that offers a SaaS platform for process and organisational change. The company positions its mission as helping organisations optimise the “building blocks” of their operations, processes, data, IT and people to realise impactful, sustainable change. Rather than focusing solely on process documentation, OrangeSpot aims to connect process modelling to change initiatives and project management. At a high level, OrangeSpot’s vision consists of three core pillars: An approach for process- and organisation-level change projects, an intuitive software platform to support such projects, and experienced consultancy services and support to. These three pillars position OrangeSpot not just as a modelling tool, but as a combined methodology and platform for steering change projects.

2.1.1 Value proposition and services

OrangeSpot’s core value proposition is providing a single environment in which organisations can map, discuss and implement change in their processes. The platform is presented as a way to align processes, data, organisation and IT, so that changes and ideas become part of a coherent “big picture” instead of isolated initiatives. From a business perspective, the platform offers several key services:

Process visualisation
The process tool enables teams to visualise current and future processes ("blueprints") quickly and collaboratively. Workshops can be run directly in the tool, allowing consultants and stakeholders to co-create process models in real time rather than relying on post-hoc documentation. This accelerates the modelling process and helps to create shared understanding across stakeholders, addressing common issues in change projects such as lack of shared mental models and "thinking in silos".

Conversations, actions and decisions
OrangeSpot emphasises that many projects fail not because of the technical solution, but because of insufficient buy-in, fragmented communication and untracked decisions. To address this, the platform provides modules for capturing:

- conversations around specific parts of the process,
- decisions taken in relation to process changes, and
- actions assigned to roles or stakeholders.

These artefacts are linked back to the processes they concern. This creates an auditable trail of why certain changes were made and who is responsible for follow-up, thereby increasing transparency and supporting governance.

Dashboarding and monitoring
A progress dashboard offers a high-level view of project and implementation status, grounded in the underlying processes. Project managers can see which process areas have been modelled, which actions are still open, and where potential bottlenecks in change implementation exist.

Taken together, these services support change projects along the full lifecycle: from initial mapping and understanding of processes, through discussion and decision-making, to monitoring implementation progress. The process models (blueprints) are central to this lifecycle: they provide the shared representation that connects the different services.

2.1.2 Customer segments / target audience

OrangeSpot mainly targets organisations undergoing process- or digital transformation, where cross-functional collaboration and stakeholder engagement are critical. Typical stakeholders include:

- Leadership and managers, who need insight into the impact of change on the organisation;
- Project teams (e.g. process owners, business analysts, consultants) who are responsible for mapping and redesigning processes; and
- Operational staff and other stakeholders, whose day-to-day work is affected by process changes and whose input is needed to make models both accurate and acceptable.

The platform is relevant for both private and public organisations, and for projects where process clarity, stakeholder alignment and traceable decision-making are essential. For the purposes of this thesis, OrangeSpot is therefore taken as a representative context where a process modelling assistant could create value by supporting consultants and stakeholders in constructing and maintaining high-quality process models.

2.1.3 Competitive landscape

OrangeSpot operates in a competitive landscape that spans three main areas of capability: collaboration, documentation and process modelling. Different products cover different parts of this spectrum. Collaboration-documentation platforms such as Notion provide rich shared workspaces for notes, pages and lightweight databases. They are excellent for documenting project context or decisions but offer only informal support for structured process models.

At the other end are collaborative whiteboarding and diagramming tools such as Miro and diagrams.net (Miro, n.d.; Draw.io, n.d.). These tools make it easy for teams to sketch and refine process flows together in real time. However, the resulting diagrams tend to remain static and are not tightly integrated with ongoing documentation, decision tracking or implementation monitoring; turning them into extensive forms of process documentation usually requires manual effort or additional systems.

A third group consists of dedicated process documentation and handbook tools, such as ProcessMaker and Navvia (ProcessMaker, n.d.; Navvia, n.d.). These platforms focus on maintaining formal process descriptions, procedures and manuals, often with strong versioning and compliance features. They typically offer modelling capabilities, but collaboration is mostly limited to reviewing and approving documents rather than supporting discussions or actions.

OrangeSpot differentiates itself by explicitly combining all three capability areas in a single platform. It offers a structured process modelling language (blueprints), uses these models for extensive documentation, and supports continuous collaboration through conversations, actions, decisions and progress dashboards. In doing so, it positions itself not as a point solution for either modelling, documentation or collaboration, but as an integrated environment in which process models, organisational knowledge and change management are tightly connected.

OrangeSpot is a local Dutch provider, presenting itself as a combination of a change approach, the platform, and tailor-made support/services. This can be a positive difference compared to very large platforms. A local player can stay closer to customers, help with onboarding and training, and adapt quickly to what client organisations need.

Figure 2 - Venn-diagram of OrangeSpot's competitive landscape

2.2 APPLICATION LANDSCAPE

This section describes the application landscape of OrangeSpot, with a focus on the elements that are most relevant for a process modelling assistant: the OrangeSpot modelling language and the main functional modules of the platform.

2.2.1 The OrangeSpot modelling language

The OrangeSpot modelling language is designed as a minimal, highly visual notation aimed at non-expert stakeholders. Rather than offering a large catalogue of symbols, it prioritises a small, intuitive set that can be understood in workshops by business users, consultants and other domain experts without specialised training in process modelling. The emphasis is on readability and on delayed complexity: stakeholders should be able to start from a high-level overview and then drill down into more detailed subprocesses only when this is necessary for the discussion.

The central data object in this language is the blueprint, a visual representation of a process or value stream. A blueprint typically consists of chains of process steps, shown as orange arrow-shaped activities, connected into a flow that indicates the order in which work is performed. These steps can be organised hierarchically. At a top level, a blueprint may show only a small number of coarse-grained activities; each of these can in turn be decomposed into a more detailed subprocess. Nesting different levels of granularity allows the same notation to support both strategic overviews and operational detail, while preserving a consistent visual language across levels of abstraction. Examples of such a decomposition is included in a screenshot of the process tool in Figure 3 below.

A distinctive feature of the blueprint concept is that each process step can be enriched with attributes drawn from multiple perspectives. For every step, it is possible to associate the data objects that are used or produced (for example forms, documents or datasets), the applications or IT systems that support the work, and the organisational roles that are responsible for carrying it out. As a result, a single blueprint conveys who does what, with which system and which information. This integrated view is particularly important in change projects,

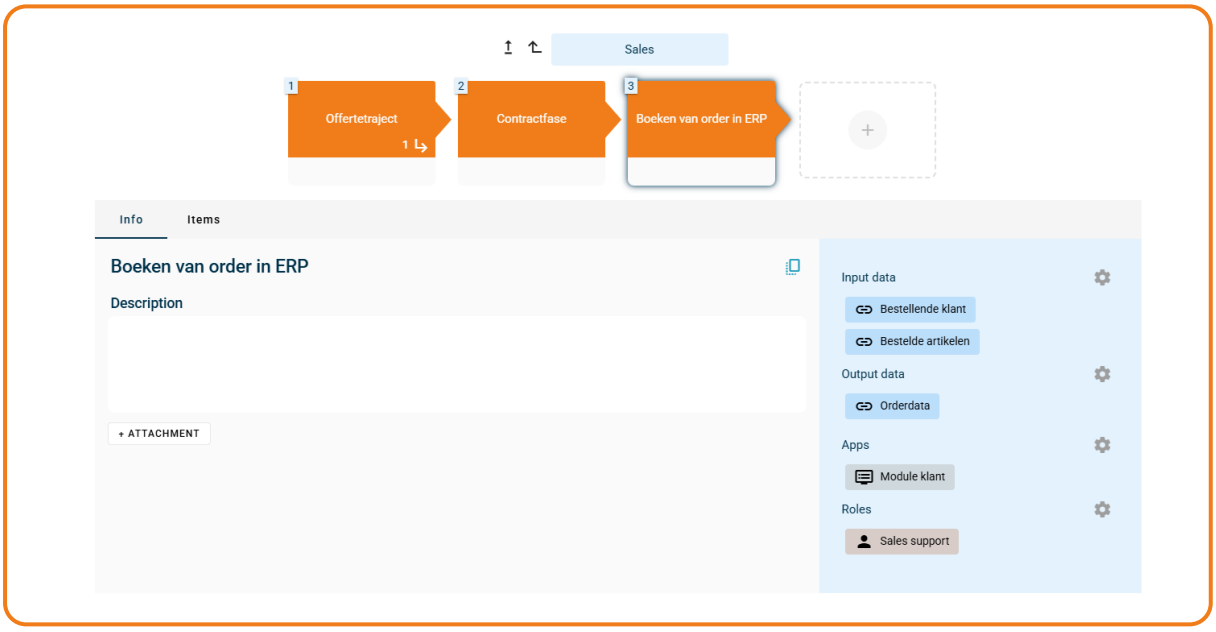


Figure 3 - Screenshot of the OrangeSpot process tool

where modifications to processes often have implications for data management, system integrations and role definitions.

When compared to standard notations such as BPMN 2.0, the OrangeSpot language deliberately constrains its symbol set and downplays formal control-flow constructs such as gateways and events. BPMN provides rich semantics for branching, synchronisation and exception handling, but many business stakeholders find BPMN diagrams difficult to read without training. In contrast, the OrangeSpot notation trades formal expressiveness for accessibility and shared understanding. The primary concerns are the sequence and decomposition of steps and the completeness of the attributes attached to them, rather than the detailed documentation of every possible restriction or exception.

2.2.2 Functional modules

Within the OrangeSpot application, the modelling language is used across several functional modules. Firstly, the process tool is where processes are created and maintained. Secondly, the Actions & Decisions module records important changes and their implications. Thirdly, the Conversations module allows people across the organisation to participate in discussions about these changes. Finally, the progress dashboard aggregates information from all modules into an overview that makes it possible to follow the progress of the change project. Across these modules, the quality of the underlying process models is crucial. They provide the common reference point that ties actions, discussions and monitoring together.

Process tool

The process tool forms the core modelling environment. Here, users create and edit blueprints, define and order process steps, introduce subprocesses, and attach data, applications and roles to each step. This environment is used both during interactive workshops, where models are constructed collaboratively with stakeholders, and when further refining the processes. The quality of the blueprints produced in the process tool is critical, because all other modules reference these models. Inaccurate or incomplete blueprints quickly propagate through the rest of the platform, whereas clear and well-structured models provide a solid foundation for subsequent documentation and coordination.

Actions & Decisions module

The Actions & Decisions module builds on the blueprints by allowing users to log and track actions and decisions that arise during change projects. Actions represent concrete tasks that must be carried out, while decisions capture choices made about process design or implementation. Both are explicitly linked back to specific process steps or to entire blueprints. This linkage means that ambiguities or gaps in the process model immediately translate into unclear responsibilities or poorly documented decisions. Opposingly, when a blueprint is well structured and precise, it provides a clear anchor for assigning actions, explaining the rationale for design choices and revisiting these choices as processes evolve. High-quality process models are therefore vital for reliable action and decision management.

Conversations module

The Conversations module adds further possibilities for collaboration. It enables stakeholders to conduct and document discussions about specific blueprints, individual steps, or actions and decisions. Rather than treating the process model as static documentation, the platform turns it into a living context around which conversations are organised. Here too, the granularity and clarity of the process models matter. If steps are defined too broadly, conversations tend to become unfocused and mix different issues. If models are inconsistent, participants may rely

on different implicit interpretations of the same step, which can lead to misunderstandings and misaligned expectations. Well-defined, consistent models help to keep discussions concrete and ensure that participants are talking about the same issues.

Progress dashboard
The progress dashboard provides a high-level view of project and implementation status, often aggregated along process dimensions. It can show, for example, which blueprints have been completed or validated, where actions remain open, and which parts of the organisation are lagging in implementing agreed changes. Because these dashboards are grounded in the process models and their associated artefacts, any inaccuracies or omissions in the underlying blueprints can lead directly to misleading management information. Maintaining quality and completeness of blueprints over time is therefore not only a modelling concern but also a governance concern: the reliability of monitoring and steering mechanisms depends on the quality of the models on which they are based.

Taken together, the modelling language and functional modules of OrangeSpot define an application landscape in which process models sit at the. High-quality process models enable each module to function effectively and ensure that change projects can be planned, discussed and monitored based on a shared and accurate representation of how the organisation works.

2.3 TECHNICAL ARCHITECTURE

OrangeSpot is delivered as a web-based SaaS platform with a clear separation between frontend and backend concerns. Both layers are deployed as Docker containers, which encapsulate their runtime environments and dependencies. This containerised setup simplifies deployment and makes it easier to run the platform consistently across development, test and production environments. Blueprint information is sent between the two environments through a unified blueprint representation in JSON format.

2.3.1 Frontend

The frontend of OrangeSpot is a browser-based web application built with JavaScript, using React and Vue for the user interface. Screens are composed of React/Vue components that render process diagrams, actions and decisions, conversations and dashboards. These components handle user interaction (for example adding or editing process steps) and communicate with the backend via HTTP requests to JSON-based APIs.

When a user works with a blueprint, the frontend sends a request to the Blueprint API, receives the corresponding Blueprint JSON, and translates this into visual elements on the screen. When the user makes changes, such as renaming a step, adding a role or linking an application, the frontend updates its local state and sends the relevant changes back to the backend.

Authentication and authorisation are handled in cooperation with the Django backend. After a user logs in, the backend establishes an authenticated session (for example via a token or session cookie). The frontend includes this authentication information with each API call, so that the backend can enforce access control per organisation, project and blueprint. In this way, users only see and modify the processes and related information they are allowed to access.

The compiled frontend assets (HTML, JavaScript and CSS) are packaged into a Docker image. Inside this container, a lightweight web server serves the static files, while configuration values such as the base URL of the backend API are provided via environment variables at startup. This

containerised setup keeps the frontend technically separate from the backend, makes deployments repeatable and allows new UI versions to be rolled out by updating the corresponding Docker image.

2.3.2 Backend

The backend of OrangeSpot is built with Django and is responsible for all core business logic and data storage. It exposes a set of HTTP APIs that the frontend calls to work with blueprints, actions and decisions, conversations and dashboard data. For each incoming request, the backend first checks the user's authentication and authorisation, then applies the relevant business rules and finally reads or writes data in the database. Responses are returned as JSON, which the frontend uses to update the user interface.

Authentication and authorisation are handled using Django's built-in mechanisms. When a user logs in, the backend creates an authenticated session (for example using a session cookie or token). Each subsequent API request includes this authentication information, allowing the backend to enforce fine-grained access control. In practice, this means that access is checked at the level of organisation, project and sometimes individual blueprint or element, so that users can only view and change the data they are allowed to work with.

Blueprints and related objects (such as actions, decisions, conversations and roles) are stored in a relational database managed by Django's ORM. The backend translates API calls into database operations and maintains the integrity of references between entities, including the links represented in the Blueprint JSON.

The Django backend also runs in its own Docker container. This container bundles the application code, its Python dependencies and runtime configuration. Environment variables are used to provide settings such as database connection details, secret keys and allowed origins. Deployments consist of building and distributing an updated backend image, which can then be started or scaled independently of the frontend container. This separation keeps concerns clear: the backend focuses on secure, consistent data and business rules, while the frontend focuses on user interaction.

2.3.3 Blueprint data representation

In OrangeSpot, each blueprint is stored as one JSON document. The main part of this document is an elements object, which contains a byId map. In this map, every key is an element ID and every value describes one element in the blueprint. A blueprint is therefore seen as a set of elements, not as a deeply nested tree.

Each element has at least an id, a type and a name. The type indicates what kind of object it is, for example:

- "BPProcess" for process steps,
- "BPData" for data objects,
- "BPApp" for applications / IT systems,
- "BPRole" for roles in the organisation.

Process steps (BPProcess) also contain structural information, such as a parent (the ID of the step above it) and a list of children (IDs of steps below it). In this way, the process hierarchy is built up by linking step IDs together. On top of that, a step can refer to related elements through ID lists, such as input_data, output_data, apps and roles. These fields store the IDs of data objects, applications and roles that are defined elsewhere in the same byId map.

Because elements only refer to each other by ID, the blueprint JSON forms a graph of simple, typed objects. The same element (for example a role or application) can be reused in many steps without duplication. Different parts of the platform can then "walk" through this graph in whatever direction they need: from a step to its roles, from a role to all its steps, from a step to its data, and so on. The Blueprint API exposes operations to create, read, update and delete elements in this structure, and to change the links between them. A simplified example of this JSON format is shown in Appendix A: Appendix A: Blueprint JSON format example

2.4 CLIENT ONBOARDING

2.4.1 Current practice

When a new client first starts with OrangeSpot, they enter an almost empty environment. The process tool contains no blueprints yet, so the organisation must describe its own processes before it can use actions and decisions, conversations or dashboards in a meaningful way. Although clients know their own work very well, most are not experienced process modellers. Faced with a blank canvas, they often struggle with questions such as where to begin, what level of detail to choose and what a "good" process model should look like. To lower this barrier, OrangeSpot usually offers a consultancy-led onboarding trajectory in which a consultant helps the client set up the initial models. Alternatively, clients can make use of an importable excel template that offers a more user-friendly introduction to the blueprint template.

Onboarding consultancy

The consultancy trajectory starts with a preparation phase. The consultant studies the client's organisation and the goals of the change project, examining available documentation and any material shared in advance. Based on this research, the consultant produces an initial high-level sketch of the main processes. This sketch is not yet a definitive model, but a starting hypothesis about how the organisation operates.

The second phase is a kick-off meeting with management. In this meeting, the consultant presents the initial sketch, clarifies the project objectives and agrees on the scope of the modelling work. Together, they decide which processes and departments will be included and what level of detail is required. The kick-off serves to confirm or correct assumptions from the preparation phase and to align expectations about the outcome of the onboarding.

The core of the trajectory consists of cross-sectional interviews with employees from different parts of the organisation. In each interview, the consultant walks through the interviewee's daily work: the activities they perform, how handovers take place, which systems they use and which data they handle. After each round of interviews, the consultant refines the process models in OrangeSpot, adjusting the initial sketch and adding detail where needed. In this way, the blueprints gradually evolve into a coherent and realistic description of the organisation's processes.

Finally, the onboarding ends with an evaluation meeting. Here, the consultant presents the completed blueprints to management and, where relevant, to other stakeholders. Remaining questions are discussed, small corrections are made, and agreements are reached about how the models will be used in ongoing change projects. After this meeting, the client has a set of validated process models in OrangeSpot that can serve as the backbone for further work.

Figure 4 - Visual representation of the onboarding process summarises this consultancy-led onboarding trajectory. The green bar at the top shows that the entire process typically spans two to four weeks. The orange blocks indicate the approximate effort per phase: roughly a working day for preparation, a short kick-off meeting, one to two working days of interviews, one to three days of modelling and consolidation, and a short evaluation meeting. Together, these steps amount to about 25-40 hours of consultant time and represent a rough representation for the current practice of bringing new clients onto the platform.

Self-serviced onboarding

Alongside the consultancy-led trajectory, OrangeSpot's client can choose for a self-service onboarding route. In this route, clients are expected to set up their own initial process models without intensive support from a consultant. They can do this in two ways: either by modelling directly in an empty process tool, or by filling in a structured template (in spreadsheet form) that can be imported into the platform. The spreadsheet template is explicitly intended to give clients a more familiar format to work in: many organisations are used to working with Excel and tables, and the template leverages this familiarity to lower the threshold for getting started. In both variants, however, clients must translate their own way of working into a set of process steps, group these into a hierarchy and decide which data, applications and roles are involved.

The template makes this structure explicit. Each row represents a process step and has fields for a unique ID, a name, an optional description and a parent ID. By filling in the Parent (ID) column, clients define the hierarchy between steps: high-level processes act as parents, while subprocesses and sub-subprocesses are linked as children. In addition to the basic hierarchy, the template allows clients to link process steps to other perspectives. Columns such as Data incoming, Data outgoing, App and Role accept one or more IDs (often comma-separated) that refer to data objects, applications and roles defined elsewhere. For readability, the template also includes helper columns like Name parent (automatically filled) or Name app (automatically filled), which are populated during or after import so that users can check whether the numeric IDs they entered match the intended objects. Once the template is completed and uploaded, OrangeSpot converts its contents into blueprint elements and relationships in the platform, effectively turning the spreadsheet into a first set of process models.

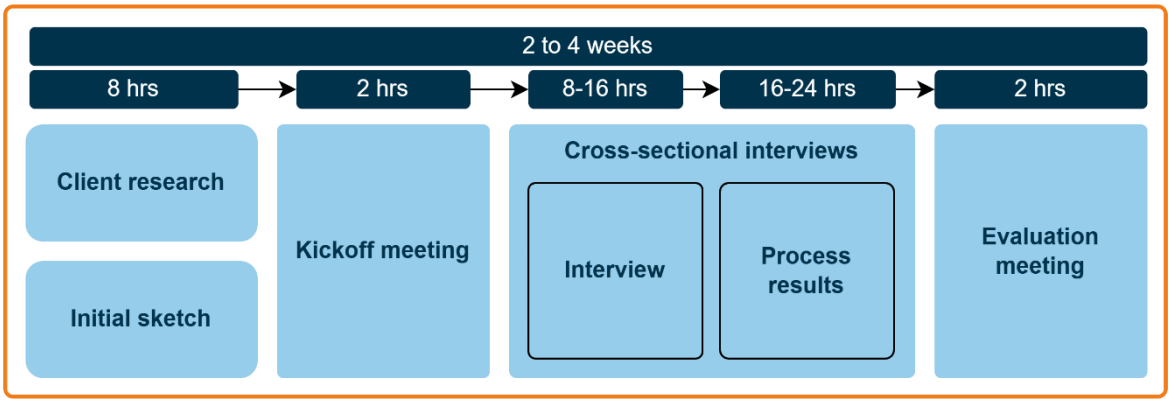


Figure 4 - Visual representation of the onboarding process

In practice, this self-service route presents several challenges, regardless of whether users work in the tool or in the template. In both cases, they start from a blank structure and must decide on the breakdown of their processes, the right level of detail and the grouping of steps. For users without modelling experience, this is a demanding task and can lead to writer's block or to inconsistent levels of detail across processes. The spreadsheet, although more familiar than a modelling canvas, introduces its own complexity: it disconnects the modelling activity from the immediate visual feedback of the process tool. Users only see the actual diagrams after import, which makes it harder to spot structural issues early. As a result, self-service onboarding often requires more effort and iteration than clients expect, and the quality and consistency of the resulting models can vary considerably between organisations.

2.4.2 The problem and opportunity

The current methods of onboarding impose a structural bottleneck on OrangeSpot: they lack capacity to scale to smaller clients, and smaller clients lack experience to onboard autonomously. The consultancy-led onboarding trajectory works well in terms of outcome. It produces high-quality process models, helps clients quickly understand the value of OrangeSpot, and gives the platform a strong starting position within the organisation. However, this approach is resource intensive. It depends on the time and expertise of OrangeSpot consultants, who must prepare, interview stakeholders and refine models. As the number of clients grows, this consultancy capacity becomes a limit: every new onboarding requires a share of expert time. At the same time, the associated consultancy effort makes onboarding relatively expensive, which can discourage smaller organisations that might otherwise be a good fit for the platform.

Self-service onboarding arises as the natural alternative. In this route, clients are expected to model their own processes, either directly in the tool or via the spreadsheet template. In theory, this should scale better and lower the cost for smaller clients. In practice, however, self-service onboarding does not work as well as intended. Many new users struggle with the "blank canvas" problem: even with a readable modelling language and a familiar spreadsheet format, they find it difficult to decide where to start, how to structure their processes and what a good model should look like. The result is slow progress, inconsistent or incomplete models, and a need for ad-hoc support that partly defeats the purpose of self-service.

On the one hand, consultancy-led onboarding consumes too much internal capacity to be offered broadly, and its cost makes it unattractive for smaller organisations. On the other hand, the self-service alternative does not provide a reliable path to good models and confident users. OrangeSpot therefore finds itself in a position where it cannot easily serve more (and smaller) clients with its current consultancy capacity, while at the same time smaller clients lack a convincing, low-effort way to adopt the platform.

An opportunity lies in designing a new way of onboarding that increases user autonomy without sacrificing model quality. Such an approach should help new clients move from an empty environment to a useful first set of process models with much less reliance on consultant time, and with more guidance than the current self-service route provides. In other words, OrangeSpot needs an onboarding mechanism that is both scalable for the company and accessible for smaller clients, by supporting users more actively in constructing their own process models.

2.5 A PROCESS MODELLING ASSISTANT

The previous sections showed that OrangeSpot faces a structural bottleneck in onboarding new clients. Consultancy-led onboarding produces high-quality models but consumes scarce expert capacity and is too costly for many smaller organisations. Self-service onboarding, in turn, scales in theory but, in practice, leaves many users facing a blank canvas and struggling to construct useful models on their own.

At OrangeSpot, the idea of introducing a modelling assistant has been proposed as a potential way to support users during blueprint creation. This section describes that proposal and uses it to explore the solution at a requirements level: what such an assistant should be able to achieve and what constraints follow from the organisational context. Together with the exploration of possible technological methods in chapter 3, these requirements form the key knowledge to start the design of a prototype.

In line with DSM, the next step is to identify who the relevant stakeholders are and what goals they have with respect to the assistant. For this, a stakeholder classification framework by Alexander (2007) is used to structure the different groups and their roles. These stakeholder goals are then combined with a structured checklist based on the ISO & IEC (2023) software product quality model, which provides a vocabulary for formulating non-functional requirements such as usability, performance and reliability. Together, the stakeholder analysis and the ISO-based quality checklist form the basis to extrapolate the functional and non-functional requirements of the process modelling assistant.

2.5.1 Stakeholder analysis

To understand for whom the process modelling assistant is being developed, and what it should achieve, this thesis uses the stakeholder classification proposed by Alexander (2007). Instead of treating “the user” as a single group, Alexander distinguishes between several stakeholder classes based on how they relate to the system. Normal operators are the people who use the system in their day-to-day work and directly interact with its functions. Operational support covers those who keep the system running and support normal operators, for example by handling configuration and support requests. Functional beneficiaries are the people who benefit from the system’s outputs without necessarily using it themselves; they care about the quality and usefulness of the results rather than the details of interaction. Finally, sponsors are the roles or organisations that initiate, fund and steer the system and decide whether it is successful enough to continue investing in it.

Applying this classification helps to avoid a narrow focus on one visible group (for example, consultants or end-users) and instead ensures that requirements reflect the needs and goals of everyone who is affected by the process modelling assistant. In the context of OrangeSpot and its onboarding process, several concrete stakeholder groups can be mapped onto these classes. Table 2 summarises these groups, their roles and their main goals with respect to the assistant.

Table 2 - Stakeholder groups and goals for the process modelling assistant

Class	Stakeholder	Description	Goals
Normal operators	End-users	project managers, IT managers, or sustainability coordinators	Easier onboarding, reduced "blank canvas" feeling, understandable models, low effort.
Normal operators	Consultants	Consultants modelling processes on behalf of clients	Faster, more efficient modelling; support for high-quality models; assistant as productivity aid.
Operational support	OrangeSpot support staff	Internal staff helping clients after onboarding	Smoother onboarding trajectories, fewer repetitive questions, reduced support load.
Functional beneficiaries	Client employees	Staff who mainly *use* process models (e.g. For training, analysis)	Clear, correct and useful models that support real change projects.
Sponsor	OrangeSpot B.V.	Company allocating resources for artifact development	Scalable onboarding, ability to handle more clients, attract smaller clients, reduced dependency on consultancy capacity.

End-users on the client side are the people who will interact with the assistant most directly. They include project managers, IT managers, sustainability coordinators and process owners who are responsible for capturing processes in OrangeSpot. Their main goal is to move from an empty environment to usable process models without needing deep modelling expertise. For them, the assistant should lower the effort needed to get started, provide clear guidance and produce models that are easy to understand and refine.

Consultants are also normal operators, but with a different relationship to the system. They use OrangeSpot as part of their consultancy work for clients and typically have more modelling experience. They are less affected by the blank-canvas problem but still spend time on repetitive tasks such as drafting typical processes or maintaining consistency across models. For this group, the assistant should function primarily as a productivity tool that accelerates routine work and suggests consistent patterns, while leaving control with the consultant.

OrangeSpot's implementation and support staff fall under operational support. They help clients during onboarding, answer questions about how to model certain situations and resolve configuration issues. Their goal is to make onboarding smoother and more predictable, and to avoid solving the same basic modelling problems repeatedly. An assistant that helps clients can reduce their support load, standardise common solutions and free time for handling more complex or exceptional cases.

Client organisation employees who mainly consume process models, rather than build them, are functional beneficiaries. They might use the models to understand new processes, to participate in change projects or to identify improvement opportunities. They may never interact with the assistant directly, but they are affected by its output. From their perspective, the crucial point is that assistant-generated models are clear, accurate enough for their purpose and aligned with how work is done, so that they support rather than hinder change.

Finally, OrangeSpot B.V. as a company acts as the sponsor of the process modelling assistant. It decides to invest in its development, integrate it into the platform and maintain it over time. Their goals are primarily strategic: to onboard more clients with less strain on consultancy capacity, to lower the cost and effort of onboarding for smaller organisations and to maintain or improve overall customer satisfaction. The assistant will be considered successful from this

perspective if it contributes to scalable, high-quality onboarding, rather than introducing new risks or support burdens.

2.5.2 Conflicts of interest

The stakeholder goals are not always fully aligned and give rise to several conflicts and trade-offs that are relevant for the requirements of the assistant.

A first tension exists between client autonomy and billable consultancy work. End-users and OrangeSpot as sponsor benefit if clients can onboard themselves with minimal consultant involvement: this reduces capacity strain on OrangeSpot and makes the platform more attractive for smaller organisations. Consultants, however, may perceive extensive automation as a potential threat to billable hours or to their perceived added value in the onboarding process. The assistant therefore promises substantial added value for consultancy work by speeding up routine modelling tasks and improving consistency, but its design and positioning should remain mindful of how increased automation may affect their role in the future.

A second tension concerns speed and accuracy. While all stakeholders would like onboarding to be both fast and correct, those responsible for modelling prefer the former, and those depending on the results prefer the latter. In practice, there is a trade-off: generating models very quickly likely means relying on limited information and assumptions, while achieving high accuracy often requires more interaction and refinement. The assistant must therefore strike a balance between providing rapid suggestions that help users overcome the blank-canvas problem and allowing sufficient review steps to avoid misleading or overly simplistic models.

A third tension arises between flexibility and standardisation. Different client organisations have their own terminology, structures and domain-specific practices, and they would expect the assistant to accommodate this variability. At the same time, OrangeSpot promotes its own approach in modelling processes. The assistant should therefore give clients room to express their specific domain context, while still guiding them towards models that follow the OrangeSpot approach.

2.5.3 Requirements

To structure the requirements for the process modelling assistant, this thesis uses the ISO & IEC (2023) software product quality model as a reference. The standard defines a set of product quality characteristics (such as functional suitability, performance efficiency, usability, reliability, security, maintainability, portability and compatibility) and a set of quality-in-use characteristics (such as effectiveness, efficiency, satisfaction and freedom from risk). The standard is not used here to perform a full formal evaluation, but as a checklist and shared vocabulary for formulating non-functional requirements in a systematic way.

A semi-structured interview with OrangeSpot's CEO and CTO translated these abstract characteristics into concrete expectations for the onboarding context. For example, interviewees emphasised that suggestions should be generated quickly, that onboarding should be completable within roughly half an hour when using the assistant, that clients should perceive the generated models as "useful" even if not perfect, and that model accuracy should be comparable to models created by implementation partners. These expectations are used below to derive functional and non-functional requirements.

In the context of a process modelling assistant for OrangeSpot, not all characteristics are equally relevant. Based on the stakeholder analysis and the onboarding problem, several

characteristics stand out as particularly important. Functional suitability is central, with the focus on generating process models from user input and supporting iterative refinement. Usability and quality-in-use are crucial, as the assistant targets users with limited modelling experience and must be experienced as easier and more helpful than starting from scratch. Performance efficiency matters because the assistant will be used interactively; long waiting times for suggestions would undermine its value. Reliability is important to prevent corrupt or misleading models, and security is needed because process descriptions can contain sensitive organisational information. Finally, maintainability is relevant because the assistant must evolve alongside OrangeSpot's application landscape.

Functional requirements

First, the assistant must be able to automatically generate process models from user input. Rather than expecting users to define every process step manually, the assistant should take one or more forms of input, such as a textual description of the organisation's work, a short list of key activities or a simple structured form and produce an initial set of process steps and relationships that follow the OrangeSpot blueprint structure. This initial model does not need to be perfect, but it should be meaningful enough for users to be able to start refining it.

Second, the assistant must support iterative refinement of these generated models. Users should be able to adjust the model based on their domain knowledge, add or remove steps, and ask the assistant for suggestions or alternatives. The assistant should handle this as a dialogue or stepwise process rather than a one-off generation, so that users can gradually converge on a model that fits their domain context.

Third, the assistant must respect OrangeSpot's blueprint structure and notation. Any models it generates, or edits must be valid with respect to the existing element types (such as process steps, roles, applications and data objects), identifiers and relationships used in the Blueprint JSON.

Finally, the assistant must integrate with the existing process tool in a way that aligns with current workflows. Users should be able to invoke the assistant from within the tool, inspect and modify its suggestions using intuitive interactions and, where needed, fall back to manual modelling. The assistant's role is to augment and accelerate the existing modelling process rather than to introduce a separate way of working. These functional requirements can be summarised as shown in Table 3.

Table 3 - Functional requirements

ID	Requirement	Short description
F1	Initial model generation	Generate a first draft process model from user input.
F2	Iterative refinement	Support stepwise refinement of the model through suggestions and adjustments.
F3	Conformity to blueprint structure	Produce and edit models that are valid OrangeSpot blueprints.
F4	Integration with existing modelling workflow	Integrate into the process tool and augment, rather than replace, manual modelling.

Non-functional requirements

The non-functional requirements are derived from the same interview and are structured using ISO/IEC 25010 characteristics, in particular performance efficiency, usability, functional suitability and quality-in-use. The focus is on five properties that OrangeSpot considers critical for the assistant to be a viable alternative to consultancy-led onboarding.

First, in terms of performance efficiency, the assistant should match the current experience of the process tool, where it is easy to incorporate new ideas into the model due to the intuitive modelling language. To preserve this feeling of fluency, the assistant must provide swift responses: process suggestions should be generated within roughly 30 seconds. This ensures that users can work interactively with the assistant without long interruptions.

Second, OrangeSpot sees potential to considerably shorten the total onboarding duration because the assistant is available at any time and does not depend on scheduling interviews. The goal is that a basic onboarding trajectory, from an empty environment to a first usable model, can be completed within 30 minutes when using the assistant. This requirement combines performance efficiency with quality-in-use.

Third, the assistant must achieve an acceptable level of perceived usefulness. In interviews, a threshold of 60% model usefulness was established as the point at which clients can be considered satisfied with the assistant's output. This reflects the idea that the assistant does not need to produce perfect models, but its suggestions must be useful enough that users recognise their own processes and see clear value compared to starting from a blank canvas.

Fourth, with respect to accuracy, models generated by the assistant should be comparable to those created via implementation partners in typical onboarding scenarios. This requirement is linked to functional suitability: the assistant's output should be sufficiently correct and complete to serve as a basis for further refinement, even if some expert adjustment is still needed.

Finally, usability is captured in the requirement that users must prefer using the assistant over starting from scratch in the process tool. If users do not experience the assistant as easier or more helpful than manual modelling, it will not address the self-onboarding problems identified earlier. These non-functional requirements can be summarised as shown in Table 4.

Table 4 - Non-functional requirements

ID	Property	Current state *consultancy*	Current state *self-onboarding*	Requirement for assistant
N1	Duration - suggestions	Not applicable	Not applicable	Generation of process suggestions within 30 seconds.
N2	Duration - total	2-3 weeks	3+ weeks	Completion of the basic onboarding process within 30 minutes.
N3	Perceived usefulness	Relatively high	Relatively low	Clients must rate model usefulness at 60% or higher.
N4	Accuracy	Relatively high	Relatively low	Models must be comparable in accuracy to those created via implementation partners.
N5	Usability	Relatively high	Relatively low	Users must prefer using the assistant over starting from scratch.

2.6 Answer to SQ1

This chapter set out to answer SQ1: “What are the requirements of a process modelling assistant?” in the context of OrangeSpot. Following the problem investigation phase of Wieringa’s Design Science Methodology, expert interviews with employees at OrangeSpot were held to discover stakeholder goals and come to a set of requirements. It revealed that OrangeSpot depends on good process models for most of its functionality, but that the current onboarding options have clear drawbacks. Consultancy-led onboarding produces high-quality models, yet it uses a lot of consultant time and is too expensive for many smaller organisations. Self-service onboarding, either in the tool or via the spreadsheet template, should in theory solve this, but in practice many users get stuck when starting from a blank environment and struggle to create consistent models on their own.

A stakeholder analysis based on a classification framework by Alexander (2007) clarified for whom the assistant is being designed. End-users and consultants want easier and faster modelling, without losing control over the results. Implementation and support staff want smoother onboarding and lower workload. Employees who use the process models need clear and accurate process descriptions to support change projects. OrangeSpot as a company wants to be able to serve more clients with the same consultancy capacity and to attract more (especially smaller) clients by making onboarding cheaper and easier. These goals are partly aligned but also create tensions, between client autonomy and billable consultancy work, or between flexibility for different organisations and a consistent OrangeSpot way of modelling.

On this basis, the chapter defined a set of requirements for the process modelling assistant, structured with the help of the ISO/IEC 25010:2023 quality model. The functional requirements state that the assistant must be able to:

- automatically generate an initial process model from user input,
- support step-by-step refinement so users can add their own expertise,
- follow OrangeSpot’s blueprint structure and notation, and
- work inside the existing process tool, as an addition to the current way of modelling rather than a separate system.

The non-functional requirements specify how well the assistant must do this:

- process suggestions should be generated within about 30 seconds,
- a basic onboarding trajectory should be completable within about 30 minutes,
- clients should rate the assistant’s models as at least 60% useful,
- the accuracy of these models should be comparable to models created by implementation partners in typical cases, and
- users should prefer using the assistant over starting from a blank canvas.

Together, these functional and non-functional requirements give a concrete answer to SQ1. In the following chapters, these requirements will be used to guide the choice of model generation methods and to evaluate prototype implementations of the artifact.

3 PROCESS MODEL GENERATION METHODS

The goal of this chapter is to answer the following sub-questions:

- **SQ2:** What is the state-of-the-art in process model generation methods?
- **SQ3:** Which process model generation methods fit the requirements?
- **SQ4:** How can the quality of (generated) business process models be assessed?

Where Chapter 2 established what a process modelling assistant should achieve in terms of functional and non-functional requirements, this chapter investigates how such an assistant could be realised using existing process model generation techniques. In other words, the focus moves from OrangeSpot's problem context and requirements to the broader technical landscape of process model generation methods.

To obtain a structured and context-aware overview, the research in this chapter builds on two complementary literature studies: an exploratory literature review and a semi-systematic literature review. The exploratory review was used to map key terminology, concepts and broad categories of methods and evaluation approaches, and to refine the search terms. The semi-systematic review then applied these terms in a more focused way to academic and grey literature, to analyse representative methods in depth and compare them against OrangeSpot's requirements.

This study combines an exploratory literature review with a semi-systematic literature review. Pre-emptively, an exploratory review was conducted to identify core terminology, concepts and methods. Its findings then steered a more focused semi-systematic review that summarizes relevant scientific and grey literature on process model generation and evaluation methods. A visual representation of the methodology is shown in Figure 5. The left-hand side shows how the exploratory phase starts from broad search terms to discover key concepts and three main methods: Natural Language Processing (NLP), Process Mining (PM), and Large Language Models (LLMs). The right-hand side shows how the semi-systematic review then applies extended queries, filters the most relevant results, screens abstracts and full texts, and uses snowballing to arrive at the final set of papers that underpin the remainder of this chapter.
The remainder of this chapter is structured as follows:

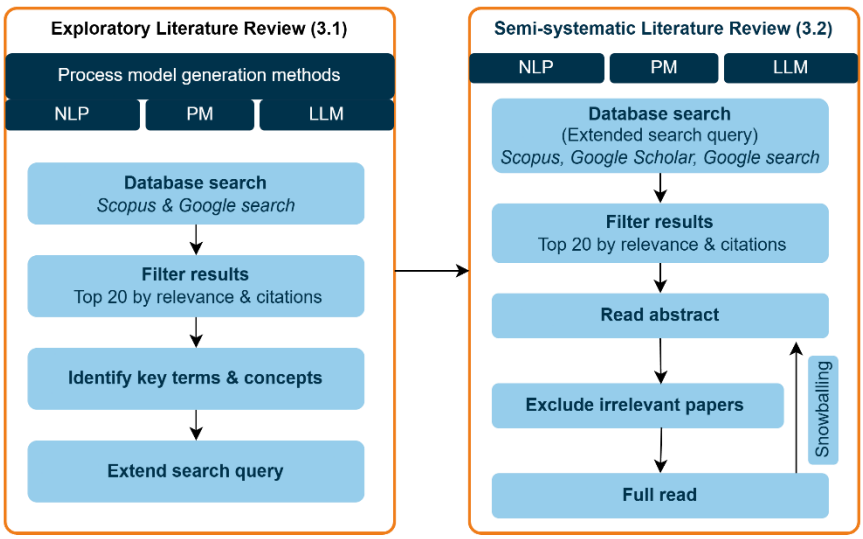


Figure 5 - Visual representation of the literature review methodology

- Section 3.1 explains the exploratory literature review for this chapter, summarising the initial process through which the corpus was obtained.
- Sections 3.2, 3.3 and 3.4 then discuss three main families of methods in more detail: NLP-based generation, Process Mining-based generation, and Generative AI / LLM-based approaches, including relevant sub methods such as prompt engineering and multi-agent systems.
- Section 3.5 describes evaluation approaches for (generated) process models, covering both quantitative metrics and qualitative assessment strategies.
- Section 3.6 synthesises these findings, compares the methods against the requirements from Chapter 2, and identifies which approaches are most suitable as a backbone for the process modelling assistant that will be designed and validated in the subsequent chapters.

3.1 Exploratory review

An initial exploratory literature review was conducted to identify core terminology and concepts related to business process model generation (SQ2-SQ3). The goal of this phase was not to be exhaustive, but to explore the available literature, discover recurring topics, and construct extended search queries for the semi-systematic review. The Scopus database was used because it provides a wide range of peer-reviewed sources to support initial exploration of important works. Using initial search terms related to the to-be-explored topic, Scopus and the Google search engine were queried for related literature. The initial search query of:

(process modelling OR process management OR business process model) AND (generation method)

Yields 912 results, of which in the Top 20 by relevance or citations there were two literature reviews on process model generation methods. Distinctions are made between the input of the generation methods, consisting of natural text, codebases, event logs, other business models, tables, or ontologies (Wiśniewski et al., 2019) (Schüler & Alpers, 2024). Different identified methods for process model generation include natural language processing, process mining, and large language models / generative AI. Obtaining information out of existing process models, rather than the inverse, is also a recurring topic. Due to the clear distinction between different approaches in the results, further exploration is needed on each of the discovered methods.

Natural Language Processing (NLP)

The initial search query of:

(natural language processing OR nlp) AND (process model) AND (generation)

Yields 52 results. All of them are scanned for relevance by title and abstract. Friedrich et al. (2011) is often cited and returns as a foundational piece of literature in most results. Key terms such as “tokenization”, “lexical analysis” or “syntactical analysis” occur as steps of the NLP process. Inserting the same search query into the Google search engine provides additional scientific sources on process model generation

using NLP, and more general explanations of NLP. To conclude, the original search query together with snowballing is deemed sufficient to deliver enough of a comprehensive overview for the semi-systematic literature review.

Large Language Models (LLM)
The initial search query of:

(generative ai OR large language model) AND (process model) AND (generation)

Yields 36 results. Nine of these discuss applying generative AI to generate process models from textual descriptions. Others focus on using large language models to explain existing process models. Results are rarely cited further, and mostly result from recent conference proceedings, necessitating caution to their validity. However, many papers claim high accuracy of their models and conclude high potential in the technology. Therefore, it should be explored further in the semi-systematic literature review. Recurring features of process model generation methods with generative AI include “fine-tuning”, “prompt engineering”, “multi-agent systems”, and “Retrieval-Augmented Generation”. Inserting the same prompt into the Google search engine results in further scientific documents, articles by large software providers such as SAP, IBM, or Microsoft on the use of Generative AI in their own applications, and blog posts by communities such as datascience-pm.com on definitions and best practices. The content in these results ranges widely from general explanations and definitions to in-depth frameworks and implementation guidelines. To incorporate the discovered techniques in the semi-structured literature review, the search query is extended to:

(generative ai OR large language model) AND (process model) AND (generation) AND (fine tuning OR prompt engineering OR retrieval augmented generation OR rag OR multi-agent systems)

Process Mining (PM)
The initial search query of:

(process mining) AND (process management)

Yields 22993 results. This is significantly more than the other topics. However, the goal of this step was not exhaustive coverage but orientation toward foundational work. Therefore, the same filtering approach was used to quickly identify the most recurring and influential sources. A scan of the initial 20 results for both citations and the search engine’s metric of relevancy revealed foundational literature in the works of W. Van der Aalst, contributing as main author on both important early works (Van Der Aalst et al., 2004) and recent summarizing works (Van der Aalst et al., 2012). A very distinctive categorization is made between “process discovery”, “conformance checking”, and “enhancement” for different process mining techniques. Additionally, Van der Aalst’s *Process Mining Data Science in Action (Second Edition, 2016)* details more practical process mining methods such as the α-algorithm, heuristic mining, genetic process mining, region-based mining, and inductive mining. Inserting the same query into the Google search engine provides additional scientific literature, as well as further insights into the practical implementation of process mining. The Process and Data Science group at RWTH Aachen University (Process Discovery - Process Mining, n.d.), mentions additional process mining methods including fuzzy mining, declarative process mining, hybrid mining

techniques, clustering-based process mining, and the Split miner method (Augusto, 2017). Pre-packaged solutions such as from ERP software can implement process mining with ease, as they have access to the event logs across the entire process. However, data scattered across systems, and the rapid change of processes in small organisations can inhibit process mining from being used effectively (IBM, 2021). For the semi-systematic literature review, it will be important to focus on the fundamentals of process mining and whether it is applicable to the context of OrangeSpot. To conclude, the original search query is deemed sufficient to provide a fundamental overview of process mining.

Evaluation methods

Unlike previous sub-questions, no distinct search query was constructed for this topic, as the evaluation of business process models typically depends on the type of generation method used. Instead, relevant insights were gathered during the analysis of literature retrieved for SQ2 and SQ3. Retrieved papers with a practical implementation of model generation were further analysed to discover various evaluation methods and metrics.

Process mining offers a well-established set of quantitative evaluation metrics for assessing the quality of generated models based on event logs. Distinctions are made between conformance checking and enhancement. Conformance checking is verifying event logs against existing models, and enhancement refers to optimizing additional metrics such as throughput times (IBM, 2021). Buijs et al. (2014) defines metrics that are commonly used such as fitness, precision, generalization, and simplicity. Beyond process mining, other quantitative approaches include distance-based evaluation techniques, where generated models are compared against a reference or "ideal" model. Skobtsov and Kalenkova (2019) describe formal metrics for measuring the structural and behavioral differences between models, which are partially used by Han et al. (2024).

In addition, more subjective forms of evaluation are found in industry practice. For example, Nour Eldin et al. (2025) propose expert-driven qualitative scoring methods, in which domain experts assess the usability, accuracy, and relevance of generated models. These methods allow for contextual considerations and qualitative feedback, which are often necessary when evaluating models meant for specific organizational use cases. Finally, comparative assessments, such as those used by Klievtsova et al. (2025), rank different models on various metrics to evaluate subjectively. These assessments are particularly relevant when selecting a process model generation method based on the specific needs and constraints of a target organization.

3.2 RESULTS

Historically speaking, the oldest method of generating a business process model is by hand. Process models can be designed in many different languages, abstraction levels, contexts, and can serve various goals restricted by different kinds of constraints (Van der Aalst, 2011). What makes a good, understandable, and useful process model is often dependent on capturing the alignment between an ideal and reality. Therefore, manually creating process models is, within the BPM community, generally considered more of an art than an exact science (Van der Aalst, 2016). To tackle this, standardized modelling notations have been created such as Petri nets, and most notably the Business Process Model and Notation (BPMN). BPMN provides a formal, graphical language for representing business processes through a consistent set of elements such as activities, gateways, and events. Its standardization enables business professionals to communicate processes unambiguously across teams and organisations, while also

supporting scientific rigor by allowing models to be compared, analysed, and executed in a uniform way. As a result, BPMN has become the standard for process modelling, forming the foundation upon which both traditional and automated model generation methods build. Using standards like BPMN, several process model generation methods have been developed in scientific literature like Natural Language Processing (NLP), Process Mining (PM), and generative ai or large language models (LLMs). Firstly, NLP mainly focuses on standardizing the process through which models are created from unstructured input. Secondly, PM focuses on standardizing the input through the utilization of event logs. Finally, LLMs are emerging as a new trend in process model generation methods. While process modelling through generative ai is still in its infancy, the scientific community acknowledges the potential of this new technology (Grohs et al., 2024). The following section of the semi-systematic literature review describes these three process model generation methods in further detail.

3.2.1 Natural Language Processing

Process modelling experts that work by hand essentially translate the domain-specific information they receive to a modelling language. Doing this correctly requires domain expertise, significant time investment, and is prone to inconsistencies due to subjective interpretation (Mendling et al., 2019). Back in its infancy, early studies indicated that acquiring as-is models can consume up to 60% of project time (Herbst & Karagiannis, 1999), suggesting that automated methods offer significant potential for efficiency gains. NLP has since emerged as an approach to automate business process model generation, addressing these limitations through predefined linguistic rules (Friedrich et al., 2011).

Creating a process model starts with some form of input. With natural language processing, this often comes in the form of an unstructured piece of text that describes a business process. In some cases, like in (Ackermann & Volz, 2013) and (Ferreira et al., 2017), process description templates are used to remove some of the ambiguity. However, in case already existing descriptions or documents are used, these templates will not be adhered. Therefore, there are various methods to extract process information from unstructured natural text.

The extraction process typically begins with tokenization. This is an NLP technique where text is divided into smaller units (tokens) and reduced to their simplest form to prepare them for further analysis. Common ways to do this is through n-grams and Part-Of-Speech (POS) tagging:

- *N-gram tokenization* involves breaking text into sequences of n words, called n-grams, to analyse frequently occurring sets of words in a corpus. A unigram consists of a single word (e.g., "process"), a bigram consists of two consecutive words (e.g., "process model"), and a trigram consists of three words (e.g., "business process model"). By recognizing frequently occurring n-grams, this method helps identify meaningful phrases and improve the accuracy of following text processing tasks. (Khurana et al., 2023)

After this initial preparation, a lexical analysis of the tokens can be performed to gain better understanding of the words' meanings. Examples of such techniques include:

- *Stemming* is the process removing prefixes or suffixes, often using simple predetermined rules. For example, the word "running" might be stemmed to "runn", which is not always a valid word. A commonly used stemmer is Porter's Stemmer (Chowdharry, 2020).
- *Lemmatization* reduces a word to its dictionary form while considering its meaning and grammatical role. Using the same example, "running" would be lemmatized to "run".

- *Part-of-Speech tagging* is the process of assigning grammatical categories, such as noun, verb, adjective, or adverb, to words in a sentence based on their context and usage. For example, in the sentence "She runs fast", "runs" would be tagged as a verb.
- *Anaphora resolution* "involves the identification of the concepts which are references using pronouns ("we","he","it") and certain articles ("this", "that")" (Friedrich et al., 2011). It is important to mention that this does not necessarily occur within a single sentence, but between sentences. Finding all expressions that refer to a single entity in a text is called *Coreference Resolution*.

Subsequently, the lexically augmented tokens can be further processed for syntactical meaning, using one of the following techniques:

- *Named Entity Recognition* (NER) is used for information extraction to recognize name entities and then classify them to different classes (Khurana et al., 2023).
- *Shallow parsing / Chunking* is a technique that identifies and groups syntactically related words into meaningful units. Unlike deep parsing, which analyses the complete grammatical structure of a sentence, shallow parsing focuses on extracting key phrases, such as noun phrases (NPs), verb phrases (VPs), or prepositional phrases (PPs), making it computationally more efficient.
- *Regular expressions* (regex) are patterns used to match, search, and manipulate text based on predefined rules. They enable efficient identification of specific sequences such as dates or capitalized words. Regular expressions are often used to enable Named Entity Recognition or Chunking.

After tokens have been augmented with additional information, deductions can be made to discover their semantic meanings in the process modelling context. Gonçalves et al. (2011) refers to this as Domain Analysis. Their approach, as well as that in (Ghose et al., 2007), is to match the correspondences between the scenario model and grammar elements with each other, such that rules can be constructed such as: "*Activity: Sentence containing a Verb Phrase"* or "*Action: A verb Phrase at an Activity"*. A similar approach is taken in Sonbol et al. (2023), where a *semantic transfer* is used. The BPMN language is deconstructed to the semantic meanings of its elements and relations, after which tokens, chunks, or sentences can be semantically transferred to these meanings. After this, a model can be reconstructed back to a BPMN model.

Despite the promising potential of Natural Language Processing (NLP) in process extraction, several significant disadvantages hinder its widespread adoption and effectiveness. One of the primary challenges is the ambiguous nature of natural language. Words and phrases can have multiple meanings depending on their context, making it difficult for NLP tools to consistently extract accurate process models. This is further emphasized by Gonçalves et al. (2011), stating that "the quality of the process model in terms of completeness and correctness is directly dependent on the textual material provided by the tellers of the stories". Additionally, different writing styles and variability across domains further complicate the task. Texts describing processes may vary in formality, structure, and terminology, which makes the extraction process highly sensitive to the specific characteristics of the input data. (Bellan et al., 2020)

Another key limitation is the lack of recent research in the field. Many of the contributions to NLP-based process extraction are based on work from many years ago, meaning they may be outdated and not reflective of current advances in NLP technology or process modelling

practices. As a result, existing approaches may fail to incorporate recent advancements in machine learning, making them less efficient or inaccurate in modern applications.

Finally, evaluating generated process models is challenging due to the lack of a definitive standard in the field for what constitutes a valid or high-quality text-to-model output. In the absence of a validated benchmark, it becomes difficult to measure the effectiveness of NLP methods compared to each other. This is most emphasized in Bellan et al. (2020), indicating that the most modern techniques are “highly tailored to the input data”, and “the inability of existing tools to scale to real-world scenarios”. In conclusion, this makes most research difficult to generalize.

3.2.2 Process Mining

Process mining has recently been one of the most prominent sub-streams in BPM research (Reijers, 2021). It comprises a set of methodologies that have evolved to extract and analyze business process information from event data. Initially, process mining starts with process discovery. the most popular techniques include (Van der Aalst, 2016):

- *The Alpha Algorithm* establishes causal dependencies between events in an event log to construct a process model, typically in the form of a Petri net. It identifies sequences based on direct succession and excels in causal dependencies. However, it is limited in handling concurrency, noise, and incomplete data. As a result, it produces accurate models only when applied to structured, well-defined event-logs.
- *The Heuristic Miner* improves upon the Alpha Algorithm by incorporating frequency-based analysis to filter out infrequent or erroneous behaviors. This approach allows it to handle noise and discover parallelism, making it more applicable to real-world event logs. However, the model’s quality depends on user-defined thresholds, and potentially leads to inconsistencies or inaccuracies.
- *The Inductive Miner* addresses these issues by using a recursive, divide-and-conquer strategy to derive structured and sound process models. It systematically decomposes an event log into smaller components, ensuring correctness while supporting concurrency and loops. Despite its advantages, it enforces a structured representation that may not always align with highly unstructured processes, and its computational requirements increase with complex datasets.

Following this discovery phase, recorded behaviour can be compared to existing process models to identify deviations and for instance verify compliance with predefined process regulations. This is called conformance checking, and is done with methods such as (Van der Aalst, 2016):

- *Token-based replay* is a technique where an event log follows the path of a predefined petri-net. As it follows the available steps, it keeps track of any deviations such as when a defined activity cannot be performed, or how many remaining tokens are left at the end of the simulation. Based on such metrics, it calculates a fitness-score.
- *Alignments* is a technique, which performs an exhaustive search to find out the optimal alignment between the observed trace and the process model. Hence, it is guaranteed to return the closest model run in comparison to the trace. (Process Mining Group, n.d.)

Finally, process enhancement techniques extend or refine the initial models by incorporating additional performance metrics (i.e. waiting times or resource utilization) to support systematic process improvement and optimization. By utilizing all the previously mentioned techniques,

these approaches facilitate a comprehensive cycle of understanding, verifying, and enhancing business processes. (Van der Aalst, 2016).

The IEEE Task Force on Process Mining, a research initiative with over 70 experts from academia and industry that promotes the development, application, and standardization of process mining techniques, has created a “Process Mining Manifesto”, outlining key advantages, guiding principles, and important challenges. One of its key advantages is its ability to provide objective insights into the real-world execution of processes, revealing inefficiencies, deviations, and opportunities for improvement. Therefore, process mining supports evidence-based decision-making and reduces reliance on subjective assumptions. However, its application is constrained by challenges related to data quality, availability, and integration. Event logs must be complete, well-structured, and properly recorded, which is often difficult in consulting environments where client systems may vary widely. Additionally, ensuring data privacy further complicates implementation. While process mining remains a valuable tool for process improvement, its reliance on high-quality input data makes it less practical as an initial step in projects where standardized event logs are not readily available. (Van der Aalst et al., 2012)

3.2.3 Large Language Models

Generative Artificial Intelligence, or GAI for short, is defined as “computational techniques that are capable of generating seemingly new, meaningful content such as text, images, or audio from training data“ (Feuerriegel et al., 2024). Often referred to as Large Language models (LLMs), they have recently emerged as a novel and promising approach within the field of business process modelling (Fill et al., 2023). In contrast to more established methods such as NLP and PM, GenAI is not focused on rule-based extraction from unstructured text or analysis of structured event logs. Rather, it learns patterns and relationships from a large dataset to create something similar, but new and original (Daclin et al., 2024).

Many related studies acknowledge the potential of GAI for Business Process Management. A relatively early study by Vidgof et al. (2023) studied the opportunities of GAI projected on the BPM Lifecycle, a popular methodology for designing business processes, and found potential applications in every step, especially in “extracting process-relevant information from textual sources for modelling purposes”. Similarly, Klievtsova et al. (2023) identified gathering process descriptions, relevant activities, actors, and relations as potential applications of GAI. The potential of generative AI lies in its ability to lower the technical barriers of process modelling, enabling a faster and more inclusive design process (Grohs et al., 2024). Early studies report encouraging results, with LLMs demonstrating the ability to generate syntactically correct and semantically relevant process models based on minimal input (Fill et al., 2023). However, much of the existing research remains exploratory in nature, often limited to controlled experiments or simplified use cases (vidgof et al., 2023) (Klievtsova et al., 2023).

A common problem within Generative AI is the occurrence of so-called “hallucinations”, which can be defined as “the generated content that is nonsensical or unfaithful to the provided source content” (Ji et al., 2023). These can be distinct in the form of intrinsic hallucinations, stemming from the model's internal biases or limitations, and extrinsic hallucinations, where the output contradicts or deviates from external information. In the context of process model generation, it is difficult and unpractical to gather a large dataset of text descriptions and process models to fine-tune a new model and try to prevent hallucinations. Therefore, modern GAI-based generation methods are almost exclusively based on the Generative Pretrained

Transformer (GPT), which allows models to generalize across a wide range of tasks without task-specific training. GPTs have reduced the number of hallucinations significantly (Petroni et al., 2019). However, there are more advanced techniques to further eliminate the odds of hallucinations. *Prompt engineering* is optimizing the input of a model to provide better context, *multi-agent systems (MAS) use* multiple models in concert with each other to distribute specific tasks, and *Retrieval Augmented Generation (RAG)* models force the model to retrieve its answer from a predetermined knowledge base, guaranteeing results are grounded from screened information.

Fine-tuning

A common disadvantage of large language models is overgeneralization. LLMs often generate false answers by mimicking common misconceptions found in their training data. In domain-specific applications, pre-trained models potentially benefit more from additional training of their parameters through domain-specific examples, rather than the hyper scaled all-purpose models (Lin et al., 2022). Therefore, popular large language model providers such as OpenAI, Google, or Meta provide methods to calibrate the model parameters, also known as fine-tuning. Fine-tuning can be done in different ways, and is differentiated by OpenAI into three different methods (Fine-tuning - OpenAI API, n.d.):

- *Supervised fine-tuning* makes use of human-provided truthful input-output combinations to backpropagate the model parameters. This is especially applicable in use cases of classification tasks and generation of outputs in specific formats.
- *Direct Preference Optimization* is the technique of providing both a correct and incorrect answer to an input-prompt. This is particularly useful for directing the output of a model to subjective preferences. Making sure summaries focus on the right things and writes in the correct tone and style.
- *Reinforcement Finetuning* defines a scoring method to the model's output, which it can then optimize like traditional reinforcement learning practices. This method requires a thorough understanding of the ideal output of the model and finds applications in well-established domains such as law and healthcare.

Fine-tuning an all-purpose model can greatly increase its domain-specific knowledge. However, fine-tuning depends on the availability of sufficient truthful training data. Meta advises to aim for a minimum of around a hundred examples. However, it acknowledges that even around twenty examples can achieve reasonable results. Additionally, fine-tuning on all model parameters, which can run into the billions, can be computationally very expensive. To tackle this, Parameter Efficient Fine Tuning (PEFT) can be used. Common techniques include Low-Rank adaptation (LoRa) and Quantized Low-Rank adaptation, which only adjust a portion of the parameters, and BitFit, only adjusting the biases of the model. It is generally considered best practices to start with LoRa, or QLoRa in the case of very limited computational power, and scaling up to full-parameter fine-tuning when this is deemed insufficient (*Fine-tuning | How-to Guides*, n.d).

Prompt engineering

In cases where there is insufficient training data, in both quantity and quality, fine-tuning is often not an option. Additionally, in cases where the input of the model is highly variable or dependent on the end-user, input quality cannot be guaranteed. Instead of fine-tuning the model parameters, the input of the model can then be steered using what is called "prompt engineering": an approach that provides structured or contextual input to elicit targeted responses (Vatsal & Dubey, 2024). This approach is also referred to as "in-context learning"

(Bellan et al., 2022). Compared with NLP methods, GPT solutions tend to do better out-of-the-box, even without extensive prompt engineering (Grohs et al., 2024). However, performance can be further increased by applying different prompt engineering techniques:

- *Few-shot prompting* refers to the practice of providing GAI with examples to guide the model's behavior. Opposingly, *zero-shot learning* does not provide examples. In general, few-shot learning is preferred (Grohs et al., 2024). However, when more creative freedom is required, it might only restrict the model (Brown et al., 2020)
- *Chain-of-thought prompting* refers to introducing a series of intermediate reasoning steps to the model, potentially increasing complex reasoning possibilities. (Wei et al., 2023)
- *Role prompting* or *expert prompting* is establishing a clear role for the LLM as an expert, leading to a higher quality output (Xu et al., 2025).
- *Least-to-most prompting* is a prompt engineering technique introduced by Zhou et al. (2023). Instead of presenting the full problem upfront, this method decomposes it into a sequence of incrementally more difficult subproblems. The model solves each subproblem in order, using earlier answers to inform subsequent steps.

Prompt engineering is very popular in recent GAI-based process model generation techniques. For instance, ProMoAI (Kourani et al., 2024), a popular paper recent similar implementations often compare themselves to. They use few-shot prompting, role prompting and least-to-most prompting in their implementation. Furthermore, ontologies can be provided to the prompt to reduce the amount of tokens required (Köpke & Safan, 2024).

To secure the syntactical validity of the output created by Generative AI models, an intermediate representation is often used. PromoAI makes use of an intermediate representation with the Partially Ordered Workflow Language (POWL) (Kourani & van Zelst, 2023). POWL is a hierarchical modelling language, meaning it can provide guarantees on the validity of the provided output. (Kourani et al., 2024). Other forms of intermediate representations include petri-nets as often used in process mining (Beheshti et al., 2023), or simply JSON, which can then be translated into the desired process model syntax through predefined functions (Köpke & Safan, 2024; Nour Eldin et al., 2025; Kourani et al., 2024). Despite the promising capabilities of Generative AI in combination with prompt engineering to automate process model generation, several challenges remain that hinder practical application. A key concern is the quality and structure of prompts, which heavily influences the output. Issues such as process representation in prompts, limited context windows, and prompt transferability across use cases remain challenging (Busch et al., 2023). While LLMs have shown strong performance in generating syntactically valid models (Fill et al., 2023), their understanding of complex dependencies such as those involving intricate message flows remains limited (Lin et al., 2024). Additionally, data privacy concerns arise when uploading sensitive business processes to public LLM platforms (Berti et al., 2024), pushing the need for secure, local deployment of language models. Foremost, research is still exploratory, relying on narrow and simplified scenarios (Vidgof et al., 2023), limiting real-world generalizability.

Retrieval Augmented Generation

Large Language Models (LLMs) such as GPT-4 are trained on extensive corpora of text and can generate coherent and contextually appropriate responses. However, these models are inherently statistical and closed in nature. They generate outputs based on learned

probabilities rather than querying a live, up-to-date knowledge source. This leads to three key limitations (Martineau, 2024):

- *No Grounded Sources*: Generated outputs are not tied to verifiable documents or references, which undermines reliability.
- *Outdated Knowledge:* LLMs are only as current as their training data. For example, a model trained on data up to 2023 cannot provide accurate updates on events occurring afterward.
- *Hallucinations and Overconfidence:* LLMs will produce an answer, even if they do not know. This may cause the LLM to hallucinate false information, rather than saying it does not know.

These drawbacks limit the use of general-purpose models in knowledge-intensive applications, where up-to-date, verifiable, and context-aware information is essential (Martineau, 2024).

Retrieval-Augmented Generation, or RAG for short, addresses the limitations of LLMs by integrating an external knowledge retrieval mechanism. It augments the generation process with relevant documents fetched from a knowledge base (typically a vector store), allowing the model to generate outputs grounded in external, up-to-date sources (*Retrieval - OpenAI API*, n.d.). The foundational work on RAG was introduced by Lewis et al. (2020), in which the authors combined dense retrieval (via DPR) with generative models (like BART) to create a system capable of open-domain question answering with verifiable results. In state-of-the-art systems, a "baseline RAG" pipeline consists of:

1. *Query encoding:* Encodes the provides input query to a vector space in preparation for retrieval.
2. *Retrieval:* Using k-nearest neighbours, this step retrieves relevant documentation from a predetermined vector store, and outputs a set of relevant documents or passages. This step is what essentially guarantees the output of the pipeline to directly cite sections of the document.

Because the input query is matched to the nearest equivalent vector of the knowledge base, these initial two steps essentially guarantee the output of the pipeline to directly cite parts of the documents within it. To then select the most relevant part of the retrieved documents, the RAG pipeline uses very similar approach to a traditional pretrained transformer:

3. *Document encoding:* Encodes the selected passages conform a standard transformer encoder in preparation for decoding. Outputs a hidden layer of encoded passages to be handled by the decoder.

Traditionally by Lewis et al. (2020), the input query consists of a single piece of text. A variant on these two initial steps in the pipeline is RAG-Turn, which runs step 1 and 2 for each turn of dialogue instead. Additionally, each passage-turn pair is re-encoded every turn to incorporate new dialogue context as new turns occur (Shuster et al., 2021). RAG-Turn is preferred when a user participates in dialogue multiple times, refining the output.

4. *Decoding:* Decodes the list of encoded passages to output the next token in the sequence. Separate from whether you are using baseline RAG or RAG-turn, this step has several variations (Shuster et al., 2021):
   - *RAG-Sequence* generates a separate token for each passage and selects the most probable token. This technique is used when high precision is deemed important, as tokens are not considered across documents.
   - *RAG-Token* backpropagates and iterates over all passages, and outputs a single token. This can increase output quality, as the decoder is fine-tuned during decoding. This technique is preferable when generalization is deemed important, such as with summaries.
   - *Fusion-in-Decoder:* fuses the encoded passages into one object before decoding, allowing RAG to process large knowledge bases at the cost of retriever fine-tuning. This technique is used when the model is expected to scan through large amounts of documents.

Following this pipeline and choosing the right techniques for domain it is applied to, Shuster et al. (2021) have shown to significantly reduce the hallucination problem, while maintaining conversational ability. However, significant challenges with baseline RAG remain. For instance, it still struggles to "connect the dots", interpreting the semantic meaning of its outputs and apply complex reasoning tasks to it. Additionally, "Baseline RAG struggles with queries that require aggregation of information across the dataset to compose an answer" (Potts, 2024). To tackle these issues, Microsoft Research has developed a new approach using an LLM to create a knowledge graph of the dataset: GraphRAG. This knowledge graph can be used to provide prompt augmentation to the model at query time, facilitating a better semantic understanding. GraphRAG transforms large datasets into a knowledge graph of entities and relationships, allowing LLMs to produce coherent and thematically consistent responses (W&B, 2025). GraphRAG has found successful application in summarizing large corpa of podcast transcripts and news articles (Edge et al., 2025). Additionally, it outperforms regular RAG models in the medical and legal domain (Wu et al., 2024; Zhao et al., 2024; (Zhang et al., 2025).

While several enterprise platforms (Appian, 2025.; UiPath, 2025.; Pega, 2025; Celonis, 2025) explicitly use Retrieval-Augmented Generation (RAG) for grounded responses in business process management, none currently use RAG to directly generate structured business process models such as BPMN diagrams. RAG is instead applied to question answering, documentation synthesis, and insight generation. Furthermore, no applications have been found using GraphRAG in process management.

Multi-Agent Systems (MAS)

Despite their impressive performance in a wide range of natural language tasks, LLMs struggle with complex, layered tasks due to limitations in planning, context alignment, and reasoning. A single LLM often lacks the reliability required for complex workflows, especially when tasks involve verification, long-term consistency, or multi-step reasoning (Han et al., 2024). LLMs do not inherently reason well about their own outputs (Potts, 2024). This issue is amplified when models are tasked with multi-turn generation, planning, or decision-making without intermediate checks or structured feedback mechanisms. As we have seen, these challenges can be solved in the presence of a factual knowledge base. However, when applied in unexplored domains or in the absence of elaborate expert reports, other paradigms must be considered.

A solution to this challenge is found in multi-agent systems, also referred to as agentic AI. A multi-agent system in the context of generative AI is composed of multiple interacting agents that collaborate to solve tasks through distributed responsibilities. These agents may assume specialized roles such as planner, generator, fact-checker, or critic, and work either hierarchically, in parallel, or through dynamic negotiation (Dmitrievna, 2025). The agentic structure draws inspiration from human collaboration and division of labour, where complex goals are broken down into subgoals distributed across individuals or teams with varying expertise (Tan, 2025).

Anthropic, one of the leading companies in applied AI with their "Claude" model series, draws an important distinction between workflows and agents. "Workflows are systems where LLMs and tools are orchestrated through predefined code paths. Agents, on the other hand, are systems where LLMs dynamically direct their own processes and tool usage, maintaining control over how they accomplish tasks" (Anthropic, n.d.). A key feature of agentic AI is the autonomous orchestration of execution of actions through leveraging the reasoning abilities of LLMs.

Different nodes or agents in AI multi-agent systems can take on several structures (Han et al., 2024):

- *Hierarchical structures* place a single agent responsible for the allocation and division of tasks. It delegates the necessary actions sequentially to more specified sub-level agents, and coordinates their outputs to form a coherent response. Using hierarchical structures simplifies issues with memory management in multi-agent systems, maintaining a ground truth.
- *Equi-level structures* have agents operate at the same hierarchical level, meaning no agent has authority over the others. Each agent has its own role and strategy, but no agent has definite preference. Depending on their objectives, the agents may collaborate, negotiate, or debate. Using Equi-level structures allows for a very dynamic environment, where agents can respond to rapid changes in urgency and priority.
- *Nested structures* combine both previously mentioned structures, where parts of an application may require different approaches.

Separating responsibilities of different tasks using multi-agent systems allows to combine the best of different optimization methods. For instance, separate agents can be conditionally prompt engineered, utilize different RAG techniques depending on usage, or be fine-tuned without affecting other parts of the system. Lin et al. (2024) demonstrate the advantages of multi-agent systems in process model generation in MAO: A Framework for Process Model Generation with Multi-Agent Orchestration. MAO uses a nested structure of three separate agents, a team leader responsible for business context, a process design expert for modelling tasks, and a process reviewer for semantic hallucination detection. The different agents are sequentially in discussion with each other, constantly refining the result. Additionally, agents are augmented with prompt engineering techniques such as few-shot prompting and chain-of-thought prompting. Finally, the result is run through external tooling to verify the validity of the process model syntax. This approach consistently outperforms the previously considered state-of-the-art by PromoAI (Kourani et al., 2024) and manual modelling. However, it still struggles to understand large amounts of messaging flows (Lin et al., 2024).

3.2.4 Evaluating process models

As the field of process model generation continues to evolve, rigorous evaluation becomes essential to assess the accuracy, reliability, and usability of generated models. Evaluation methods can be broadly categorized into quantitative and qualitative approaches, each contributing distinct insights into model quality. This section outlines the state-of-the-art techniques and considerations for evaluating generated process models quantitatively and qualitatively.

Datasets

Standardized annotated datasets of process models and descriptions can help scientific researchers to compare performance across different studies. Among the explored literature, one dataset saw recurring usage: The PET dataset (Bellan et al., 2022). The PET dataset is a collection of textual descriptions and annotated BPMN process models created to help researchers test and compare methods for process modelling and prediction. Although the logs are fictitious, they are built to look like real business processes, including elements like loops, choices, and parallel activities. No other datasets have been discovered in the search space of this paper.

Quantitative evaluation

Quantitative evaluation techniques offer measurable and repeatable metrics to assess the similarity, correctness, and quality of generated models compared to predefined benchmarks or reference models. Notably, the field of Process Mining has developed metrics suitable to the process mining methodology specifically.

A foundational framework for assessing the quality of process discovery methods is provided by Buijs et al. (2014), which identifies four key dimensions:

- *Fitness* measures how well the model can reproduce observed behaviour in the event log. A model with high fitness can replay most or all of the cases seen in the data.
- *Precision* ensures that the model does not allow for too much behaviour beyond what was observed, overgeneralization is penalized.
- *Generalization* addresses how well a model can handle unseen but plausible cases, preventing overfitting to specific traces in the log.
- *Simplicity* promotes models that are easy to understand and not overly complex, aligning with the principle of Occam's Razor.

These dimensions are especially relevant in scenarios where process models are discovered from event logs. Popular tools to facilitate such evaluations include ProM (Prom Tools, n.d.) and PM4Py (PIS, n.d.).

Another quantitative approach focuses on measuring structural similarity between the generated process model and a previously determined ground truth. Several methodologies exist to come to such a metric. Skobtsov & Kalenkova (2019) provide an overview of these techniques, with the *Graph Edit Distance (GED)* being the most popular. The GED calculates the minimum number of operations (insertions, deletions, substitutions) required to transform one model into another. This technique is particularly suitable when models are expressed in graph-based notations such as BPMN or Petri nets. State-of-the-art variations include the Dijkman en Montani variant, which have been enhanced further by Waspada & Sarno (2020) to incorporate differentiation in gateway structures. Alternative distance metrics are A*, Greedy, Tabu search, simulated annealing, ant colony optimization., which can be applied using standardized software packages such as BPMNDiffViz.

Qualitative Evaluation
Quantitative measures alone are insufficient to assess model usability, domain relevance, or user satisfaction. Qualitative methods address these gaps by incorporating expert judgments and subjective preferences.

In contexts where no ground truth is available, or where interpretability is key, expert evaluation can provide a valid alternative to evaluate generated process models. Nour Eldin et al. (2025) propose an evaluation framework as part of their NALA2BPMN system, where experts rate models based on:

- *Understandability:* How easily stakeholders can comprehend the model.
- *Correctness:* Whether the model accurately reflects the described process.
- *Completeness:* The extent to which all relevant process elements are included.
- *Relevance:* Whether included elements are pertinent to the process.
- *Ease of Updating:* How easily the model can be adapted when changes occur.

Such dimensions emphasize practical applicability over formal correctness and are particularly important when models are intended for collaborative settings involving non-technical users.

A complementary approach to expert interviews is comparative assessment. Klievtsova et al. (2025) conducted a large-scale study in which 40 participants were asked to choose between human-generated and AI-generated process models. The models were evaluated based on perceived value, including trustworthiness, clarity, and preference. This method captures implicit human preferences that may not be reflected in formal correctness metrics. These insights are particularly valuable in design and early prototyping phases, where user satisfaction and stakeholder buy-in play critical roles in model acceptance.

3.3 REQUIREMENTS FIT

Methods & techniques
Traditional NLP-based generation methods do not fit these requirements well. They typically assume relatively structured, unambiguous textual descriptions, whereas onboarding input at OrangeSpot is highly variable in quality, completeness and writing style. In such conditions, rule-based NLP pipelines are fragile: they tend to break down on informal phrasing or inconsistent terminology, leading to low accuracy and unreliable model extraction.

Process Mining is similarly unsuitable in the onboarding phase. It presupposes the availability of complete, system-recorded event logs from which behaviour can be reconstructed. At the start of an OrangeSpot implementation, such logs are generally not available: users have not yet modelled their processes, and many client systems do not record events at the required level of granularity. As a result, PM methods cannot satisfy the requirements for automatic model generation, iterative refinement, short interaction times and fast onboarding in this context.

Given these constraints, Generative AI methods emerge as the most promising category. They can work directly with informal, incomplete user input and support interactive, dialogue-based refinement, which aligns well with the requirements for rapid suggestion generation, reduced “blank canvas” effects and improved perceived usefulness.
Within this category, several sub-methods can be distinguished:

- **Prompt engineering** is highly suitable as a primary mechanism. Carefully designed prompts can steer a general-purpose LLM towards producing structured, domain-aligned outputs (e.g. blueprint-like process descriptions) with minimal setup overhead. This keeps the

solution lightweight while still accommodating OrangeSpot-specific terminology and formatting requirements.

- **Multi-Agent Systems (MAS)** are also promising as an extension. By assigning complementary roles such as planner, generator and reviewer to different agents, MAS can emulate aspects of a consultant's reasoning process across multiple turns. This can improve coherence and robustness of the generated models, particularly in longer or more complex onboarding interactions.

By contrast, other Generative AI sub-methods are less suitable in the current situation:

- **RAG and GraphRAG** depend on the existence of a rich, structured knowledge base that can be indexed and queried at generation time. OrangeSpot does not yet possess such a domain-wide, stable corpus for onboarding scenarios, which means the additional infrastructure and maintenance effort of RAG-based solutions would offer limited practical benefit.
- **Fine-tuning** would require a curated dataset of many high-quality example models and inputs (on the order of tens to hundreds of cases). At present, such a dataset is not available, and collecting and maintaining it would be costly. Given the diversity of small clients and their idiosyncratic processes, the return on investment of fine-tuning is uncertain compared to prompt-based approaches.

In summary, a Generative AI-based solution that is primarily driven by prompt engineering, optionally complemented with a lightweight Multi-Agent setup for reasoning and validation, offers the best fit with the identified requirements. It supports automatic model generation and iterative refinement within acceptable interaction and onboarding times, while avoiding the data prerequisites and infrastructure complexity associated with NLP pipelines, Process Mining, RAG-based methods and fine-tuning.

Evaluation

The evaluation of process model generation methods must be tightly aligned with the OrangeSpot requirements. It should capture both technical output quality (e.g. correctness and completeness of models) and user experience (e.g. perceived usefulness, ease of use and preference over manual modelling). Given this dual focus, a combination of quantitative and qualitative approaches is needed:

- **Quantitative evaluation** can be used where a ground truth or reference model is available. Structural metrics such as Graph Edit Distance (GED) allow comparison between generated models and an "ideal" model by counting the minimum number of insertions, deletions and substitutions required to transform one into the other. Process mining metrics such as fitness, precision, generalization and simplicity offer another set of objective criteria when event logs and discovery algorithms are involved. These techniques are well suited for controlled experiments but are less applicable in day-to-day onboarding, where client-specific ground truths are often unavailable or costly to obtain.

- **Qualitative evaluation** therefore plays a central role in this context. Expert-based evaluation allows process experts to judge generated models on dimensions such as correctness, completeness, relevance and understandability. Comparative user assessments, in which users compare assistant-generated models to manually or consultant-created models, provide direct evidence on perceived usefulness, clarity and trustworthiness. These qualitative methods align closely with OrangeSpot's non-

functional requirements: they can be used to test whether users recognise their own processes, experience the assistant as helpful, and prefer using it over starting from a blank canvas.

Together, these evaluation strategies make it possible to assess whether the selected Generative AI-based approach not only produces technically sound process models but also satisfies the practical onboarding goals of short interaction times, reduced total onboarding duration and high user-rated usefulness and usability.

3.4 ANSWER TO SQ2-SQ4

Taken together, the exploratory and semi-systematic literature reviews show that the state-of-the-art in process model generation methods can be grouped into three main families: rule-based Natural Language Processing (NLP), Process Mining (PM), and Generative AI / Large Language Models (LLMs). NLP approaches transform textual process descriptions into models through pipelines of linguistic analysis steps such as tokenisation, part-of-speech tagging, syntactic parsing and domain-specific mapping rules. Process Mining starts from structured event logs and applies discovery, conformance checking and enhancement algorithms (e.g. Alpha, Heuristic and Inductive Miner) to reconstruct executable models and assess how well they reflect observed behaviour. Generative AI approaches, finally, use LLMs to directly generate structured process representations from informal input. Recent work combines base LLMs with techniques such as prompt engineering, intermediate representations, Retrieval-Augmented Generation (RAG), fine-tuning and multi-agent systems to improve syntactic correctness and reduce hallucinations. This classification provides a concise answer to SQ2: these three method families, and the LLM-based variants in particular, represent the current state-of-the-art in automated process model generation.

When these methods are compared against the requirements for a process modelling assistant at OrangeSpot, important differences in practical suitability emerge (SQ3). NLP-based techniques assume relatively clean, well-structured and unambiguous input texts, and much of the literature is tailored to specific datasets or templates. In contrast, onboarding input at OrangeSpot is highly variable in quality, completeness and writing style. Under these conditions, rule-based NLP pipelines are fragile and hard to maintain, which makes them a poor fit for automatic model generation and fast, interactive refinement in a production tool. Process Mining is similarly misaligned with the onboarding context: it presupposes complete, system-recorded event logs at the right level of granularity, while new OrangeSpot clients typically lack such logs at the start of an implementation. As a result, PM methods cannot satisfy the requirements for rapid first-model generation, short interaction times and limited total onboarding duration.

Within the Generative AI family, a more nuanced picture appears. Prompt-engineering-based use of general-purpose LLMs aligns well with OrangeSpot's constraints: carefully designed prompts can steer a model to produce blueprint-like process descriptions, using OrangeSpot-specific terminology, without the need for large domain-specific training datasets or complex infrastructure. Multi-Agent Systems (MAS) extend this by assigning complementary roles (e.g. planner, modeller, reviewer) to different agents, so that iterative refinement can mimic a consultant's reasoning process and catch inconsistencies over multiple turns. By contrast, RAG and GraphRAG require a rich, stable and well-maintained knowledge base, which OrangeSpot does not yet possess for onboarding scenarios, and fine-tuning demands tens to hundreds of curated training examples, which would be costly to assemble and keep up-to-

date for a diverse SME client base. Consistent with the earlier Research Topics study, this thesis therefore concludes that a Generative-AI-based assistant driven primarily by prompt engineering, optionally enhanced with a lightweight MAS layer for reasoning and validation, offers the best fit with the requirements, while traditional NLP, PM, heavy RAG setups and extensive fine-tuning are unsuitable or disproportionally complex.

Answering SQ4, the literature shows that the quality of (generated) business process models can be assessed through both quantitative and qualitative approaches, each addressing different facets of model quality. On the quantitative side, Process-Mining-based metrics such as fitness, precision, generalisation and simplicity evaluate how well a model reproduces observed behaviour in an event log, avoids over-generalisation, handles plausible but unseen cases and remains understandable. Distance-based metrics, such as Graph Edit Distance and its variants, compare generated models to a reference “ground truth” model by counting the minimal number of insertions, deletions and substitutions needed to transform one into the other, and can be implemented using tools like BPMNDiffViz. Where suitable datasets exist - for example the PET dataset of paired BPMN models and textual descriptions - these metrics support repeatable benchmarking across methods. However, such datasets are still scarce and often focus on BPMN rather than OrangeSpot’s own blueprint notation.

Quantitative measures alone are not sufficient for OrangeSpot’s use case, where no event logs are available at onboarding and there is rarely a single authoritative ground-truth model. Qualitative evaluation methods therefore play a central role. Expert-based assessments invite process management specialists to rate generated models on dimensions such as understandability, correctness, completeness, relevance and ease of updating, directly linking evaluation to the practical usefulness of models in change projects. Comparative studies, in which end-users and stakeholders choose between consultant-created, manually created and AI-generated models and rate them on perceived clarity, trustworthiness and overall preference, capture user experience factors that formal metrics cannot. In line with the earlier Research Topics conclusion, this thesis recommends a mixed evaluation strategy for future OrangeSpot prototypes: use quantitative checks where feasible (e.g. basic structural validity, limited distance-based comparisons on synthetic or benchmark tasks) but prioritise expert reviews and user-centred comparative studies as primary indicators of success. This combination best reflects OrangeSpot’s non-functional requirements of perceived usefulness, usability and autonomy, while acknowledging that the field and underlying LLM technology continue to evolve rapidly.

4 ARTIFACT DESIGN & REFINEMENT

The goal of this chapter is to answer SQ5: What would a framework for guided process modelling look like? Where Chapter 2 identified the requirements for a process modelling assistant and Chapter 3 explored candidate generation methods, this chapter describes how these insights were translated into a working artifact. It outlines the basic technical architecture, the experimental test environment, and four successive design iterations (V1-V4). The final section synthesises these results into a framework for guided process modelling.

The remainder of this chapter is structured as follows:

- Section 4.1 describes experimental test environment, as well as the base architecture of the assistant and agent configurations
- Section 4.2 discusses four design iterations (V1-V4), each with its interaction design, agent configuration and pilot-test findings.
- Section 4.3 synthesises these results into a conceptual framework and provides an explicit answer to SQ5.

4.1 TEST ENVIRONMENT

To experiment safely with different assistant designs, the project used a dedicated test environment that sits completely outside the production OrangeSpot platform. A separate test environment was used primarily to support rapid iteration during development. It allowed changes to be made freely without the risk of breaking unrelated parts of the OrangeSpot platform, and without the risk that test users would accidentally gain access to other areas of the application. In addition, it made it possible to set up a lightweight, dedicated database and data model focused only on the elements required for this research, without having to integrate with parts of OrangeSpot that were not relevant to the evaluation. This was considered acceptable here because the core UI/UX needed for blueprint modelling was largely recreated in the test environment, enabling participants to perform comparable tasks while allowing faster development and safer experimentation. A screenshot of the test environment interface is included in Appendix J.

The Archimate models for V1-V4 all share the same basic technology stack: a browser-based frontend, a lightweight backend, and a separate AI orchestration component built with LangChain and FlowiseAI, all running in Docker containers.

The frontend is a visual modelling interface that deliberately mimics OrangeSpot's own editor. Users can open a blueprint, navigate through its layers and use familiar controls to create, update or delete steps and entities. On top of this, it shows the process modelling assistant through a chat interface. The frontend is the only interface between user and assistant. Each query, suggestion, acceptance or rejection is logged together with basic session information, so that later we can analyse how the assistant was used. Technically, the frontend is served from a Docker container on Cloudflare's edge platform.

The backend provides a thin but important layer between this visual editor and the underlying blueprint data. Its main responsibility is to expose a set of HTTP endpoints that the frontend and the assistant can safely call. One group of endpoints implements straightforward CRUD operations on steps and entities, which are used whenever the user models manually. A

second group of endpoints is reserved for the assistant: these accept proposed changes, validate them and apply them to the stored blueprint. The backend is itself a small, containerised service, deployed on Fly.io, and keeps simple logs of every request and response to support debugging and usage analysis.

The process modelling assistant runs in a separate Docker container using FlowiseAI, hosted on Railway. FlowiseAI is an open-source, drag-and-drop environment for building applications and agents powered by large language models (FlowiseAI, n.d.-b). It sits on top of the LangChain framework and exposes LangChain's building blocks as nodes in a visual canvas (FlowiseAI, n.d.-a; LangChain, n.d.). LangChain itself is an open-source framework for developing applications that use large language models, offering standard abstractions for models, prompts, tools, memory and multi-step workflows (LangChain, n.d.). In this project, Flowise is used to design and update the agent hierarchies of different iterations. In Flowise, one can assign individual prompts to agents, choose which model to use, assign conversation memory, or set reasoning capabilities. Keeping Flowise in its own container means that different versions of the process modelling assistant can be changed without touching either the frontend or the backend. It also means the final design of the process modelling assistant can be easily integrated into OrangeSpot's production environment.

4.1.1 Prompt template

To keep the behaviour of different agents consistent and maintainable, the project uses a generic prompt template that is instantiated per agent. The template is organised into the following sections:

- **Role:** describes the agent's "profession" in natural language (e.g. process design consultant, translator, reviewer) and its position in the multi-agent system.
- **Inputs:** enumerates the JSON fields or other structured inputs the agent receives, with one or more small examples to illustrate their typical content.
- **Output:** specifies the exact output contract (keys, types and structure) the agent must produce, again with brief examples. This ensures that downstream components can reliably parse the result.
- **Chain-of-thought:** gives high-level reasoning steps the agent should follow to transform inputs into outputs (for example: first interpret the user's intent, then scope the relevant part of the process graph, then formulate design instructions).
- **Constraints:** lists common failure modes that should be avoided (such as inventing non-existent process steps, modifying entities when only processes should be changed, or ignoring the provided IDs), as well as structural constraints on the output. These constraints were gradually refined based on issues observed during pilot tests.
- **Input variables:** binds the abstract template to concrete runtime variables such as the current question, the selected process fragment, the modelling purpose and the business description. This links the prompt directly to the information available in the orchestration layer.

This structured prompting approach was created using the prompt-engineering techniques identified in Chapter 3: role prompting, few-shot learning, and chain-of-thought. At the same time, it remained lightweight enough to be iterated rapidly between design versions. An example of the prompt template can be seen in Figure 6.

ROLE
You are an {{AGENT ROLE}} in a multi-agent system designed to aid users in modelling their business processes.

INPUTS
Your input consists of:
{{sets of JSON input structure examples and descriptions}}

OUTPUT
Your output consists of:
{{sets of JSON output structure examples and descriptions}}

CHAIN-OF-THOUGHT
You ought to use your input information, and derive information conform the output structure through the following thinking steps:
{{chain of thought description}}

CONSTRAINTS
Refrain from doing the following:
{{any common mistakes caught during testing}}
{{any explicit structure constraints}}

INPUT VARIABLES
{{sets of input variables received from either request headers or previous agents}}

Figure 6 - Prompt template

4.1.2 Agent settings

Flowise allows different configurations for each agent node. For this project, three settings are particularly relevant: the chosen large language model, the reasoning mode, and whether conversation history is provided to the agent.

Model choice.
This setting determines which underlying foundation model powers each agent and therefore influences response quality, latency and cost. Early experiments use the OpenAI GPT-4o model. After OpenAI's 4.1 model became available, later iterations use GPT-4.1 for most agents because of its improved reasoning performance (OpenAI, 2025a). The assistant was not upgraded to GPT-5, due to noticeably higher latency during development, far exceeding the project's requirements.

Reasoning mode.
This setting controls whether OpenAI's structured reasoning capability is used and at what intensity (low/medium/high). OpenAI's structured reasoning feature is enabled selectively and configured at different levels. In general, agents such as the Designer that must make some form of subjective interpretation are assigned medium reasoning, while more mechanical agents such as the Translator and Response formatter run with reasoning disabled to minimise latency and avoid unnecessary reasoning effort on straightforward tasks.

Conversation history.
This setting specifies whether an agent receives previous dialogue turns as additional context when generating its response. Only those agents that benefit from cross-turn context receive access to the conversation history. Agents such as the Manager and Response agents are configured with history enabled, so they can interpret follow-up questions and maintain continuity in the dialogue. For the Designer, Evaluator and Translator agents, history is disabled: they receive only the scoped inputs passed from the Manager. This reduces token usage and limits the risk of agents drifting away from the currently selected process fragment.

4.2 DESIGN APPROACH

This section discusses four successive design iterations of the process modelling assistant (V1-V4). Together, they show how the initial idea of an "AI assistant for process modelling" was refined into the final artifact. For each iteration, an implementation was deployed in the dedicated test environment described in Section 4.1 and evaluated through pilot sessions with expert users.

In addition to the textual descriptions, each iteration is documented visually in the appendices. For every version (V1-V4) there is an ArchiMate model that shows the high-level technical architecture (frontend, backend and AI orchestration) and a corresponding Flowise chat flow that specifies the multi-agent configuration.

For each version (V1-V4), pilot sessions were conducted with three expert participants with backgrounds in industrial engineering and IT, all familiar with process modelling. The relevant experience of these participants is given in Table 5:

Table 5 - Pilot participant experience

ID	Relevant experience
EP1	Director & managing consultant for an IT consultancy company. Approx. 10 years of experience in computer science, IT, and organisational change/digitisation projects.
EP2	Technical lead with a software engineering background and around 7 years of experience with OrangeSpot
EP3	Business & IT consultant at IT consultancy company, bridging processes and IT systems, and having completed a master's in business information technology at the University of Twente.

Participants completed a short modelling task while verbalising their reasoning; observations and log data (response times, token usage and suggestions) formed the main input for the next iteration. Across the iterations, two design dimensions change most visibly. First, the multi-agent system (MAS) architecture evolves from a single agent to a small collection of specialised agents with distinct roles, prompts and settings. Second, the interaction architecture shifts from a fully automatic assistant that directly manipulates the blueprint to a suggestion-based, human-in-the-loop assistant in which users explicitly decide which changes are applied.

Within these dimensions, each version explores specific questions:

- Does the current hierarchy of agents and their responsibilities work well?
- Does the interaction between the user and the assistant work well?
- How can we adjust LLM settings to balance response quality against latency and API cost?
- To what extent should we leave responsibilities to the assistant instead of the user?

4.2.1 V1: Single-agent baseline

Purpose and design focus

The first iteration of the process modelling assistant started at a minimal implementation. The aim was to see what a single large language model agent could accomplish when given full responsibility for interpreting user queries, reasoning about the blueprint and applying changes. This version was designed to answer a pragmatic question: how far can we get with the simplest possible architecture, and what are its limitations in the onboarding context? A high-level ArchiMate diagram and a simple Flowise chat flow for this single-agent setup are included in the appendices under Appendix B: Process assistant V1.

Agent configuration and interaction design

In V1, the assistant consisted of one Flowise agent configured with the OpenAI GPT-4o model, medium structured reasoning and full conversation history. The agent received the user's natural language query, a JSON representation of the relevant blueprint fragment and a generic prompt template that specified its role, expected inputs and outputs, high-level reasoning steps and constraints.

The interaction pattern was intentionally straightforward. Users interacted with the assistant via a chat panel embedded in the modelling interface and could describe changes in plain language (e.g. "Design the sub-process of Sales" or "Add a step after 'Check application' that validates documents"). The assistant responded with a textual explanation of what it intended to do and directly modified the blueprint via the backend API.

Pilot findings

Pilot sessions with V1 highlighted two main problems. First, the resulting suggestions were often generic. Even with a structured prompt and access to the blueprint, the agent tended to produce vague and high-level processes, especially when tasked with bigger modelling assignments. This reinforced the impression that a single "all-purpose" agent was not sufficient to handle nuanced design tasks. Second, participants expressed a lack of control. Because every change went through the assistant and was applied immediately, users could not easily make small adjustments to its output without having to go through another iteration of the chatbot interaction. This combination of generic outputs and limited user control made V1 unsuitable as an onboarding assistant on its own. However, it provided useful feedback for V2, for which a hierarchical agent structure and manual CRUD operations were then proposed.

4.2.2 V2: A hierarchical agent structure

Motivation and design focus

The second iteration of the process modelling assistant was shaped directly by the weaknesses observed in V1. The generic suggestions of the single "all-purpose" agent suggested that a more specialised functionality was needed. At the same time, the lack of manual user control over the blueprint made it difficult to work iteratively on the blueprint. V2 therefore explored a hierarchical multi-agent setup that mirrors the way consultancy-led onboarding projects are typically organised and follows multi-agent system practices introduced in Chapter 3. The idea behind V2 was to emulate this structure with a small team of LLM agents, each with a focused responsibility.

Alongside this architectural change, V2 introduced manual operations into the modelling environment. Where V1 only allowed the assistant to modify the blueprint, V2 now allows users

to create, update, or delete steps and entities directly. This allowed users to refine or correct AI-generated designs themselves, instead of having to go through another full chatbot interaction for every small adjustment. A high-level ArchiMate diagram and a Flowise chat flow of this hierarchical setup are included in the appendices under Appendix C: Process assistant V2.

Architecture and agent configuration

In V2, the assistant was restructured into a hierarchy of five agents, all based on the GPT-4.1 model. The choice for GPT-4.1 over GPT-4o was motivated by its improved reasoning performance and reliability for structured tasks, which was considered important for coordinating multiple agents and translating between business language and technical API calls (OpenAI, 2025a).

At the top of the hierarchy, a manager agent acted as the planner and coordinator. Its role consisted of four different tasks: it interpreted the user's query intent, scoped the relevant current blueprint data, determined which agent should act next and wrote concrete instructions for that agent. The Manager was configured with medium reasoning and access to the conversation history, so that it could maintain context across multiple turns and keep the overall interaction on track.

If the Manager decided that design work was needed, it handed off to a Designer agent. The Designer took the Manager's instructions together with the scoped blueprint fragment and produced a proposed design. In contrast to V1, these proposals went beyond simply listing changes: the Designer also provided motivations behind the design, explaining why particular steps were added, reordered or grouped. The Designer ran GPT-4.1 with low reasoning and without conversation history. This configuration was chosen to keep its focus on the current modelling task, while still allowing enough reasoning capacity to produce coherent, justified designs without excessive latency.

After the designer produced a proposal, a dedicated evaluator agent acted as a peer reviewer. Its job was to check whether the proposed design matched the original user intent as interpreted by the manager and whether it was consistent with the existing blueprint. The Evaluator therefore compared the designer's suggestions back against the user request and the scoped context, flagging cases where the proposal was too generic, incomplete or misaligned. Like the designer, the evaluator used GPT-4.1 with low reasoning and no conversation history, reflecting its relatively focused review role.

After every iteration of design and evaluation, the manager was instructed to decide whether to refine further, ask for more clarification, or pass control to the translator agent. The translator had a purely technical role: it translated the natural language design suggestions into actual HTTP requests to the backend API. This involved mapping described changes (such as "insert a validation step after 'Check application' and assign the Sales Support role") to concrete CRUD operations on steps and entities, with the correct identifiers and parameters. To keep this step as deterministic and predictable as possible, the Translator was configured with GPT-4.1 but without reasoning and no conversation history.

Finally, a response agent constructed a coherent answer back to the user. It received information from the manager about what had been decided and from the translator about which operations were executed. Based on this, it formulated a concise and understandable message that explained what had changed, why it had changed and, if relevant, what the next

steps were. The response agent did not require reasoning, but it did benefit from access to the conversation history to maintain a consistent tone and refer to earlier user inputs.

Overall, this architecture concentrated higher reasoning where interpretation and coordination were needed (Manager, Designer, Evaluator) and used lighter configurations where tasks were more mechanical (Translator, Response).

Interaction design

On the interaction side, V2 made a significant change compared to V1 by adding manual modelling capabilities to the interface. Users were no longer dependent on the assistant to manipulate the blueprint. Instead, the modelling environment now exposed the full set of CRUD operations that were previously only used internally by the assistant. Users could create, edit and delete process steps themselves, and they could attach steps as sub-steps to other steps to build and refine hierarchical sub-processes.

Beyond steps, V2 also allowed users to work directly with entities. Roles, applications and data objects could be created or selected and then attached to individual process steps. This meant that users could assign, for example, the “Sales Support” role or a specific CRM application to a step and indicate which data objects were used or produced there. This combination allowed users to choose between two modes of working. For larger or more open-ended tasks, they could rely on the assistant’s hierarchical design process to generate and implement a proposal. For small fixes or targeted adjustments, they could use the manual CRUD operations to modify steps and entities directly, without having to rephrase everything as a chat request.

Pilot findings

In practice, the hierarchical design of V2 surfaced several issues. Most notably, end-to-end response times grew dramatically. Each user request now triggered a chain of five GPT-4.1 calls, several with reasoning enabled. During pilot sessions, this resulted in waiting times of over two minutes for some interactions, which broke the feeling of a smooth conversation. Participants would prefer manual modelling simply because the assistant felt too slow.

Additionally, the looping control flow between the designer, evaluator, and manager agents also introduced a new type of failure in the form of thought loops. In some cases, the designer and evaluator repeatedly proposed small variations of the same idea, with the manager continuing to request refinements instead of moving on to the translator. Similar loops appeared on the conversational side, where the manager and response agents sometimes kept asking the user for more clarification rather than committing to a design. Several efforts were made to reduce these loops, including tightening and simplifying prompts, capping the number of allowed refinement cycles and lowering the reasoning intensity of the agents. While these interventions reduced the most extreme cases, they did not eliminate the problem. The system remained vulnerable to internal loops that produced long delays or no usable output.

Finally, although manual modelling was now available and widely appreciated by participants, AI-driven changes were still applied automatically once the agent chain finished. Participants reported that they still lacked a strong sense of transparency and recoverability when relying on the assistant. It was not always clear which exact changes in the blueprint were caused by a given assistant interaction, and there was no built-in mechanism to inspect or revert changes after they had been made.

Taken together, these findings suggested that V2, while closer to consultancy-led onboarding and MAS best practices, had become too complex. This motivated the design of V3, which simplified the agent structure and introduced an explicit suggestion mechanism to keep users in control of the blueprint.

4.2.3 V3: Human-In-The-Loop

Motivation and design focus

The third iteration of the process modelling assistant was a direct response to the problems encountered with V2. While the hierarchical multi-agent structure brought conceptual clarity, it also made the system heavy and slow in practice. Each user request triggered several GPT-4.1 calls, sometimes looping between the manager, designer and evaluator agents without converging on a result. Combined with the fact that AI-driven changes were still applied automatically to the blueprint, this led to long waiting times and a limited sense of transparency and recoverability for users.

V3 therefore pursued two main goals. First, it aimed to simplify the architecture into a linear agent setup, removing unnecessary internal loops and reducing response times. Second, it introduced an explicit suggestion mechanism to ensure that the assistant no longer modified the blueprint directly, but instead proposed changes that users could inspect and accept or reject. In doing so, the evaluation role of the V2 evaluator agent was effectively moved back to the user: instead of an internal agent judging the quality of designs, the user now became the primary decision-maker, using the accept/decline mechanism as an explicit evaluation step.

In addition, V3 extended the context available to the assistant by introducing a short business description and modelling purpose at the start of each session, so that suggestions would better reflect the organisation's onboarding context. A high-level ArchiMate diagram and a Flowise chat flow of this linear setup are included in the appendices under Appendix D: Process assistant V3

Architecture and agent configuration

Architecturally, V3 kept the same basic agent roles as V2 where possible (manager, designer, translator, response) but removed the evaluator and simplified the control flow into a linear chain. The manager agent still interpreted the user's intent and scoped the relevant blueprint data, but instead of coordinating an iterative loop between designer and evaluator, it now handed off the task to the designer once every query. Its configuration remained largely the same: GPT-4.1 with medium reasoning and access to conversation history, as it still needed to maintain context and steer the interaction.

The designer agent also retained its core function of producing concrete design proposals (with motivations), but its prompt was tightened to reflect the new suggestion-based interaction. Additionally, it now also includes domain context given by the user beforehand, which will be further discussed in the section "interaction design". The designer continued to run GPT-4.1 with low reasoning and no conversation history.

The evaluator agent was removed entirely in V3. Rather than inserting a separate peer-review step, the end-user was given more influence by a human-in-the-loop component. This reduced the number of GPT calls per interaction and eliminated one of the main sources of thought loops observed in V2, while maintaining quality and reducing latency.

The translator agent's role remained similar, but its target changed. Instead of translating the designer's output into immediate CRUD operations on the blueprint, the translator now

generated HTTP requests for a newly introduced suggestions API. The translator still used GPT-4.1 without reasoning and without conversation history.

Finally, the response agent continued to turn the internal results into a coherent answer for the user. Its configuration was unchanged: GPT-4.1 without reasoning but with access to the conversation history. Over time, this conversation history effectively captured the user's evaluation preferences, which the manager could use to steer future interactions.

Interaction design

In V3, the interaction with the assistant was reshaped around two key changes. First, before any modelling takes place, the user now provides a short business description and selects a modelling purpose (for example, onboarding, training or process improvement). This information is entered once at the start of the session and is then reused in subsequent interactions. It gives the assistant an explicit context for its proposals, so that suggestions are not only structurally correct but also aligned with why the blueprint is being modelled in the first place.

The second change is that the assistant no longer edits the blueprint directly. Instead, when the user asks for help (e.g. "Refine the sub-process for Sales" or "Add a validation step after 'Check application' and assign it to Sales Support"), the assistant returns a set of suggestions. These suggestions are shown in the modelling interface as proposed edits, indicating which steps or entities would be added, removed or updated. The user then explicitly accepts or rejects each suggestion. Only accepted suggestions are translated into actual CRUD operations on the blueprint; rejected ones are discarded and never affect the underlying model.

As a result, the evaluation step that was previously handled by the evaluator agent in V2 is effectively brought back to the user. The accept/decline mechanism gives users a clear, fast way to approve or reject the assistant's proposals, while the conversation history captures these decisions and any accompanying feedback. This allows the assistant to incrementally adapt its behaviour over the course of a session, while ensuring that nothing becomes part of the "real" blueprint without the user's explicit confirmation.

Pilot findings

Pilot sessions with V3 indicated that this linear, suggestion-based design solved several of the most pressing issues from V2. Response times improved noticeably, since each request now triggered fewer LLM calls and there were no longer nested loops between a designer and an evaluator. Participants experienced the assistant as more responsive and predictable, which made them more inclined to use it for everyday modelling tasks.

The introduction of suggestions and the shift of evaluation back to the user had a clear positive effect on perceived control and safety. Participants appreciated that the assistant could no longer silently change the blueprint. Instead, they could inspect each proposed change and decide whether to apply it. The accept/decline mechanism was experienced as a natural way work with the process modelling assistant, while staying in the driver's seat.

At the same time, the pilots showed that important challenges remained. Despite the manager's scoping and the additional context from the business description and modelling purpose, the assistant still struggled with scoping in some situations. It occasionally proposed changes to parts of the blueprint that the user did not intend to modify, such as a different process layer or a related sub-process. In addition, suggestions often mixed process-level and entity-level changes in a single proposal. While perhaps technically correct, this made it harder

for users to maintain a clear overview of what was being changed, especially when they were currently focused on either the process flow or the underlying roles and systems.

These scoping and transparency remaining issues led to the decision of moving even more of the assistant's current responsibilities back to the end-user for V4.

4.2.4 V4: Context selection

Motivation and design focus

The fourth iteration of the process modelling assistant was driven by the remaining issues identified in V3. Although the human-in-the-loop suggestion mechanism clearly improved control and transparency, the assistant still struggled with scoping. It sometimes proposed changes in parts of the blueprint that the user did not intend to touch, and it almost always mixed process-level and entity-level changes regardless of the user query. This made it harder for users to maintain a clear overview, especially when they wanted to focus on either the process flow or the underlying roles, applications and data objects.

To address this, V4 moved even more responsibility for context and scope back to the end-user. The central idea was that the assistant should only operate within a clearly defined context that the user selects explicitly, both in terms of where in the blueprint to work and what type of elements are in scope. V4 therefore introduced two key concepts: explicit context selection (choosing the relevant process layer before requesting suggestions) and separate modes for process and entity modelling, so that suggestions no longer mix the two by default).

At the same time, V4 retained the linear, suggestion-based architecture of V3 and the existing manual modelling capabilities. The aim was not to add more automation, but to make the existing automation more predictable and aligned with the user. A high-level ArchiMate diagram and Flowise chat flows of this dual-lane setup are included in the appendices under Appendix E: Process assistant V4.

Architecture and agent configuration

Architecturally, V4 builds directly on top of V3's linear multi-agent structure. It keeps the manager, designer, translator and response agents, but splits the agent pipeline into two parallel lanes: one for process modelling and one for entity modelling. Each lane follows the same basic sequence as V3, but with prompts and constraints tailored to its specific type of task.

In the process lane, the manager and designer agents are configured to work only on process-related elements: ordering and grouping steps, refining sub-processes and adjusting flows between activities. Their prompts explicitly refer to process layers and sub-process hierarchies, and they are instructed not to introduce or modify entities such as roles or applications. The translator in this lane generates suggestions that only affect process steps and their relationships.

In the entity lane, the manager and designer agents use prompts that focus exclusively on roles, applications and data objects. Here, the assistant is allowed to propose adding, updating or removing entities and attaching them to steps, but it treats the existing process flow as fixed context. The corresponding translator generates suggestions that only affect entities and their links to process steps.

The response agent remains shared between the two lanes. It still runs GPT-4.1 without reasoning but with access to the conversation history, and its job is to present the suggestions

from whichever lane is active in a coherent, user-friendly way. The manager and designer agents in both lanes continue to use GPT-4.1 with the same reasoning and history settings as in V3 (medium reasoning with history for managers, low reasoning without history for designers), so that behaviour remains consistent while the prompts and allowed operations differ.

V4 also explicitly considered, but ultimately rejected, switching to the newly released GPT-5 during development. While GPT-5 offered potential quality improvements (OpenAI, 2025b), experiments revealed significant higher latency and API costs. Retaining GPT-4.1 across all agents therefore allowed the design changes in V4 to focus on scope and division of responsibility rather than on model changes.

Interaction design

In V4, the interaction with the assistant is further improved by adding explicit context selection before each modelling request. As in V3, users still begin a session by entering a short business description and selecting a modelling purpose (e.g. onboarding, training or process improvement). This information remains available to the assistant throughout the session and continues to shape its suggestions.

On top of this, V4 asks the user to make two additional choices when requesting help from the assistant. First, the user selects the relevant process layer or fragment they want to work on. This can be the end-to-end process, a specific sub-process or a smaller segment within the blueprint. This selection is passed to the assistant and used to limit the scope of any suggestions to that part of the model.

Second, the user selects a modelling mode: either process or entity. In process mode, the assistant is only allowed to propose changes to steps and flows; entities are treated as fixed context. In entity mode, the assistant is only allowed to propose changes to roles, applications and data objects and their assignments to steps; the process structure itself remains untouched. This choice determines which lane, process or entity, is activated for the current request.

Manual modelling remains available in the same way as in V2 and V3. Users can still create, edit and delete steps, attach sub-steps and manage entities directly using the CRUD controls. The difference in V4 is that whenever they choose to involve the assistant, they do so with a clearly defined context and mode. This keeps the assistant's responsibilities narrow and well aligned with the user's current focus.

Pilot findings

Pilot sessions with V4 showed that the combination of context selection and parallel lanes largely solved the remaining issues from V3. Participants reported that suggestions were now much closer to what they had in mind. Because they first selected a process layer, suggestions stopped "jumping around" to other parts of the blueprint. And by choosing between process mode and entity mode, they no longer received mixed suggestions that changed both steps and entities at the same time.

The human-in-the-loop suggestion mechanism from V3 continued to work well. Users still reviewed each proposed change and decided whether to apply it, and they felt that this was easier now that suggestions were better scoped. The extra step of selecting context and mode was seen as a small effort, but worth it because it led to more relevant and predictable outcomes.

Overall, participants experienced V4 as more predictable and trustworthy than earlier versions. The assistant not only required explicit approval for every change but also asked users to define where and how it was allowed to operate. Based on these results, V4 was considered mature enough to be used as the basis for the evaluations in the following chapter.

4.3 Answer to SQ5

This chapter translated the requirements from Chapter 2 and the method insights from Chapter 3 into a working process modelling assistant for OrangeSpot. To enable rapid experimentation, a dedicated test environment was built with a frontend modelling interface, a backend exposing CRUD and assistant endpoints, and an AI orchestration layer in Flowise. Within this environment, the assistant's behaviour was controlled through a generic prompt template (role, inputs/outputs, reasoning steps and constraints) and consistent agent settings that balance quality against latency and cost via model choice, selective reasoning, and controlled use of conversation history.

Across four design iterations (V1-V4), the artifact evolved along two key dimensions: (1) from a single monolithic agent to specialised multi-agent configurations, and (2) from automatic blueprint changes to suggestion-based, human-in-the-loop interaction where users explicitly select context and approve changes.

- V1 demonstrated that a single agent could execute end-to-end modelling actions, but pilot sessions showed outputs were often generic and users lacked control because changes were applied immediately.
- V2 introduced a hierarchical agent structure and manual CRUD operations, better reflecting consultancy-led onboarding, yet pilots revealed unacceptable response times and infinite "thought loops," while transparency and recoverability were still limited because AI changes remained automatic.
- V3 addressed these issues by simplifying the architecture into a linear chain and introducing an explicit suggestion mechanism: the assistant no longer edits the blueprint directly but proposes changes that users can accept or reject. This significantly improved responsiveness and perceived control, but pilots showed scoping remained difficult: suggestions sometimes targeted unintended blueprint areas and frequently mixed process- and entity-level changes.
- V4 resolved these remaining issues by moving more scoping responsibility to the user through explicit context selection and by splitting assistance into separate process and entity modes, preventing mixed suggestions by default. Participants experienced V4 as more predictable and trustworthy, making it mature enough to serve as the basis for the treatment evaluation in the next chapter.

Taken together, these iterations provide an explicit answer to SQ5: a framework for guided process modelling should be built around shared responsibility between assistant and user. The assistant contributes speed, structure and candidate designs through a lightweight multi-agent pipeline, while the user retains authority through both explicit scoping decisions and an accept/decline functionality that turns evaluation into a natural part of the interaction. In this framework, guidance is not achieved by maximising automation, but by making the assistant's contribution transparent and controllable. A visual representation of the progression of the artifact design can be found in Figure 7.

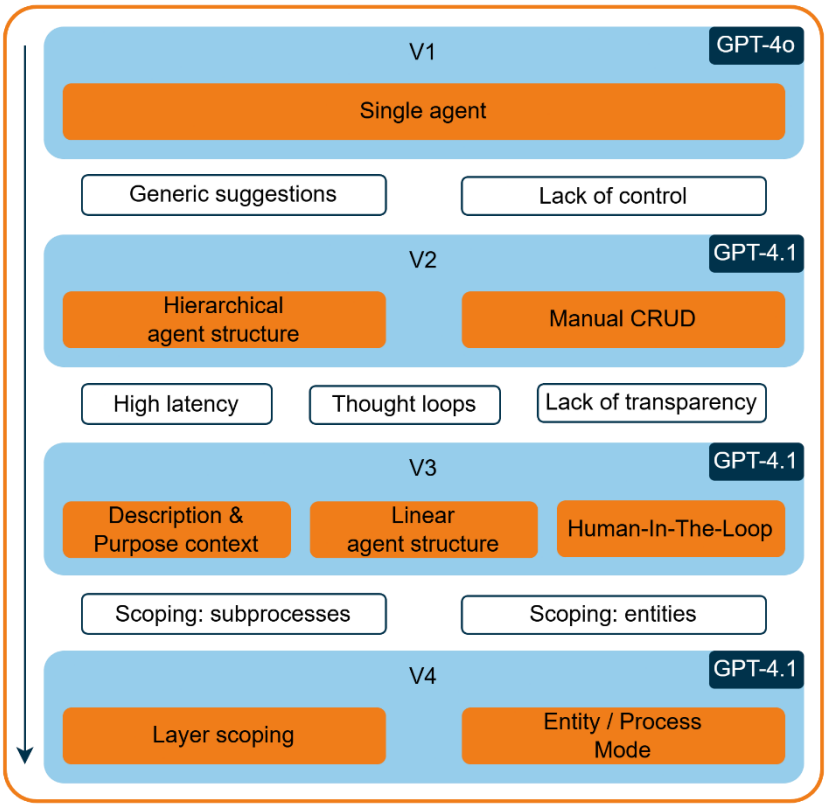


Figure 7 - Artifact design progression

5 EVALUATION

The purpose of this chapter is to evaluate V4 of the artifact by comparing AI-assisted process modelling to a manual baseline. In section 2.5.3 Requirements, it was established that the assistant aims to adhere to the following requirements:

- (N1) suggestions are delivered within ~30 seconds,
- (N2) a first usable onboarding model can be produced within ~30 minutes,
- (N3) users rate usefulness at ≥60%,
- (N4) produced models are comparable in accuracy to consultant-led onboarding, and
- (N5) users prefer the assistant over starting from a blank canvas.

Effectively, these requirements assign the following three objectives to this evaluation phase:

- (O1) Compare AI-assisted and manual modelling in terms of model quality.
- (O2) Compare AI-assisted and manual modelling in terms of perceived model quality.
- (O3) Evaluate the process modelling assistant in usability.

The first of these objectives will be measured by evaluation of experts using a model-quality rubric derived from the criteria identified in section 3.2.4: Requirements. The latter two of these objectives correspond with so-called user acceptance: if users do not think the assistant helps them or if it feels hard to use, they will not choose it in practice. To structure this part of the evaluation, the Unified Theory of Acceptance and Use of Technology (UTAUT) (Venkatesh et al., 2003) is used in this paper. According to the UTAUT framework, people's intention to use a system is shaped by four factors:

- *performance expectancy*, "the degree to which an individual believes that using the system will help him or her to attain gains in job performance".
- *effort expectancy*, "the degree of ease associated with the use of the system"
- social influence, "the degree to which an individual perceives that important others believe he or she should use the new system"
- *facilitating conditions*, "the degree to which an individual believes that an organisation's technical infrastructure exists to support the use of the system".

Together, these four factors determine the "behavioural intention" of the end-user. To which extent each factor contributes to the user's behavioural intention is determined by personal attributes of the user, namely gender, age, experience, and voluntariness of use. Age influences all four predictors. Gender influences effort and performance expectancy, and social influence. Experience influences effort expectancy, social influence, and facilitating conditions. Finally, voluntariness of use, whether a user chose to use the technology themselves, influences social influence and behavioural intention. A visual representation of this framework is given in Figure 8.

The remainder of this chapter is structured as follows. Section 5.1 details the evaluation design, describing the testing workflow, the participant sampling strategy, and the modelling scenario used to ensure comparability between the manual and AI-assisted conditions. Section 5.2 then specifies the measurement instruments used to operationalise the evaluation objectives, combining pre-task demographics, behavioural and performance logs, post-task questionnaires (aligned with UTAUT), and expert rubric-based assessments of the produced models. Section 5.3 reports the results across these instruments, covering demographic characteristics, behavioural metrics, expert-rated model quality, perceived outcomes, and

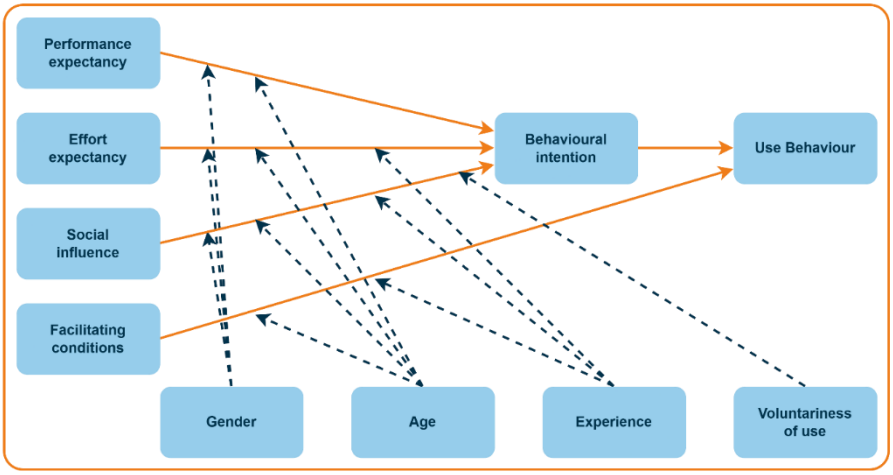


Figure 8 - The Unified Theory of Acceptance and Use of Technology (UTAUT)

correlational patterns between key variables. Finally, Section 5.4 synthesises the findings to answer SQ6 by linking the observed outcomes back to the assistant's non-functional requirements and the three evaluation objectives.

5.1 Evaluation Design

5.1.1 Workflow

During the evaluation, participants complete a modelling task under one of two conditions: manual modelling (without the assistant) or AI-assisted modelling (with V4). This separation matches the onboarding problem addressed in this thesis: starting from a blank canvas by yourself, opposed to having a process modelling assistant available. The testing workflow is summarised in Figure 9, which visualises the order of activities during the user tests. The overall flow is identical for both conditions, with the key difference being whether participants have access to the assistant during the modelling task and which version of the posterior questionnaire they receive.

The submission starts with a short tool explanation and case briefing. This step introduces the modelling interface and explains the assignment scenario. The intention is to reduce the time needed to get familiar with the assignment context and learning to work with the assistant. Next, participants complete the pre-task questionnaire, which documents background characteristics of the user demographic. This information is used to describe the sample and to make potential differences between the proxy group and the intended user population more transparent in the later analysis. Participants then perform the modelling task within the fixed 30-minute timebox. During this phase, the test environment captures behavioural and performance metrics, which are described in more detail in section 5.2 Measurement instruments. Immediately after the modelling task, participants complete a posterior questionnaire. This questionnaire focuses on acceptance criteria inspired by the UTAUT, and in the AI-assisted group it additionally includes items that evaluate the quality of the assistant's guidance. Finally, the submitted models and associated logs are stored for analysis and for subsequent expert evaluation using a model-quality rubric derived from the literature review section 3.2.4.

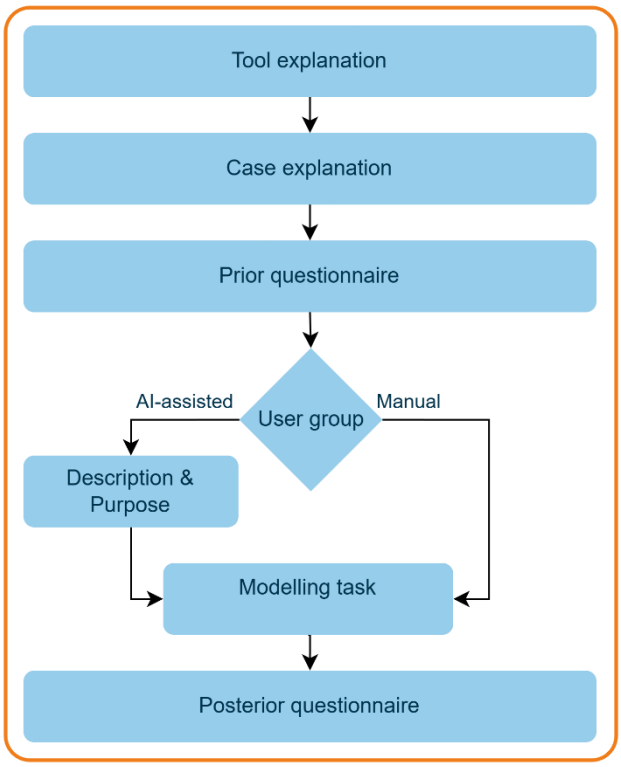


Figure 9 - Visual representation of the evaluation workflow

5.1.2 Participants and sampling

The assistant is primarily intended for client-side end-users of OrangeSpot who are responsible for capturing and structuring business processes, such as project managers, IT managers, or sustainability coordinators. In Chapter 2.5.1 these users are characterised as “normal operators”: they typically understand their own operations well but often lack process-modelling expertise and therefore struggle most with the “blank canvas” problem.

Because recruiting a sufficiently large set of real OrangeSpot clients for a controlled experiment is difficult, this evaluation uses business students as an end-user proxy group. The rationale is that this group typically has general business-operations knowledge, while not necessarily having strong modelling experience, mirroring the onboarding situation in which users can explain how work is done but struggle to structure it into a coherent process model.

At the same time, this exposes a big limitation: students may differ from the real user population in attributes such as age and domain-specific operational knowledge, which are key influencing factors according to the UTAUT framework. To make this limitation more transparent and interpretable, the study documents participant background characteristics through the pre-task questionnaire, capturing demographics and experience variables that UTAUT highlights as influential.

In addition to the end-user proxy group, the evaluation includes an expert perspective: model outputs are assessed by OrangeSpot consultants with extensive experience in process modelling using the OrangeSpot modelling language. Through this source, created process models can be evaluated on effective usability or quality.

5.1.3 Modelling scenario

The modelling task is based on a mid-sized physical supermarket scenario. This case is chosen because it is assumed to be broadly familiar to most participants, while still being complex enough to contain multiple roles, operational systems, and information handoffs. By giving all test users the same scenario, process model quality can be compared more easily.

To make sure the focus of the submissions is similar enough to reliably compare them while staying within the 30-minute timeframe, the scenario description provided to participants focuses on two specific sub-processes:

- Employee management, containing tasks related to onboarding new employees, shift planning and scheduling, or managing salaries and payroll.
- Inventory management, containing tasks related to purchasing and managing inventory.

The scenario simulates a context in which management is hiring a new managing employee, which needs to be made aware of how software applications are used throughout daily operations and processes. This steers the test users into thinking about responsibilities, applications, and information handoffs. The exact scenario description handed to users is given in Figure 10.

5.2 Measurement instruments

This section describes the instruments used to operationalise the evaluation objectives O1-O3 and non-functional requirements introduced at the beginning of this. They provide indicators of how well the assistant supports process modelling in the onboarding scenario.

Four categories of instruments are used:

- A pre-task questionnaire capturing participant characteristics and UTAUT moderators.
- Behavioural and performance metrics logged by the test environment.
- Posterior questionnaires completed after the task in the manual and AI-assisted conditions.
- An expert model-quality rubric applied to all submitted models.

The following sub-sections will describe these measurement instruments in more detail:

The Scenario:
You are a manager at Alhert Beijn, a supermarket chain in The Netherlands. For a single store, you are about to onboard a new managing employee.

To onboard the new manager, your task is to create documentation of the store's processes and how software applications are used throughout daily operations. Create a clear process documentation that will help this new manager understand how the supermarket operates.

Scope - Focus on These Two Areas:

1. Employee Management: Document processes related to onboarding new employees, shift planning and scheduling, and managing salaries and payroll.
2. Inventory Management: Document how the store tracks inventory levels, orders new stock from suppliers, and manages product receiving and storage.

Requirements:

- Include multiple roles (e.g. manager, cashier)Include multiple applications (e.g. point-of-sale system, accounting application)Where relevant, include data objects (e.g. payslip, inventory log)Use a maximum of three levels of subprocesses
- Time Limit: You have 30 minutes to complete and submit your process documentation.

Figure 10 - User test scenario description

5.2.1 Pre-task questionnaire

Before starting the modelling task, all participants complete a pre-task questionnaire. The instrument records basic characteristics of the user demographic: age, gender, academic level, academic direction (field of study), process modelling experience, AI experience, and experience with supermarkets. These variables do not measure the non-functional requirements directly, but they are important for describing and comparing the samples in the two conditions and for understanding how participant characteristics may moderate the outcomes.

Age and gender are included because they are central moderators in the UTAUT framework and may influence performance expectancy, effort expectancy and behavioural intention. Differences in usability or acceptance (O3) might, for instance, partly reflect age or gender composition rather than properties of the assistant itself, so these variables are needed to interpret the results. Academic level and field of study serve as proxies for analytical and modelling skills. Participants with a higher academic level or a business/IT background may find it easier to structure processes than participants with a lower academic level or completely different background.

Process modelling experience and AI experience capture more specific forms of expertise. Prior modelling experience is expected to influence both how good a model participants can produce (O1, N4) and how easy they perceive the modelling task to be, with or without the assistant (O3, N5). Prior AI experience may shape how quickly participants understand the assistant's behaviour, how confidently they interact with it, and how much benefit they can extract from it, which is particularly relevant when interpreting usability and perceived usefulness in the AI condition (O3, N3, N5). Finally, supermarket experience reflects familiarity with the case domain. Participants who have worked in or with supermarkets are likely to have a more accurate mental model of the underlying process, which can positively affect perceived and expert-rated correctness and completeness (O1, O2, N4) independently of the assistant.

Overall, the pre-task questionnaire provides the contextual information necessary to assess whether observed differences between manual and AI-assisted modelling can plausibly be attributed to the assistant rather than to systematic differences in the participant demographic. In Table 6, the different options for the pre-task questionnaire can be seen.

Table 6 - pre-task questionnaire

Topic	Response options
Age	Participant age
Gender	Male; Female; Other
Educational level	MBO; HBO; WO Bachelor; WO Master
Educational background	Field of study
Process Modelling Experience	Very limited; Academic setting; Professional experience
AI Experience	Very limited; Casually; For process modelling specifically
Supermarket Experience	Customer; Employee; Management; Consulting

5.2.2 Behavioural metrics

During the 30-minute timebox, the test environment automatically logs several behavioural and performance metrics. For all participants, the environment records the total time taken from the start of the task to submission of the final model. In the AI-assisted condition, it additionally records the average response time of the assistant, the number of assistant queries, and the total number of tokens used by the underlying language model.

Total time taken operationalises how quickly participants can move from a blank canvas to a usable onboarding model. It is directly linked to N2, which specifies that the assistant should enable a first model within roughly 30 minutes, and it is central for O1 and O3: any claim that AI support improves or at least does not harm efficiency must be assessed against this objective measure of task duration. Average response time reflects how long participants wait for the assistant after submitting a prompt. This metric is the concrete operationalisation of N1, which requires that suggestions be returned within an acceptable time to maintain a feeling of interactive support. It also provides an objective reference point for participants' subjective judgments of efficiency and fluency in the AI posterior questionnaire.

The number of assistant queries captures how intensively the assistant is used. If participants in the AI condition rarely call the assistant, then any benefits in model quality or time cannot be attributed to sustained AI support, and low usage would moreover suggest that the assistant is not perceived as sufficiently helpful or easy to use. In contrast, high usage would support the idea that the assistant is integrated into the modelling process and therefore relevant for O1, O3 and N5. Finally, total token usage is a proxy for computational cost. Although it is not an explicit user-facing requirement, it matters for the viability of deploying the assistant at scale: similar or better model quality (O1, N4) achieved with reasonable token consumption would support the broader goal of a cost-effective alternative to consultant-led onboarding.

In combination, these behavioural and performance metrics provide objective evidence for N1 and N2 and contextualise the data from the questionnaires, allowing a more nuanced assessment. In Table 7, these metrics are captured in table format.

Table 7 - Behavioural metrics during user-tests

Metric	Description
Total time taken	The difference in timestamps after entering pre-task questionnaire and before posterior questionnaire
Average response time	The difference in timestamps between sending a user query and receiving the response from the assistant
Total assistant queries	Amount of times the assistant has been queries (minus errors)
Total token use (cost)	The amount of OpenAI API tokens used across all user queries in a submission

5.2.3 Posterior questionnaire: manual user group

After completing the task, participants in the manual condition complete a posterior questionnaire. The instrument uses seven-point Likert items expressed as pairs of opposing statements. Its core constructs are perceived model accuracy, perceived model completeness, perceived efficiency, ease of modelling, and confidence in one's ability to model. The questionnaire concludes with an open-ended comments field.

Perceived accuracy and completeness capture how participants judge their own model in relation to the scenario. Accuracy items ask whether the model misrepresents or accurately describes the real process, while completeness items ask whether important steps, roles, applications and data objects are missing or adequately covered. Together, these items provide a self-assessment of model quality that can be compared with expert ratings and with responses in the AI-assisted condition. They are therefore directly relevant for O2 and indirectly for N4, as they reveal whether participants themselves perceive their models as accurate and complete enough for onboarding purposes.

Perceived efficiency addresses how easy the manual modelling felt to participants, compared to the time taken. Items contrast the experience of the task taking longer than it should with the feeling of being able to create a model quickly. These perceptions complement the log data on total time and are especially relevant for O3 and N2: if manual modelling is widely perceived as slow or effortful, this sets a clear target for the assistant to improve upon.

Ease of use focusses on how difficult it was to get started and continue modelling without extra help. They operationalise effort expectancy in the manual condition and contribute to O3 and N5 by providing a baseline for how demanding the task is without AI support. Confidence items ask participants to reflect on whether they felt they had sufficient knowledge and background to create a good model without AI. This captures self-efficacy and helps interpret both subjective and expert-rated model quality (O1, O2, N4): low confidence combined with low quality suggests a genuine skills gap, while low confidence combined with high quality may indicate that participants underestimate their own capabilities.

The open-ended comments field allows participants to describe, in their own words, what they found easy or difficult, whether any parts of the task or scenario were confusing, and what they would change. These qualitative remarks help to interpret the quantitative ratings, and reveal issues not anticipated in the closed questions. In Table 8, these statements are captured in table format.

Table 8 - Posterior questionnaire: manual modelling group

#	Topic	Counter-statement (1)		Statement (7)
1	Accuracy	"My model misrepresents how the real process works."	OOOOOOO	"The model I made describes the real process accurately."
2	Completeness	"Important steps, roles, applications, and data objects are missing from my model."	OOOOOOO	"My model covers all important steps, roles, applications, and data objects."
3	Efficiency	"Creating the model took longer than it should have."	OOOOOOO	"I was able to create the model quickly."
4	Ease of use	"Getting started and continuing modeling was difficult without extra help."	OOOOOOO	"It was easy to get started and keep modeling without extra help."
5	Confidence to model	"I lacked the knowledge/background to create a good model without AI."	OOOOOOO	"I had enough knowledge/background to create a good model without AI."

5.2.4 Posterior questionnaire: AI-assisted user group

Participants in the AI-assisted condition complete a posterior questionnaire that mostly mirrors the manual instrument and adds items that are specific to the presence of the assistant. As in the manual condition, seven-point Likert items are used, phrased as opposing statements.

The core constructs shared with the manual questionnaire: perceived accuracy, completeness, efficiency, ease of use, confidence, and an open-ended comments field, are adapted to the AI context. Accuracy and completeness items now refer explicitly to the model created "with the AI", thereby indicating whether participants regard the AI-assisted model as a faithful and sufficiently complete representation of the process. These items feed directly into O2 and support evaluation of N4 from the end-user perspective. Perceived efficiency asks whether using the AI saved time or made the task slower than manual modelling would have, linking directly to N2 and O3. Ease-of-use items focus on learning and operating the AI tool, rather than on modelling per se, and thus provide an assessment of effort expectancy for the AI interaction, which is central to O3 and N5. Confidence is framed as confidence in using the assistant effectively to create a good model, which indicates whether the tool is accessible to users with different levels of experience and background.

Beyond the shared constructs, the AI questionnaire contains items that specifically target the quality of AI interaction and social and behavioural aspects of AI use. First, clarity and explainability items ask whether AI suggestions were clear and easy to follow and whether the assistant explained its suggestions in a way participants could understand. These items connect directly to perceived transparency and interpretability of the tool. High clarity and explainability support N3 and N5, because users are more likely to see the assistant as useful and to trust its output if they can follow its reasoning. They are also central for O3, as a lack of clarity can make the tool feel difficult or even risky to use.

Second, items on relevance and guidance quality probe whether AI suggestions were aligned with the task and whether the assistant guided participants well through creating the model. These items reflect the performance-expectancy dimension of UTAUT in the AI context: they indicate whether the assistant contributes substantive value by proposing fitting process steps, roles and applications, and by structuring the modelling conversation in a way that resembles human consultant guidance. Strong ratings on relevance and guidance suggest that the assistant supports participants in reaching a usable model within the given time, which speaks to O1, O3, N2 and N3.

Third, a social-influence item asks participants whether people who matter in their environment would disapprove of or support the use of this AI tool. While social approval is not an explicit non-functional requirement, it considers the social influence factor of the UTAUT acceptance model. Even a technically strong tool may see limited uptake if users believe that colleagues or supervisors disapprove of AI use in this context.

Finally, a behavioural-intention item asks whether, for similar tasks in the future, participants would avoid using the tool or would use and recommend it. This item summarises their overall evaluation into a forward-looking indicator and is the most direct attitudinal operationalisation of N5. High intentions to reuse or recommend the assistant suggest that it meets its usability and usefulness targets and that it is likely to be preferred over manual modelling, thereby supporting O3. In Table 9, these statements are captured in table format.

Table 9 - Posterior questionnaire: AI-assisted group

#	Topic	Counter-statement (1)		Statement (7)
1	Accuracy	"The AI-led model misrepresents how the real process works."	OOOOOOO	"The model I made with the AI describes the real process accurately."
2	Completeness	"Important steps, roles, applications, and data objects are missing from my model."	OOOOOOO	"My model covers all important steps, roles, applications, and data objects."
3	Efficiency	"Using the AI took more time than modelling manually."	OOOOOOO	"Using the AI saved me time compared with doing it manually."
4	Ease of use	"Learning and using the AI tool was difficult."	OOOOOOO	"It was easy to learn and use the AI tool."
5	Social approval	"People who matter in my work/study would disapprove of me using this AI tool."	OOOOOOO	"People who matter in my work/study would support me using this AI tool."
6	Confidence to use	"I lacked the knowledge/background to use the AI tool effectively."	OOOOOOO	"I had enough knowledge/background to use the AI tool effectively."
7	Clarity of output	"The AI's suggestions were confusing and hard to follow."	OOOOOOO	"The AI's suggestions were clear and easy to follow."
8	Relevance	"The AI's suggestions were irrelevant to my task."	OOOOOOO	"The AI's suggestions matched the task I needed to do."
9	Explainability	"The AI did not explain its suggestions in a way I could understand."	OOOOOOO	"The AI explained why it suggested things in a way I could understand."
10	Future use	"For the same task I would avoid using this tool again or not recommend it to others."	OOOOOOO	"For the same task, I would use AI again or recommend it to others."
11	AI guidance	"The AI has not guided me well through creating the process model."	OOOOOOO	"The AI has guided me well through creating the process model."

5.2.5 Expert evaluations

All submitted process models are evaluated by an expert panel. The panel consists of four OrangeSpot consultants with extensive experience in process modelling using the OrangeSpot blueprint language. This expert perspective is essential for O1, which concerns model quality, and it complements the behavioural metrics and posterior questionnaires by providing an independent judgement of how usable the models would be in practice. The panel applies a structured rubric derived from the qualitative evaluation criteria identified in section 3.2.4 Evaluating process models. The rubric covers five dimensions: completeness, correctness, specificity, clarity and usability. Each perspective is scored on a seven-point Likert scale with a counter statement at 1 and a positive statement at 7.

A crucial design choice is that completeness is treated differently from the other four dimensions. Completeness is used primarily as an indicator of how far a participant progressed through the assignment within the 30-minute timebox, whereas correctness, specificity, clarity and usability are intended to assess the quality of the part that has actually been modelled,

regardless of how complete it is. For these topics, panel members are instructed to “zoom in” on the modelled part and judge its quality as independently as possible from how far the participant got.

Concretely, correctness captures whether the modelled fragment reflects a realistic version of the intended process: a low score indicates that many elements in the process model are not plausibly part of the supermarket scenario, while a high score indicates that the modelled part accurately reflects how such a process would work in real life. Specificity assesses whether the activities in the modelled part are sufficiently detailed and actionable to serve the scenario’s goal. Generic step labels receive low scores, while concrete, operational descriptions receive high scores. Clarity concerns how easy the modelled part is to follow, including the readability of the layout and the unambiguity of the terminology. A confusing, hard-to-read structure scores low, whereas a logically structured, clearly worded process model scores high. Finally, usability asks to what extent the modelled part could already contribute to the scenario’s goal of onboarding a new manager. At the low end, the model would have little practical use without significant further guidance or improvement. At the high end, it is considered ready for use in a real-world scenario. Essentially, this topic covers the “overall quality” of the submission. In Table 10, the statements are captured in table format.

For each submission, all four experts reach consensus on all five dimensions of a process model using this rubric. The panel scores provide a robust expert-based view of model quality. Because the expert panel is unaware which models are created with the assistant and which are not, results are accurately comparable between the AI-assisted and manual conditions.

Table 10 - Expert evaluation rubric

#	Topic	Counter-statement (1)		Statement (7)
1	Completeness	The model is missing many important elements and gaps limit its usefulness	OOOOOOO	The model includes all necessary elements
2	Correctness	The model does not reflect real-life scenario	OOOOOOO	The model accurately reflects a real-life scenario
3	Specificity	Steps are vague or too general, lack sufficient detail to serve the goal	OOOOOOO	Steps are specific and actionable, with enough detail to serve the goal
4	Clarity	The model is hard to follow, terminology is ambiguous and structure confusing	OOOOOOO	The model is easy to follow, terminology and structure are clear and logical
5	Useability	The model would have little use in practice without significant further guidance	OOOOOOO	The model is ready for practical use without further guidance

5.2.6 Summary

This section documented a strategy to evaluate the process modelling assistant against the non-functional requirements. It combines

- a pre-task questionnaire to document participant characteristics and potential UTAUT moderators
- behavioural/performance logs from the test environment (e.g., total task duration, assistant response time, usage intensity, token cost)

- posterior questionnaires capturing perceived model quality and user acceptance in both conditions
- an expert rubric in which OrangeSpot consultants score model outputs on completeness, correctness, specificity, clarity, and usability.

Together, these instruments ensure the evaluation can compare manual vs AI-assisted modelling from both an objective and perceived perspective, while also capturing the usability of the assistant itself. With these measurement instruments, it can be determined whether the process modelling assistant fulfills the stated requirements.

5.3 Results

This section reports the empirical outcomes of the evaluation, comparing a manual modelling condition (n = 9) to an AI-assisted modelling condition (n = 13). Participants were assigned to user groups alternately, and users without any model result were discarded. Results are presented primarily to characterise patterns within and between groups, and to provide context for the requirement-based conclusions presented in section 5.4.

5.3.1 Demographic characteristics

Before comparing conditions, it is important to understand who participated in each group and whether the two samples are comparable. The descriptive statistics below are based on the pre-task questionnaire as in Table 6 and are visualised in Figures 11 - 14.

Age distributions differ visibly between groups. Figure 11 shows the manual group is comparatively homogeneous, whereas the AI-assisted group shows a broader spread, including several higher-age observations. The wider age span in the AI-assisted group might be relevant when interpreting later variability in outcomes.

Process modelling experience also differs in composition. Figure 12 shows manual participants primarily report experience in an academic setting (66.7%), whereas the AI-assisted group is more mixed: a larger proportion reports very limited experience (38.5%) and a higher share report professional modelling experience (23.1%). This mix implies that the AI-assisted group includes both relatively inexperienced and relatively advanced modellers, which might contribute to a wider spread in performance and perceptions.

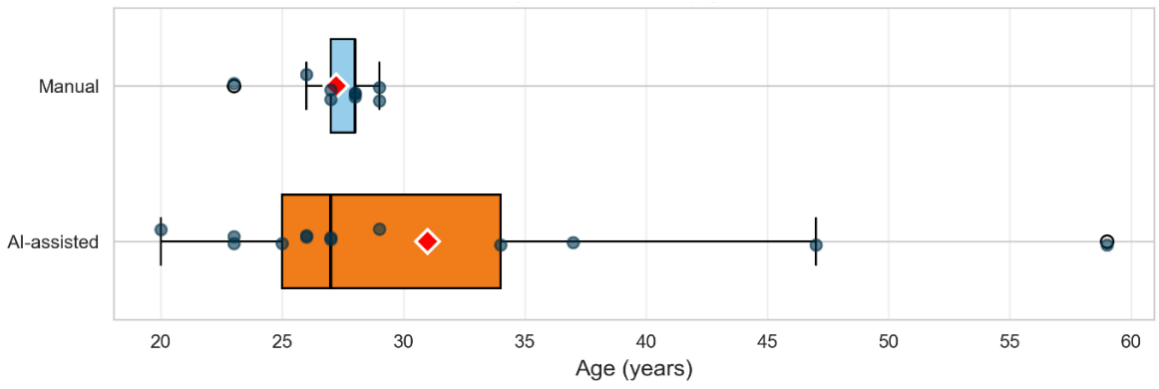


Figure 11 - Age distribution

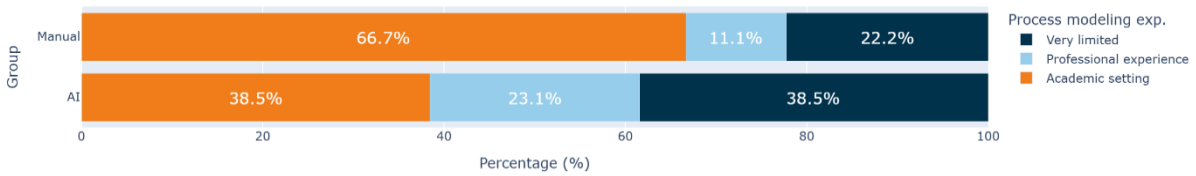


Figure 12 - Process modelling experience distribution

Across both conditions, the sample is fully male (13/13 AI-assisted, 9/9 Manual), and self-reported AI experience is fully "Casual AI experience" in the AI-assisted group. Figure 13 shows participants are predominantly WO Master students, with the AI-assisted group slightly more concentrated at this level (84.6% vs 66.7% in the Manual group). This suggests broadly similar educational backgrounds, but with a somewhat more advanced profile in the AI-assisted condition.

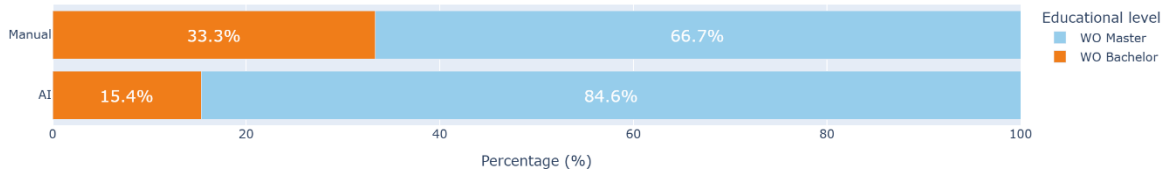


Figure 13 - Educational level distribution

Domain familiarity (supermarket experience) is mostly customer-only in both groups, but employee experience is more common in the AI-assisted condition as shown in Figure 14. This could influence how easily participants recognise the scenario and identify realistic process steps.

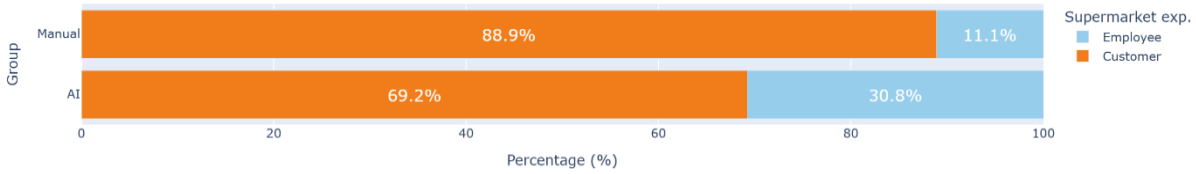


Figure 14 - Supermarket experience distribution

Overall, the Manual group is more homogeneous in age and modelling background, whereas the AI-assisted group is more heterogeneous on several baseline variables (age, modelling experience, supermarket experience). This heterogeneity is important context for interpreting later results: differences in variability between conditions may partly reflect who is in each group and not only the effect of the assistant itself.

5.3.2 Behavioural metrics

The system logs provide objective indicators of how the two workflows were experienced in practice. These metrics relate directly to N1 (response times) and N2 (total duration) and help contextualise posterior questionnaire results and expert ratings.

Task duration

During the fixed 30-minute timebox, the environment logged the time from starting the task to final submission. Participants were free to submit earlier; the 30 minutes function as an upper bound. Table 11 reports the statistics, which are visualized in Figure 15.

Table 11 - Task duration by group

Group	n	Median (min)	Mean (min)	IQR (min)	Range (min-max, min)	Capped (≥30 min)	p-value
AI-assisted	13	24.00	22.77	20.02-30.08	7.60-30.33	5/13	0.640
Manual	9	26.20	25.55	24.92-26.68	20.33-29.38	0/9	0.640

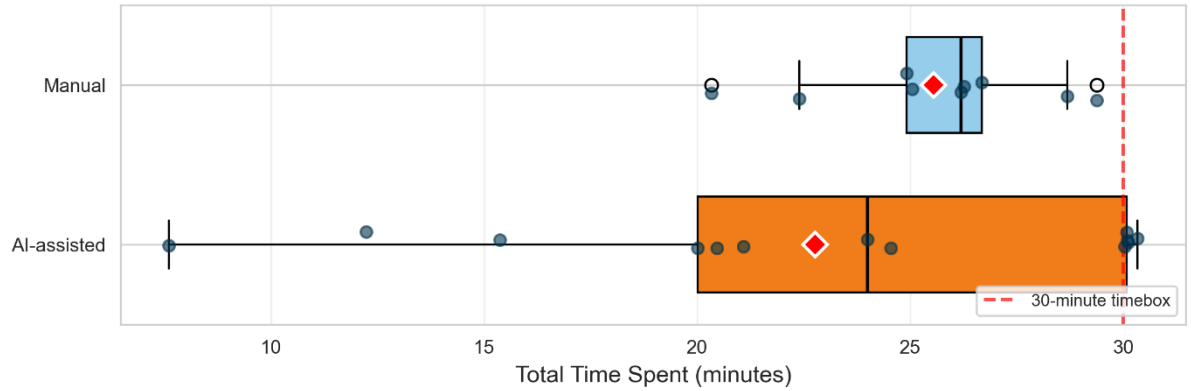


Figure 15 - Task duration by group

On average, manual participants spend somewhat more time in the modelling environment than AI-assisted participants (both in mean and median), but the difference is modest ($p = 0.640$) and clearly within the same order of magnitude. The more striking difference lies in the variability: manual durations are tightly clustered, whereas the AI-assisted group has some significantly shorter submission times. One interpretation is that some participants, especially in the AI-assisted condition, felt they had reached a "good enough" model earlier and therefore submitted sooner. However, the exploratory correlation analysis in Section 5.3.5 shows that spending more time is associated with higher expert-rated completeness ($\rho = 0.433$, $p = 0.044$), while perceived completeness trends in the opposite direction. This suggests that at least some early submissions may reflect premature satisfaction or underestimation of the intended scope, rather than faster completion. Supermarket experience does not show meaningful associations with expert-rated quality in Section 5.3.5, so domain familiarity alone is unlikely to explain the short durations. A possible mechanism is that AI assistance reduces critical self-checking (automation bias/cognitive offloading). This theme is revisited in Section 6.4.

AI usage

To understand how intensively the tool was used, Table 12 and Figure 16 visualize both the number of assistant calls and the total number of tokens consumed per participant.

Table 12 - Usage intensity

Metric	n	Median	Mean	IQR	Range (min-max)
Total AI Calls	13	9.0	9.9	5.0-15.0	4.0-17.0
Total Tokens	13	65442.0	72621.2	39813.0-96914.0	22225.0-152055.0

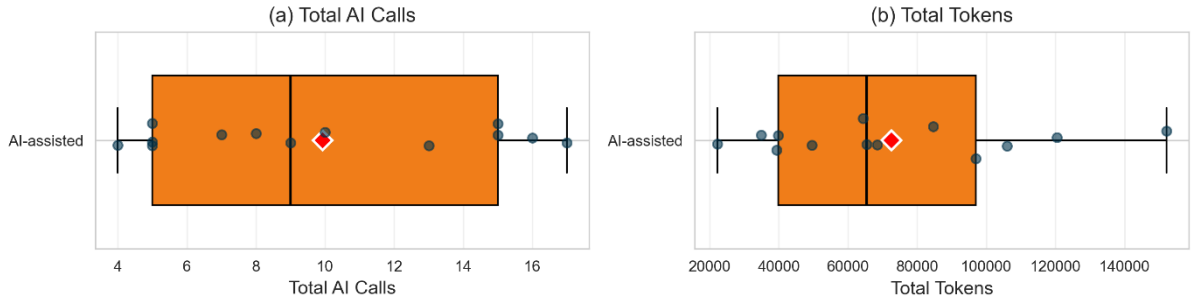


Figure 16 - Usage intensity

Usage is also reported as quite variable between submissions: some participants interact only a handful of times, others engage in extended back-and-forth. This potentially means that any effect on model quality or perceived experience can be attributed to intensity of AI usage.

From a cost perspective, these token volumes correspond to small absolute API costs. OpenAI's pricing documentation lists rates on the order of $2 per 1,000,000 input tokens, and $8 per 1,000,000 output tokens for ChatGPT 4.1. The assistant uses roughly as much input as output tokens, so using $5 / 1M as a rough conversion, the observed token usage corresponds to approximately $0.11-$0.76 per participant (median ≈ $0.33). (OpenAI, n.d.)

Assistant responsiveness

The assistant's responsiveness is operationalised as the average response time per participant, i.e. the time between submitting a query and receiving the assistant's answer, and Figure 17 shows the average response times between participants.

On average, the system slightly misses the strict N1 target of 30 seconds for every user. From a user perspective, this level of latency does not appear to have strongly hindered perceived efficiency, given the scores in section 5.3.4. Participants in the AI-assisted condition rate efficiency and ease of use substantially higher than the manual group (see Section 5.3.4). This suggests that, in the studied task, response times in the ~30-45s range were acceptable, but further optimisation is still desirable to reliably meet the N1 target in a production setting (see Section 6.5).No correlation was found between the total AI calls and average response time, indicating that the increasing context window was not responsible for an increase in response times.

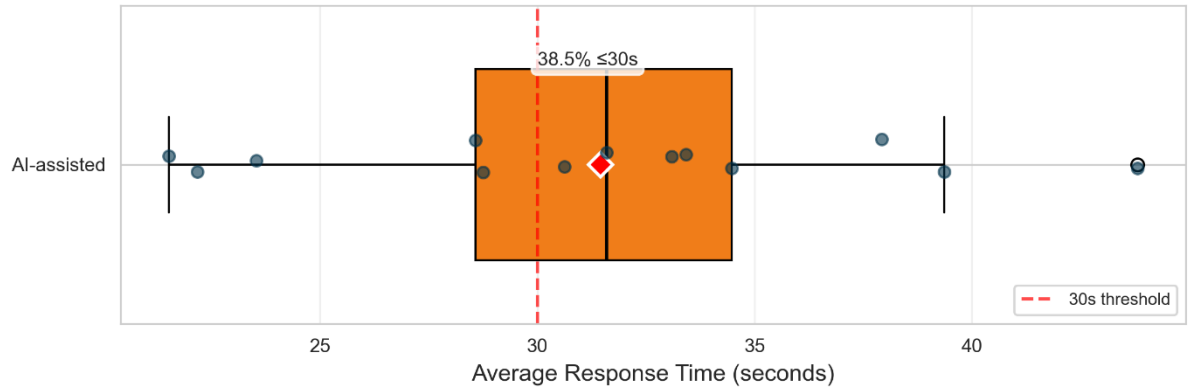


Figure 17 - Assistant reponse time

5.3.3 Model quality (expert rubric)

As described in Section 5.2.5, model quality was assessed using an expert rubric applied by OrangeSpot consultants across five dimensions (completeness, correctness, specificity, clarity, usability). The results below therefore reflect expert judgments of the submitted models, independent of participants' self-assessments. Table 13 reports group summaries (median, IQR, range) for the expert rubric dimensions. Across dimensions, central tendencies overlap substantially between conditions, but the AI-assisted group shows consistently higher dispersion (wider ranges and/or IQRs), especially visible in the low-end tails on several dimensions. Figure 18 and Figure 19 show these statistics visually to further emphasize this conclusion.

Table 13 - Expert evaluations

Dimension	Group	n	Median	Mean	IQR	Range	p-value
Completeness	AI-assisted	13	4.0	4.0	2.0-6.0	1.0-7.0	0.946
Completeness	Manual	9	5.0	4.1	3.0-5.0	2.0-6.0	0.946
Correctness	AI-assisted	13	5.0	4.3	4.0-6.0	1.0-6.0	0.862
Correctness	Manual	9	5.0	4.3	4.0-5.0	1.0-6.0	0.862
Specificity	AI-assisted	13	5.0	4.3	3.0-6.0	1.0-7.0	0.973
Specificity	Manual	9	4.0	4.4	3.0-5.0	3.0-7.0	0.973
Clarity	AI-assisted	13	4.0	3.9	3.0-6.0	1.0-6.0	0.409
Clarity	Manual	9	5.0	4.6	4.0-6.0	2.0-6.0	0.409
Usability	AI-assisted	13	4.0	3.5	1.0-5.0	1.0-6.0	0.610
Usability	Manual	9	4.0	4.0	3.0-5.0	2.0-6.0	0.610

Completeness

Manual models are rated similarly for completion on average (median 5.0 vs 4.0 for AI-assisted). More importantly, AI-assisted completeness is far more variable (range 1-7 vs 2-6), suggesting that AI assistance did not lead to a uniform improvement in progress. Instead, outcomes appear heterogeneous: some AI-assisted participants produced highly complete models, while others produced very incomplete ones.

Correctness
Both groups share the same median (5.0) and mean (4.3), indicating no clear shift in "how realistic" the modelled content is under AI assistance. The AI-assisted distribution is wider (IQR 4-6 vs 4-5), and the minimum drops to 1 in both groups, showing that low-correctness outcomes occur in both conditions. The practical implication is that AI support does not appear to systematically "protect" against incorrect modelling but also does not clearly degrade correctness at the group level.

Specificity
The AI-assisted group has a slightly higher median (5.0) than the Manual group (4.0), but means are close (AI-assisted 4.3, Manual 4.4) and distributions overlap quite strongly. The AI-assisted range is larger (1-7 vs 3-7), suggesting that AI can coincide with very high specificity for some participants, while others still produce vague or under-specified steps. This again points to heterogeneous use and/or effectiveness of AI support rather than a consistent uplift.

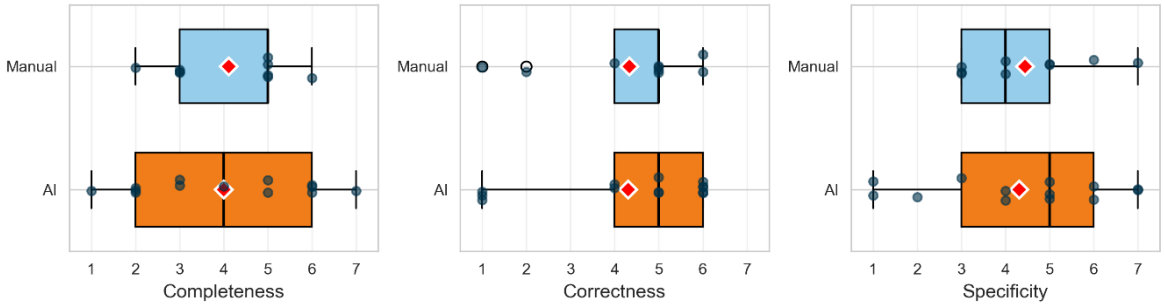


Figure 18 - Expert evaluations (1/2)

Usability
Both groups share the same median (4.0), but the AI-assisted group has a lower mean (3.5 vs Manual 4.0) and substantially wider spread (AI-assisted IQR 1-5 vs Manual 3-5). This pattern aligns with the broader observation that AI-assisted outcomes are less consistent: while some AI-assisted models reach strong usability, others are judged to have little practical value without major revision. Since usability is defined as "overall quality/readiness" of what has been modelled, this result flags a potential risk of low-end outcomes in the AI-assisted condition even when central tendency looks similar.

Clarity
Manual models are rated as clearer on average (median 5.0 vs 4.0 for AI-assisted). While central tendencies overlap, the AI-assisted group again shows broader dispersion (range 1-6 vs 2-6), suggesting that AI support can coincide with both clear and confusing outcomes depending on how the interaction unfolds. The difference is not statistically significant ($p = 0.409$), so this should be interpreted as a pattern rather than evidence of a systematic clarity effect.

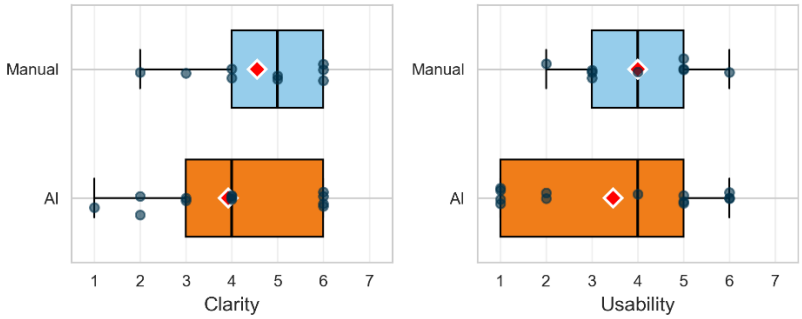


Figure 19 - Expert evaluations (2/2)

5.3.4 Perceived outcomes (post-task questionnaire)

Table 14, Figure 20, and Figure 21 summarise the items of the posterior questionnaire. Specifically, the items that were presented to both user groups: perceived correctness (Q1), perceived completeness (Q2), perceived efficiency (Q3), ease of use (Q4), and sufficient knowledge/confidence (Q6).

Table 14 - Perceived outcomes shared questions

Question	Group	n	Median	Mean	IQR	Range	p-value
Q1: Correct Models	AI-assisted	13	5.0	5.1	5.0-6.0	2.0-6.0	0.337
Q1: Correct Models	Manual	9	5.0	4.6	3.0-5.0	3.0-6.0	0.337
Q2: Complete Models	AI-assisted	13	3.0	3.8	2.0-6.0	1.0-6.0	0.659
Q2: Complete Models	Manual	9	4.0	3.3	3.0-4.0	1.0-5.0	0.659
Q3: Efficiency	AI-assisted	13	6.0	5.5	6.0-6.0	2.0-7.0	0.022
Q3: Efficiency	Manual	9	4.0	3.7	2.0-5.0	1.0-6.0	0.022
Q4: Ease of Use	AI-assisted	13	6.0	5.8	5.0-6.0	4.0-7.0	<0.001
Q4: Ease of Use	Manual	9	3.0	3.0	2.0-4.0	1.0-5.0	<0.001
Q6: Sufficient Knowledge	AI-assisted	13	6.0	5.3	5.0-7.0	2.0-7.0	0.290
Q6: Sufficient Knowledge	Manual	9	5.0	4.3	3.0-6.0	1.0-7.0	0.290

Correct models

Both groups have the same median (5.0), with the AI-assisted group slightly higher on average (mean 5.1 vs 4.6). The distributions overlap strongly ($p = 0.337$), but the shape differs: AI-assisted ratings are clustered in the 5-6 range (IQR 5-6) with one low outlier (min 2), whereas the Manual group is more concentrated in the mid-range (IQR 3-5, min 3). This suggests broadly similar perceived correctness at the centre, with the AI-assisted condition showing a slightly higher perceived correctness alongside occasional low-end experiences.

Complete models

Perceived completeness does not show a consistent upward shift with AI. The Manual group has a slightly higher median (4.0) than the AI-assisted group (3.0) ($p = 0.659$). More importantly, the AI-assisted group shows a much wider spread (IQR 2-6, range 1-6) than the Manual group (IQR 3-4, range 1-5). This wider dispersion cuts both ways: it includes very low perceived completeness for some AI-assisted participants, but it also includes high ratings up to 6, indicating that some participants felt their AI-supported model was quite complete. A possible explanation is that AI assistance can shift participants from generating structure themselves to

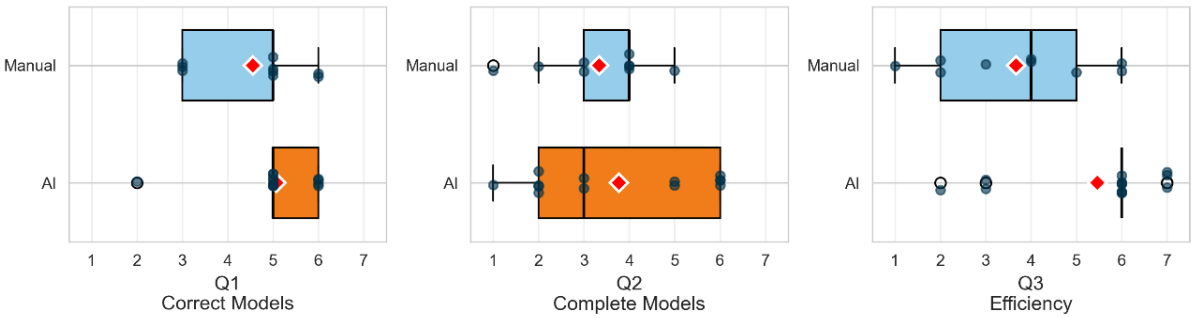


Figure 20 - Perceived outcomes (1/2)

evaluating and accepting suggestions, which may reduce critical self-checking. This aligns with known risks of overreliance on automated decision aids (automation bias/cognitive offloading), where users may stop searching for missing elements once an output appears plausible (Parasuraman & Riley, 1997; Lee & See, 2004). In this study, this mechanism would be consistent with faster submissions and higher self-assessed completeness, while expert-rated completeness still depends on the amount of effort invested in refining the model.

Efficiency

Efficiency shows a strong perceived benefit of AI support. The AI-assisted group is strongly right-shifted (median 6.0, mean 5.5) compared to the Manual group (median 4.0, mean 3.7), and the difference is statistically significant ($p = 0.022$). At the same time, the AI-assisted range still extends down to 2, meaning not everyone experienced the assistant as faster than manual modelling. In other words: many AI-assisted participants report clear efficiency gains, but there remains a minority for whom the tool did not feel efficient.

Ease of use

Ease-of-use also shows a strong separation between user groups. AI-assisted participants report consistently high ratings (median 6.0, IQR 5-6, range 4-7), while Manual participants cluster much lower (median 3.0, IQR 2-4, range 1-5). This indicates that the assistant substantially reduces the perceived effort and friction during modelling, making it easier for users to keep progressing through their modelling task.

Sufficient knowledge

Both groups are generally positive on confidence, with the AI-assisted group slightly higher (median 6.0 vs 5.0; $p = 0.290$). The Manual group shows broader spread (range 1-7) compared to AI-assisted group (range 2-7), again pointing to more varied self-efficacy in the Manual workflow, while AI-assisted responses cluster nearer the upper end (IQR 5-7)

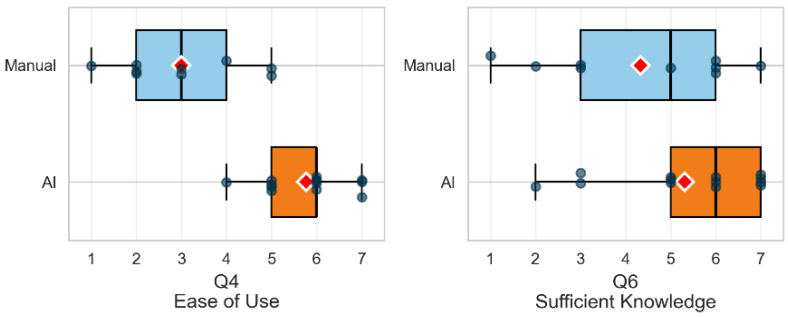


Figure 21 - Perceived outcomes (2/2)

Overall, the shared questionnaire items suggest that AI assistance mainly changes *how* participants experience the task rather than *how good* they think their final model is. Participants in both conditions judge their models as reasonably accurate, and perceived completeness does not shift consistently in favour of either workflow. What does change is the spread: in the AI-assisted group, perceptions of completeness are much more heterogeneous, ranging from "very incomplete" to "quite complete", whereas manual modellers cluster more tightly around the middle. In other words, AI support does not automatically make people feel they have a more complete model, but it does enable both stronger and weaker outcomes.

In addition to the shared items, the posterior questionnaire also included items specifically pertaining to the perceived qualities of the AI-assistant. Intuitively, these questions were only given to the AI-assisted user group, and contained questions about social approval, clarity of suggestions and motivations, the relevance of suggestions, the intent to use, and the quality of guidance form the assistant. Table 15 and Figure 22 show the statistical outcomes of said items:

Table 15 - Perceived AI-assistant qualities

Question	n	Median	Mean	IQR	Range
Q5: Social Approval	13	7.0	6.2	6.0-7.0	4.0-7.0
Q7: Clear Suggestions	13	6.0	5.8	5.0-6.0	4.0-7.0
Q8: Relevant Suggestions	13	5.0	5.1	5.0-6.0	3.0-7.0
Q9: Clear Motivations	13	6.0	5.1	5.0-6.0	2.0-7.0
Q10: Intent to Use	13	6.0	5.6	5.0-6.0	4.0-7.0
Q11: AI Guidance	13	5.0	4.8	4.0-6.0	2.0-7.0

Social approval

Responses to the social-approval item show that participants generally expect little resistance to using the assistant in their own environment. Most respondents indicate that people who matter in their work or study context would support or at least not disapprove of using this AI tool. Only a small minority are more cautious. In terms of adoption, this suggests that social norms are more enabling than constraining: if the tool is made available, participants do not anticipate strong social barriers to trying or continuing to use it.

Intent to use

Behavioural intention follows the same pattern. A clear majority indicate that they would use the assistant again for a similar task or recommend it to others, and no one expresses a strong preference to avoid it altogether. This points to a broadly favourable acceptance profile: participants do not only rate the tool positively in hindsight but also see it as something they would realistically choose in future onboarding or modelling tasks. In UTAUT terms, this is consistent with the assistant meeting the threshold for intention to use (N5).

Clarity of suggestions

Participants generally experience the assistant's suggestions as easy to follow. Most ratings are in the upper part of the scale, indicating that the phrasing and structure of the proposed steps are usually understandable without much extra effort. At the same time, a few lower scores show that clarity is not guaranteed for everyone or in every interaction. Overall, the assistant appears to communicate in a way that is typically clear enough to support modelling, but occasional unclear suggestions may still disrupt the flow for some users.

Suggestion relevance
Perceived relevance is positive on average but more uneven. Many participants report that suggestions match the task they are trying to complete and feel aligned with the supermarket scenario and modelling goal. However, some responses indicate that suggestions can occasionally miss the mark or feel only loosely connected. This combination of mostly good but sometimes weak relevance suggests that the assistant can provide highly fitting guidance for some users or topics, while for others its proposals are less well tuned to what they need at that moment. For deployment, this implies that further refinement of scoping and domain alignment could increase consistency.

Motivation clarity
Most participants feel that the assistant explains why it suggests certain changes in a way they can follow, but there are a few clearly negative experiences. For the typical user, the reasoning behind suggestions is sufficiently transparent to build at least a basic understanding of how the model is being shaped. For a small subset, however, the explanations appear opaque or incomplete, which risks undermining trust and makes it harder to critically assess the AI's output. The pattern is therefore one of "usually clear, sometimes confusing", indicating that explainability is a relative strength but not yet reliable for all users.

Guidance quality
Guidance quality is the most mixed of the AI-only items. Many participants feel that the assistant helps them move through the modelling task in a meaningful way, but others report that it does not guide them particularly well. Experiences range from perceiving the AI as a helpful, structured companion to experiencing it as a tool that offers local suggestions without providing a strong sense of direction. This wide spread is important: it indicates that the assistant can function as effective guidance, but does not do so consistently across users or sessions. For the overarching requirement of supporting users through the onboarding modelling task, guidance quality is therefore a key area where improvements could have substantial impact.

Taken together, the AI-assisted-only questions draw a clear distinction between acceptance and interaction quality. Acceptance-related indicators (social approval and intent to use) are strongly positive and tightly clustered, suggesting that participants are both comfortable with the idea of using the assistant and willing to do so again. Interaction-quality indicators (clarity, relevance, explainability, guidance) are also positive on average, but show more dispersion: most users experience clear, relevant and understandable suggestions, while a minority

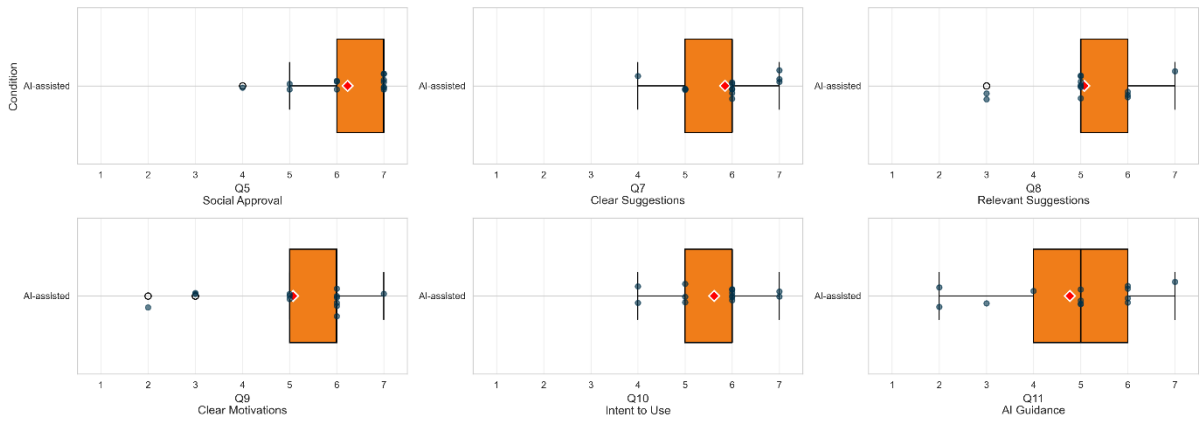


Figure 22 - Perceived AI-assistant qualities

encounter confusing, poorly aligned or insufficiently guiding behaviour. This mirrors the broader pattern in the results: the AI-assisted condition does not produce a single uniform experience, but a distribution in which some users receive excellent support and others receive support that is noticeably weaker.

5.3.5 Correlational patterns

To contextualise the comparisons above, we additionally explored Spearman rank correlations (ρ) between baseline variables and expert rubric scores, and baseline variables and post-task questionnaire ratings. These analyses are exploratory: with a small sample (n = 22 total; n = 13 AI-assisted for AI-only items), correlations are primarily used here to identify plausible relationships, not to infer causality. Appendices F, G, H, and I show the full correlation tables between demographic and measurement instruments, as well as internal correlation within demographics. The most relevant correlations are discussed in this section:

Within demographic

The correlations between the prior (demographic) questionnaire metrics are reported in Appendix I to identify potential baseline dependencies that could influence interpretation. Two relationships are statistically significant: age correlates positively with academic experience ($\rho = 0.51$, $p = 0.016$) and academic experience correlates negatively with process modelling experience ($\rho = -0.46$, $p = 0.032$). In contrast, group assignment shows only weak, non-significant correlations with the demographic variables (e.g., Group-Age $\rho = -0.05$, $p = 0.820$; Group-Process Modelling Experience $\rho = 0.26$, $p = 0.253$), suggesting that observed between-condition differences are unlikely to be driven by strong demographic imbalance. A weak trend suggests slightly higher AI experience in the Manual group (Group-AI Experience $\rho = -0.38$, $p = 0.081$), but this remains exploratory given the small sample size.

Group vs expert-rated model quality

Across the five expert rubric dimensions, the participant's group (Manual vs AI-assisted) shows negligible correlations (all $|\rho| \leq 0.19$, all $p \geq 0.403$), indicating no meaningful association between condition and expert ratings. This supports (and sharpens) the earlier rubric finding that the key difference between groups is not a uniform upward or downward shift in quality, but rather greater variability in the AI-assisted condition: outcomes include both more low- and high-end cases, producing similar central tendencies overall.

Time spent vs completeness

A key result is that more time spent is significantly associated with higher expert-rated Completeness ($\rho = 0.433$, $p = 0.044$).

This is intuitive given how completeness was operationalised: it reflects how far participants progressed within the assignment window. However, perceived completeness moves in the opposite direction, meaning time spent has a negative association with self-reported completeness (Q2: $\rho = -0.372$, $p = 0.088$). This tension is meaningful, because it implies that users who finished the task early might not have understood the scope of the assignment correctly, influencing later results. The positive association between time spent and expert-rated completeness could also suggest that completeness primarily benefits from iterative refinement rather than rapid first-pass structuring. This is important when interpreting AI assistance: if the assistant makes the model "look finished" earlier, participants may terminate the task sooner due to premature satisfaction, which is a known form of automation bias/cognitive offloading (Parasuraman & Riley, 1997; Lee & See, 2004). In practice, this would

mean that AI assistance can reduce time-to-submission, but completeness still improves when users continue to critically review, extend, and validate the model.

Group vs perceived metrics
Finally, correlations confirm the large separation already visible in the boxplots: group is strongly associated with Ease of Use (Q4: $\rho = 0.772$, $p < 0.001$) and moderately associated with Efficiency (Q3: $\rho = 0.508$, $p = 0.016$). This supports the interpretation that AI assistance primarily shifts the experience of producing a model (effort/fluency), while expert-rated quality remains comparable on average.

Age
In this study, age is not clearly linked to the quality of the models that participants produce. Older and younger participants score similarly on completeness, correctness, clarity, and usability. Within the AI-assisted group, there is a small trend that older participants report slightly lower intention to use the assistant again and find its motivations a bit less clear. This might imply that the motivations of the assistant can be further refined. However, this trend is not statistically significant, and the AI-assisted sample is small, so these results should be seen as a weak indication rather than firm evidence.

5.4 Answer to SQ6

This chapter set out to answer SQ6: "To what extent does the designed framework satisfy the requirements?". Overall, the evaluation shows that the framework enables AI-assisted modelling with quality on average comparable to manual modelling, clear gains in ease of use, and generally positive user acceptance. At the same time, several requirements are met only partially or inconsistently, mainly due to variability across users and sessions.

N1 required the assistant to deliver suggestions within 30 seconds. Assistant responses did not reliably stay within the 30-second target. However, for 38,5% of users it did, and for the remaining submissions it did not exceed the requirement drastically. Additionally, users consistently report high efficiency of the AI-assistant. This indicates that the extra response time was not perceived as much of a hindrance.

N2 required the assistant to enable users to produce a first usable model within 30 minutes. This requirement is assessed through the "completed" dimension of the expert rubric. Within the 30-minute timeframe, completeness is much more variable in the AI-assisted modelling group. This means that while it allows some people to achieve more complete models, it also hinders others in doing so. Considering this result, this requirement is deemed unsatisfied. However, it does raise questions to what extent a 30-minute timeframe is reasonable for such an assignment.

N3 required for the users to rate the AI-created model's usefulness at 60% or higher. While there was no explicit "usefulness" metric answered by the user, we can approximate such a metric by looking at the correctness model quality metric of the posterior questionnaire. There, we can see that users perceive the AI-assisted models quite consistently as correct, with an IQR of 5.0-6.0. Therefore, we consider this requirement as satisfied. An important consideration to be made here still is to what extent the users can make such a conclusion given their limited domain expertise in the test case.

N4 required produced models to be comparable in accuracy to consultant-led onboarding. This requirement can also only be evaluated indirectly. The study could not include a

consultant-led baseline, so there is no direct comparison to consultants' models. However, we can draw conclusions on the model quality through the results of the expert rubric. This rubric shows that the models created by the AI-assisted user group are on average very similar in correctness, specificity, and clarity, but that they show a much greater dispersion. This implies that while some users benefit from the assistant greatly and can create models deemed sufficient or even great by experts, other users might struggle more because of the assistant. Therefore, this requirement is deemed partially satisfied.

N5 required users to have a clear preference to using the assistant over starting from scratch. Participants that used the AI-assistant rated the task as easier and more efficient, while still judging the resulting model quality similarly as those made manually. The results show a positive attitude: most say they would use the assistant again or recommend it to others, and that they expect colleagues to support its use. While this study could not have the participants try both conditions, these findings support the idea that users liked working with the assistant and would choose it again for similar tasks. Therefore, this requirement is considered satisfied.

Overall, the results show a strongly divided picture. For some participants, the assistant works very well: it feels helpful and reasonably fast, they reach a usable and fairly complete model within the time limit, and experts rate those models as good. For others, the same assistant does not meet the requirements at all: progress is slower, models stay incomplete, and expert scores are clearly low. This split is especially visible in the expert-rated quality, which ranges from very strong to very weak, with more variability on the low end in the AI-assisted group.

In contrast, participants' perceived quality is much more positive and consistent: most users think their AI-assisted models are quite correct and useful, even when experts are more critical. This gap between what users feel about the model and how experts judge it suggests that the assistant can create a strong sense of confidence without always delivering equally strong outcomes. Because the study relies on a relatively small, homogeneous group and a single onboarding scenario, these findings are still preliminary. More research with larger and more diverse samples is needed to understand why the assistant helps some users much more than others, and how to reduce this variance between perceived and actual model quality.

6 CONCLUSION & DISCUSSION

This chapter discusses the contributions of this thesis. To this goal, the implications for scientific theory and for OrangeSpot in practice are discussed, as well as the limitations and future research directions that follow from these findings.

The remainder of this chapter is structured as follows. Section 6.1 provides explicit answers to the research questions. Section 6.2 summarises the key contributions of this thesis (artifact/design, empirical, and methodological) and positions them relative to prior work. Section 6.3 discusses the theoretical implications of the findings. Section 6.4 translates the results into practical implications and recommendations for OrangeSpot. Section 6.5 outlines the main limitations of the study and the artifact. Section 6.6 proposes a focused future work agenda aimed at improving consistency, validation, and evaluation strength. Finally, Section 6.7 closes the chapter with a short reflection.

6.1 ANSWERS TO RESEARCH QUESTIONS

This section consolidates the answers to the main research question and the six sub-questions addressed throughout this thesis. Rather than repeating the detailed findings from Chapters 2-5, it summarises the conclusions at the level of the questions themselves:

Main RQ: ***How can state-of-the-art process-modelling techniques be leveraged to guide domain experts in modelling business processes?***
This research demonstrates that guidance for domain experts can be realised by combining Large Language Models (LLMs) with an interaction design that embeds the assistant directly in the modelling workflow and keeps the user in control. In the OrangeSpot context, the central barrier is not a lack of process knowledge, but the difficulty of translating that knowledge into a structured blueprint without modelling expertise. LLM-based assistance is well-suited to reduce this “blank canvas” barrier because it can interpret incomplete and informal input, propose a first structure quickly, and support iterative refinement through dialogue.

The findings show that the assistant should not be designed as a fully automated modeller. Instead, effective guidance depends on a shared-responsibility approach: the assistant contributes speed and structure by proposing concrete modelling changes, while the user remains responsible for scoping, validating, and approving those changes.

Overall, the answer to the main research question is that state-of-the-art process-modelling techniques, specifically prompt-engineered LLMs, enhanced with a lightweight multi-agent structure, can guide domain experts when they are embedded in a human-in-the-loop modelling workflow. The evaluation confirms that such guidance can improve perceived ease of use and efficiency, while maintaining model quality on average comparable to manual modelling, but it also highlights that the main remaining challenge is reducing variance in outcomes across users and sessions.

SQ1: ***What are the requirements of a process modelling assistant?***
The requirements of a process modelling assistant in OrangeSpot’s onboarding context are both functional (what the assistant must do) and non-functional (how well it must do it). The problem investigation showed that consultancy-led onboarding produces high-quality models but is costly and capacity-constrained, while self-service onboarding often fails because users struggle to start from scratch and to produce consistent models without modelling expertise. A

modelling assistant is therefore required to reduce effort and increase autonomy without removing user control.

The functional requirements specify that the assistant must be able to:

- Automatically generate an initial process model from user input,
- Support step-by-step refinement so users can incorporate their own expertise,
- Follow OrangeSpot's blueprint structure and notation, and
- Work inside the existing process tool as an addition to the current way of modelling rather than a separate system.

The non-functional requirements define targets for performance and acceptance in onboarding-like use:

- Suggestions should be generated within about 30 seconds,
- A basic onboarding trajectory should be completable within about 30 minutes,
- Clients should rate the assistant's models as at least 60% useful,
- Model accuracy should be comparable to models created by implementation partners in typical cases, and
- Users should prefer using the assistant over starting from a blank canvas.

Together, these requirements provide a concrete answer to SQ1 and create a basis for selecting suitable generation methods, designing interaction patterns, and evaluating the artifact against measurable targets.

SQ2: *What is the state-of-the-art in process model generation methods?*
The literature review shows that the state-of-the-art in process model generation can be grouped into three main groups: rule-based Natural Language Processing (NLP), Process Mining (PM), and Generative AI / LLM-based approaches. NLP methods transform textual descriptions into models through stepwise linguistic processing and domain-specific mapping rules. Process Mining methods reconstruct models from event logs through discovery and conformance techniques, and they provide established quantitative evaluation measures tied to recorded behaviour. Generative AI approaches, finally, use LLMs to generate structured representations from informal input, and recent work improves these outputs through techniques such as prompt engineering, Retrieval-Augmented Generation (RAG), fine-tuning, and multi-agent systems.

This classification answers SQ2 by positioning LLM-based generation as the most recent and rapidly developing group of methods, while recognising that NLP and PM remain relevant in settings where inputs are more structured (clean text, event logs) and where quantitative evaluation against reference behaviour is feasible.

SQ3: *Which process model generation methods fit the requirements?*
When the method groups are compared against OrangeSpot's requirements, clear differences in practical suitability emerge. Rule-based NLP approaches assume relatively clean and unambiguous textual input and tend to be brittle when input quality varies, which conflicts with the reality of onboarding information that is incomplete, inconsistent, and expressed in different styles. Process Mining methods require event logs of sufficient quality and granularity, which are typically not available at the start of an OrangeSpot implementation. Both groups therefore struggle to meet the requirements for rapid first-model generation and interactive refinement in a self-service onboarding setting.

Within the Generative AI group, prompt-engineered use of general-purpose LLMs provides the best fit with OrangeSpot's constraints. Prompt engineering can steer LLMs toward OrangeSpot-specific terminology and blueprint structure without requiring large training datasets or heavy infrastructure. A lightweight multi-agent setup can further improve reasoning and validation by distributing roles (e.g., planning, proposing, reviewing), but the research also finds that complexity must be balanced against responsiveness and predictability. Heavier approaches such as RAG, GraphRAG, or extensive fine-tuning are currently less suitable for OrangeSpot's onboarding context because they require a stable and well-maintained knowledge base or large curated datasets.

Therefore, SQ3 is answered by concluding that an LLM-based assistant driven primarily by prompt engineering, enhanced with a lightweight multi-agent layer, best matches OrangeSpot's requirements and constraints.

SQ4: ***How can the quality of (generated) business process models be assessed?***
The literature indicates that model quality can be assessed through a combination of quantitative and qualitative approaches, each capturing different facets of quality. Quantitative evaluation is possible when a reference is available, for example through Process-Mining-based measures (fitness, precision, generalisation, simplicity) when event logs exist, or through distance-based metrics (such as Graph Edit Distance) when a ground-truth model is available for comparison. These methods enable repeatable benchmarking, but they depend on datasets, logs, or reference models that are rarely present in early onboarding contexts.

For OrangeSpot's use case, where no event logs are available at onboarding and where there is rarely a single authoritative "correct" model, qualitative evaluation plays a central role. Expert-based assessments allow process specialists to judge models on correctness, completeness, clarity, and usability. User-centred comparative evaluations capture perceived usefulness, trust, and preference: dimensions that align closely with OrangeSpot's requirements around autonomy and ease of use. As a result, SQ4 is answered by recommending a mixed evaluation strategy: use quantitative checks where feasible but prioritise expert review and user-centred measures as primary indicators of success.

SQ5: ***What would a framework for guided process modelling look like?***
The design and refinement phase translated the requirements and literature insights into a working artifact through four iterations (V1-V4). Across these iterations, two design lessons emerged as central. First, assistant behaviour should move away from automatic model editing toward suggestion-based interaction, because direct edits reduce transparency and user control and make it difficult to recover from mistakes. Second, scoping must be explicit: users need clear mechanisms to define where the assistant is allowed to operate, otherwise suggestions may jump across unrelated parts of the blueprint and become confusing or untrustworthy.

The resulting framework for guided process modelling is therefore built around shared responsibility. The assistant provides speed, structure, and candidate designs, supported by a lightweight multi-agent pipeline and prompt constraints, while the user retains authority through explicit scoping decisions and an accept/decline mechanism that makes review a natural step in the workflow.

Taken together, SQ5 is answered by concluding that guided process modelling should not be achieved by maximising automation, but by making the assistant's contribution transparent, controllable, and integrated into the modelling workflow.

SQ6: *To what extend does the designed framework satisfy the requirements?*
The evaluation shows that the framework enables AI-assisted modelling with model quality on average comparable to manual modelling, clear gains in ease of use, and generally positive user acceptance. At the same time, several requirements are satisfied only partially or inconsistently, mainly due to variability across users and sessions.

Regarding the non-functional requirements, responsiveness (N1) was not reliably within the 30-second target, although a substantial portion of sessions did meet it and users still reported high perceived efficiency. The "30-minute usable model" requirement (N2) was not met consistently: completeness outcomes in the AI-assisted condition were highly variable, suggesting that the assistant can help some users progress faster but can also hinder others. Perceived usefulness (N3) is supported by consistently positive user judgements of correctness and overall perceived quality. The accuracy requirement relative to implementation partners (N4) could only be assessed indirectly. Expert ratings indicate that AI-assisted models are comparable on average but show much greater dispersion, leading this requirement to be considered partially satisfied. Finally, user preference (N5) is supported by acceptance results indicating that participants generally liked working with the assistant and would use or recommend it.

Overall, SQ6 is answered by concluding that the designed framework partially meets the requirements, especially on usability and user experience, but that its current limitation is reliability: the assistant does not yet guide users toward consistently strong outcomes and reducing variance between perceived and expert-assessed quality remains a key challenge for further work.

6.2 Contribution

This thesis set out to design, implement, and evaluate a process modelling assistant that can help domain experts build an OrangeSpot process blueprint with less effort and more independence. This goal follows directly from the problem described in Chapter 1: organisations using OrangeSpot have a lot of process knowledge but turning that knowledge into a structured model is difficult without modelling expertise and often requires support from consultants. It also follows from the company context in Chapter 2, where reducing reliance on consultancy and lowering the "blank canvas" barrier were identified as important directions for OrangeSpot's onboarding.

Building on these chapters, this thesis makes three main contributions:

- An artifact and design knowledge: a working assistant and a guided modelling approach that was derived from requirements and iteratively refined (Chapters 2-4; SQ1 and SQ5).
- Empirical evidence: results from a controlled onboarding-like modelling task showing how the assistant changes user experience and modelling outcomes (Chapter 5; SQ6).
- A mixed evaluation approach: a way to evaluate modelling assistants in settings where there is no ground truth to compare against (Chapters 3 and 5).

Artifact and design contribution
The first contribution is the designed and implemented assistant itself, together with the design knowledge captured in the guided process modelling framework. This contribution can be traced through the thesis as follows:

- In Chapter 2, the thesis identified functional and non-functional requirements for a process modelling assistant in OrangeSpot's context. The requirements were grounded in the product context (who the users are, what onboarding needs, and why modelling is hard) and were also structured using a software quality model to make them measurable later in evaluation.
- In Chapter 3, the literature review showed that many existing approaches focus on generating models from text and evaluating them in controlled ways (often by comparing to a reference model). At the same time, OrangeSpot's context has important constraints: modelling is done by non-experts, inputs are incomplete, and there is typically no agreed ground-truth model for the onboarding situation.
- In Chapter 4, these requirements and constraints were translated into a working artifact through an iterative design and refinement process. The assistant was built in a safe test environment and connected to OrangeSpot's modelling actions through a backend layer, so that the assistant could propose concrete changes instead of only producing free text.

A key part of this contribution is that the assistant is not only a "chatbot next to the modeller". It is integrated into the modelling workflow: it can propose changes to the process model in a structured way, and these proposals are designed to be understandable and manageable for users. The chapter also documents how the assistant design evolved across multiple iterations (V1-V4) and how the final setup balances AI support with user control. The result of this work is summarised in the framework for guided process modelling (the explicit answer to SQ5 in Chapter 4), which captures the design choices that proved most promising in the thesis context.

Methodological contribution
The second contribution is methodological: the thesis provides a practical evaluation approach for interactive modelling assistants in settings where there is no single correct target model. This contribution connects directly back to the challenge discussed in Chapter 3: many evaluation methods in the literature rely on having a reference model or a "correct" solution, which is often not available in early onboarding or exploratory modelling tasks. In OrangeSpot's onboarding setting, the "right" model depends on purpose, can vary in level of detail, and different modellers might produce different yet acceptable outputs. To address this, Chapter 5 combines three complementary perspectives:

- Expert evaluation of the produced model (quality of the output).
- Behavioural/system metrics (what happened during the process, including performance indicators relevant to non-functional requirements).
- User questionnaires (how users perceive the tool, including acceptance-related measures).

The contribution is not just that these measures exist, but that they are combined in a way that makes trade-offs visible. For example, the evaluation can capture situations where users feel supported even when experts are more critical, and it can connect system-level constraints

(like responsiveness) back to requirements. This provides a repeatable evaluation template that is suitable for early-stage assistants where output quality and user acceptance both matter.

Empirical contribution
The empirical contribution of this thesis is what the evaluation reveals about AI-assisted process modelling in the studied setting (Chapter 5). The results show that introducing the assistant changes the modelling experience and can support progress, but it does not lead to uniform improvements in the quality of the resulting process models.

A key outcome is the strong variation across participants. Some users produce models that are judged as strong according to the expert assessment, while others end with models that remain incomplete or weaker in structure. This pattern matters because it suggests that the assistant can be effective, but that effectiveness depends on how it is used and how users interact with it. In addition, the evaluation indicates that users can feel supported and confident during the task even when expert ratings remain critical, pointing to a gap between perceived quality and expert-assessed quality.

Together, these findings provide evidence that the main challenge is not only whether an assistant can produce plausible modelling suggestions, but whether the interaction reliably leads users toward strong model outcomes. This empirical insight directly motivates the theoretical implications discussed in Section 6.2 and the practical recommendations that follow.

Contribution to the scientific body
In addition to the artifact and evaluation results, this thesis contributes to the scientific body of knowledge by focusing on the interaction part of AI-assisted process modelling. As discussed in Chapter 3, much prior work has studied LLM-based process modelling mainly as a one-shot task: a model is generated from a textual description and then evaluated against a reference model or an expert solution. This line of research is valuable for comparing generation quality, but it gives less attention to how people work with an assistant during modelling.

This thesis adds knowledge by treating the assistant as part of the modelling workflow and by studying how support is shaped by interaction choices such as scoping, proposing changes, and user review before applying changes (Chapter 4). Instead of only asking whether an LLM can produce a good model, the thesis investigates how an LLM can be used in a way that keeps the user in control and makes changes inspectable. The evaluation in Chapter 5 then provides evidence about what users experience and what kinds of outcomes emerge when support is provided through this interaction approach.

6.3 THEORETICAL IMPLICATIONS

The findings of this thesis have several implications for how AI-assisted process modelling should be understood theoretically. Rather than acting as a uniformly beneficial “upgrade” to manual modelling, the assistant behaves as an amplifier that interacts with user behaviour, domain knowledge and interaction design. This section discusses three interrelated implications: the assistant as an amplifier of model quality, the balance of responsibilities between user and assistant, and the gap between actual and perceived quality in AI-assisted modelling.

Assistant as an amplifier

The evaluation shows that introducing the assistant does not shift expert-rated model quality upwards or downwards in a straightforward way. Median and mean scores on the expert rubric are broadly similar between the AI-assisted and manual groups, but the dispersion is markedly higher in the AI condition across most dimensions. Completeness, for example, ranges from very low to very high in the AI group (1-7), while remaining more tightly clustered in the manual group, and similar patterns hold for specificity, clarity and usability. This suggests that the assistant functions as an amplifier: it enables some participants to reach highly complete and specific models within the same timebox, while others end up with weaker outcomes than the manual baseline. Theoretical accounts of AI-assisted modelling should therefore treat the assistant less as a uniform "boost" and more as a mechanism that increases variance in outcomes.

Shifting responsibility to the user: usability-expertise trade-off

The guided process modelling framework developed in Chapter 4 treats modelling as a shared responsibility between assistant and user. The iterative design from V1 to V4 makes explicit how shifting responsibilities along this spectrum changes both control and risk. Across the framework iterations, more responsibility was gradually moved from the assistant to the user. Later versions limited autonomous changes, made suggestions more explicit, and required users to select where and how changes should be applied.

Validation experts judged these versions as more usable: the interaction became more predictable and easier to follow. However, this higher usability comes with a trade-off. As responsibility shifts to the user, the framework increasingly relies on the user's modelling capability and their ability to formulate precise requests, interpret suggestions and judge their impact. In other words, shared responsibility improves transparency and control, but it also makes the system more sensitive to differences in user skill. Designing guided modelling support thus becomes a question of finding a workable balance: enough assistant autonomy to lower the barrier for inexperienced users, but enough user control to keep changes understandable and aligned with intent.

Expert- versus user-rated model quality

Another implication concerns the relationship between actual and perceived quality in AI-assisted modelling. Across conditions, expert rubric scores portray a mixed picture with considerable spread, especially in completeness and usability for the AI-assisted group. In contrast, AI-assisted participants report high perceived correctness, efficiency, ease of use and sufficient knowledge to work with the tool. This gap between user experience and outcome suggests that users often feel they are producing good models with the assistant, even when expert evaluations are more critical. Theoretically, this points to problem. Participants may overestimate the quality of AI-assisted models, particularly when they lack domain knowledge or experience with process modelling.

If users cannot easily detect omissions, inconsistencies or unrealistic flows, the assistant's fluent language, structured suggestions and positive interaction experience can create a strong sense of confidence that is not fully warranted by the underlying model quality. From a UTAUT perspective, effort expectancy and performance expectancy are clearly positive for the assistant, but this does not automatically imply that objective performance has improved.

This misalignment has two theoretical consequences. First, it challenges the assumption that user acceptance and satisfaction are reliable proxies for quality in AI-assisted design tools. A system can be widely liked, recommended and perceived as efficient while still producing

outputs of uneven or uncertain quality. Second, it emphasises the need to treat AI-assisted modelling as an additional educational problem: users must learn not only how to use the assistant, but also how much to trust its outputs and under which conditions that trust should be limited.

6.4 PRACTICAL IMPLICATIONS

This section translates the findings of this thesis into concrete implications for OrangeSpot. It focuses on how the assistant should be positioned in the onboarding offering, what this means for the product design, how consultants and clients should work with it in practice, and how the current prototype relates to the original requirements and a possible deployment to production.

Positioning the assistant

As discussed in the previous section, the evaluation shows that the assistant does not systematically increase or decrease average expert-rated model quality compared to manual modelling, but that it widens the spread of outcomes. At the same time, user experience is consistently positive: AI-assisted participants perceive the task as easier and more efficient, rate their models as correct, and indicate that they would use or recommend the assistant again.

In practical terms, this means the assistant, in its current state, should be positioned as inspiration, rather than a replacement for the consultant. Its primary value is removing the blank-canvas barrier and offering structured suggestions that help users articulate and refine their processes. For OrangeSpot, this means it should think about the presentation towards clients. This message should emphasize that the assistant provides inspiration and structure which must be reviewed and adapted, rather than accepted without further thought.

Product implications

The validation phase across V1-V4 provides clear guidance on which design choices are mature enough to keep, and which require further work. The move from a single, fully automatic agent (V1-V2) to a suggestion-based, human-in-the-loop design (V3-V4) greatly improved control and predictability. The final version adds explicit context selection, which pilots showed to reduce confusing scope jumps and mixed suggestions. Taken together, this suggests that a production assistant should retain at least three elements as core principles:

- suggestions instead of automatic edits.
- explicit scoping by the user (context selection).
- separate, clearly labelled modes for process and entity changes.

These patterns operationalise the intended shared-responsibility framework and are supported by both pilot findings and the main evaluation. For future iterations, the high variance in expert-rated quality and the relatively optimistic user perceptions indicate that the focus should not be on adding more features, but on improving consistency. Concretely, this means further exploring the balance between user interaction and assistant responsibility: tightening guardrails on what the assistant can change, making its suggestions more transparent and revisable, and using monitoring to understand where users struggle to steer it. The product trajectory should therefore prioritise making the assistant more predictable and aligned with user intent. Suggestions for such improvements are made in section 6.5.

Workflow implications

The observed variance in outcomes and the gap between expert and user judgments mean that

the assistant, in its current state, should not replace consultants but change when and how they are involved. A possible pragmatic approach could be a hybrid onboarding workflow:

- Consultants perform an initial structuring step. As in current onboarding, consultants help define the main end-to-end process, key sub-processes and critical entities. This establishes a sound backbone and reduces the risk that users make critical mistakes right from the start.
- Clients use the assistant to elaborate and refine. Once a basic structure exists, domain experts on the client side use the assistant to add detail, adjust responsibilities and connect roles, applications and data. In this phase, the assistant primarily accelerates work and offers inspiration, while the predefined structure supports consistency.
- Consultants remain available as a safety net. For clients who get stuck, for complex domains or for critical processes, consultants stay involved as reviewers and sparring partners. Monitoring data or simple quality checks can be used to flag models that should be reviewed before being relied on in change projects.

This hybrid setup directly addresses the variance problem: weaker users are less likely to drift into poor models because they start from a consultant-prepared structure and can escalate when needed. It also preserves consultancy-led onboarding as an option for clients who prefer a more guided trajectory or whose processes are too sensitive or complex for self-service alone. To make this work, OrangeSpot should again explicitly communicate that assistant output is not a substitute for consultant validation and provide short guidance on how to use the assistant effectively.

Path to production

Revisiting the non-functional requirements from Chapter 2 shows that the assistant is close to deployable, but not without deliberate choices from OrangeSpot.

For N1 (suggestions within ~30 seconds), the assistant is almost but not consistently within target. Median response times are close to 30 seconds and there are no extreme delays, but the guarantee is not hard. OrangeSpot will need to decide whether this level of latency is acceptable in practice, or whether to prioritise further optimisation. If lower latency is deemed critical, this could mean simplifying the agent architecture, experimenting with lighter models, or postponing until faster LLMs become available.

N2 (first usable model within ~30 minutes) is clearly more problematic. Due to large differences in completeness between AI-assisted participants, several users did not reach a clearly usable model within the allotted time. This raises the question whether the original 30-minute target is realistic. In practice, OrangeSpot can either reframe N2 (for example, aiming for a first backbone or high-level model within 60 minutes), or keep N2 as an aspirational benchmark and treat current performance as a step towards it rather than a hard gate.

N3 (perceived usefulness ≥ 60%) and N5 (preference for using the assistant over starting from scratch) are clearly met: users experience the assistant as helpful, efficient and worth reusing. This lowers adoption risk and supports introducing the assistant into real onboarding scenarios. N4 (accuracy comparable to consultant-led onboarding) is only partially met. On average, expert-rated quality is comparable between AI-assisted and manual models, but with much higher dispersion for the assistant: some models are very strong, others clearly fall below consultant-led quality. This dispersion is the key strategic question for OrangeSpot: whether to

deploy now, accepting variability and compensating with guardrails and human oversight, or to delay wider rollout until further iterations improve consistency.

Given this balance, the assistant in its current form is not yet a drop-in replacement for consultant-led onboarding, but it is a realistic candidate for supported self-onboarding and as a co-pilot in consultant-led trajectories, especially when embedded in a hybrid workflow where consultants review or refine assistant-generated models.

From a technical standpoint, the prototype is already very close to OrangeSpot's production stack: it uses the same blueprint structure, similar modelling interactions and a familiar architecture with a browser-based frontend, a thin backend exposing CRUD and assistant endpoints, and a separate orchestration component. This makes the integration path relatively lightweight. The main technical steps towards integration for OrangeSpot are:

- Introduce a suggestions API endpoint in the production backend.
  This endpoint receives proposed changes from the assistant, validates them and exposes them to the frontend as suggestions, mirroring the behaviour used in the test environment.
- Map accepted suggestions onto existing CRUD operations.
  When a user accepts a suggestion, the frontend translates it into standard CRUD calls on the existing blueprint API. This keeps the assistant as a thin layer on top of the current modelling infrastructure rather than a parallel system.

With these steps, the assistant can be introduced in OrangeSpot's production environment.

6.5 LIMITATIONS

Although the evaluation provides useful first evidence on how the assistant shapes modelling experience and outcomes, the conclusions should be interpreted within clear boundaries set by the study setup and the prototype itself. This section therefore outlines the main limitations that affect validity and interpretability, covering the sample and use-case choices, the timeboxed between-subject design, and measurement challenges. It also reflects on limitations introduced by the assistant's design evolution and open-ended interaction style, which make it harder to differentiate tool capability from user strategy.

Sample and participant limitation.
A first major limitation is the composition and size of the user sample. The evaluation relied on a relatively small, homogeneous group of participants (13 in the AI-assisted condition and 9 in the manual condition), all male and predominantly master's students with academic rather than professional process-modelling experience.

Their intrinsic motivation to optimise model quality was likely lower than that of real clients, and their domain expertise in the chosen scenario was limited. This combination can distort both expert-rated outcomes and perceived usefulness: some participants may have stopped once the model felt "good enough" for the assignment, while others may have lacked the domain knowledge to notice important omissions. This is consistent with the thought of automation bias and cognitive offloading discussed in section 5.3.4. Additionally, the consistently positive perceptions of correctness and usefulness in the AI group, despite more variable expert ratings, are a plausible support for this concern.

Future evaluations should therefore involve larger, more diverse, and more representative samples, ideally including OrangeSpot clients, consultants, and operational staff in different sectors. Incentive structures could also be aligned more directly with model quality (for example through graded assignments or small rewards for completeness) to better approximate real-world motivation.

Use-case limitation.
The evaluation was conducted on a single onboarding scenario in the supermarket domain. This scenario was chosen because the expected sample of students was assumed to have at least some prior familiarity with supermarkets, while keeping the task identical across participants for comparability. In practice, this assumption did not fully hold. Participants reported limited understanding of the underlying processes and terminology of the use case domain, which likely constrained model quality in both conditions and made it harder to exploit the assistant's potential.

At the same time, the student sample likely lacked the inherent motivation that real OrangeSpot clients have when modelling their own processes. For them, the scenario was an abstract exercise rather than a pressing business problem, which may have reduced the depth of engagement and the effort invested in refining models, again limiting external validity.

In hindsight, and especially given the relatively small sample size, it might have been preferable to allow multiple use-cases that better matched participants' actual domain knowledge, even at the cost of additional evaluation overhead or fewer participants. Future studies should therefore focus more on realistic tasks that match participants' actual expertise and motivation, rather than forcing everyone into one shared scenario.

Experimental design limitation.
The study used a between-subject design with a single, timeboxed session of 30 minutes per participant. While this setup reduces learning and carry-over effects between conditions, it also limits causal inference and comparability of preferences. Participants only experienced one way of working (manual or AI-assisted), so differences in perceived ease of use and efficiency cannot be complemented with within-person comparisons of using the assistant. Moreover, a single short session does not capture how users might adapt over time, nor how the assistant might perform once users have gotten used to its interaction patterns. The time limit also entangles completeness with progress: lower completeness scores can reflect either slow modelling or deliberate focus on a subset of the scenario. Future evaluation designs should therefore include within-subject or crossover setups where participants experience both workflows, as well as longitudinal or multi-session studies that allow learning curves, sustained usage, and preference changes to be observed more robustly.

Measurement and interpretation limitation.
This study combines several types of measurements: expert rubric scores, behavioural metrics (such as number of actions and time spent), and UTAUT-inspired questionnaires. Using multiple measures is helpful because it gives a broader view of how the assistant behaves. At the same time, it also makes the results harder to interpret.

A central limitation is the "completeness" dimension in the expert rubric. In this study, completeness mainly captures how far participants got within the 30-minute session. This creates a knock-on effect for the other rubric dimensions. If a model only covers a small part of the process, experts can only judge the accuracy, clarity or usability of that small part.

Participants who progressed further will naturally be exposed to more opportunities for errors, inconsistencies or unclear steps, and their scores on these dimensions may therefore look worse simply because more of the process was present to be evaluated. As a result, differences in completeness can blur the interpretation of differences in accuracy or other quality dimensions.

Future work should therefore separate progress from quality more explicitly. One way to do this is to define a minimal viable process model (for example: clear start and end, a complete main flow, and no obvious gaps) and measure how long it takes users to reach that point. The time to reach such a minimal viable model can then be analysed separately from the quality of the final model. In addition, quality could be assessed on comparable parts of the process (e.g. the main flow only), so that scores on accuracy or clarity are less dependent on how much of the process the participant managed to model within the timebox.

Design evolution and interaction limitation.
A final limitation concerns how the assistant itself was designed and evolved. In this project, the starting point was very ambitious. In the early versions (V1-V2), the assistant was allowed to do practically anything for the user. Only when problems such as hallucinated changes, unclear edits and confusing scope jumps appeared, responsibilities were gradually pulled back. By V4, the design had moved to a suggestion-based, human-in-the-loop setup with explicit context selection and separate modes.

In hindsight, this outward-in approach makes it more understandable that the results show high variability. Users were given a lot of freedom to ask the assistant almost anything at any time, and the assistant could respond with a wide range of changes. Some participants used this freedom very effectively, others asked vague or overly broad questions which led to scattered suggestions and slower progress. Because the interaction remained open-ended and chat-based, it is difficult to separate the assistant's capabilities from differences in user strategy.

A more pragmatic design path might be to invert this logic: first study what users want to do with the assistant in concrete terms (for example: generate a first backbone, refine step names, check for missing handovers), and then build the assistant up feature by feature around those specific tasks. Each capability could be introduced as a narrow, well-defined operation and expanded only when it proves useful and reliable. This also indirectly suggests that a general-purpose chatbot interface may not be the optimal UI for guided process modelling; more structured controls integrated into the modelling environment might fit better. However, this study did not systematically compare interface styles, so the question of the "best" interaction pattern remains open for future work.

6.6 FUTURE WORK

The current prototype shows that an LLM-based assistant can make process modelling feel easier and more accessible, but it also reveals limitations in robustness, consistency and evaluability. Future work should therefore focus on strengthening the empirical evidence, exploring alternative technical options, refining the prototyping strategy, and systematically increasing measurement capabilities. This section outlines a corresponding agenda along these four topics.

Stronger evaluation

The evaluation in this thesis provides useful first insights, but it remains constrained by a between-subject design, a relatively small and homogeneous sample, and a single synthetic scenario. Future studies should adopt stronger and more varied designs to understand when, for whom and under what conditions the assistant is effective.

- Allowing the same participants to model both with and without the assistant, in counterbalanced order, would support more robust causal claims about relative effort, perceived usefulness and expert-rated quality. Crossover designs could also reveal learning effects and whether skills acquired with the assistant transfer to manual modelling (and vice versa).
- Evaluations should move beyond student populations to include consultants, client stakeholders and operational staff from different organisations and sectors. This would improve external validity and uncover how domain expertise, organisational roles and motivation interact with AI support.
- Instead of a single onboarding scenario, future studies should cover a portfolio of processes (e.g. HR, finance, logistics, customer support) and a range of complexities. This will show whether the assistant generalises across domains or requires domain-specific tuning.
- Field pilots with real OrangeSpot projects are needed to see how the assistant behaves outside controlled conditions. Logging usage over time, across multiple onboarding trajectories, would reveal whether perceived usefulness, and quality improve or deteriorate as users gain experience, and how the assistant fits into actual project workflows.

Exploring other technologies

The current assistant relies on a general-purpose LLM, constrained mainly through prompting and multi-agent orchestration. As OrangeSpot accumulates more real models and as the model landscape evolves, two main directions become relevant: using collected process examples more explicitly and leveraging newer foundation models and agent setups.

- Over time, OrangeSpot can build a curated corpus of high-quality blueprints and modelling decisions (with proper governance and anonymisation). This corpus would make it possible to explore fine-tuning or continued training so that the model better internalises OrangeSpot's conventions, terminology and typical level of detail.
- The same corpus can also be used without changing the base model, by grounding suggestions in a structured repository of existing processes, templates and best practices. Retrieval-augmented generation (RAG) or graph-based variants (GraphRAG) could then be applied. This would turn collected blueprints into a living knowledge base, supporting more consistent, organisation-specific guidance.
- As new foundation models become available, future work should systematically compare them on quality, latency, controllability and cost for this specific use-case. Different models may be better suited for different sub-tasks (for example, natural-language clarification vs. structural transformation of blueprints), and mixing models could improve both responsiveness and reliability. With the current technological setup of the assistant, newer models are easily interchangeable, thanks to the LangChain framework.
- If future models become significantly faster, cheaper or more reliable, this opens the door to more complex agent configurations. For instance, specialised sub-agents could

focus on validation, explanation, optimisation or documentation. The current multi-agent design can act as a baseline. Future work can then examine in which situations added architectural complexity improves outcomes and when a simpler, single-agent setup is sufficient.

- Even without new models or training, there is still room to improve the current assistant through better prompting. Future work can experiment with more structured system prompts, clearer step-by-step instructions, richer few-shot examples and specialised prompt templates for common modelling actions. These changes are relatively low-cost and could already help to stabilise behaviour, reduce errors and make the assistant's responses more predictable.

Improving measurement instruments

Finally, future work should exploit the current architecture's logging capabilities to improve measurement. Richer data on how the assistant is used will help diagnose issues, target design changes and support stronger evaluations.

- Prompts entered by users can be automatically classified into high-level intent categories (e.g. "ask for next steps", "clarify terminology", "edit existing model", "meta questions about the tool"). This will show which use-cases dominate in practice and where additional features or guardrails are needed.
- For each type of suggestion or AI action, acceptance and rejection rates can be logged over time, broken down by user type, scenario and project phase. Low acceptance rates may indicate poor relevance or trust; very high rates without corresponding quality improvements may signal automation bias.
- Where expert ratings or project outcomes are available, they can be linked back to usage logs to explore which interaction patterns correlate with higher-quality models or faster progress.
- On top of this data, internal dashboards can provide real-time monitoring of assistant usage, performance and reliability, supporting both day-to-day product stewardship and future research.

Because the current test environment already centralises interactions with the assistant, these measurement extensions can be implemented incrementally and used both in experimental studies and in future client pilots.

Prototyping strategy

As discussed in section 6.4, this research followed an outward-in prototyping strategy. The first versions of the assistant behaved as an almost all-purpose agent that could interpret free form prompts and directly manipulate the blueprint, and capabilities were scaled back when pilots exposed issues. Even in the current version, the chatbot still gives users considerable freedom to move in unproductive directions or to use the assistant in ways that are hard to predict and control. An alternative pragmatic prototyping strategy for future iterations is to start small and build up:

- Use the testing environment to discover real usage patterns. The existing test environment can be used to observe what users try to do with the chatbot, which prompts they submit, which problems they are trying to solve, and where they struggle. Analysing this data will help identify a handful of recurring intents that are genuinely valuable in practice.

- Use this set of limited, well-defined AI actions. Rather than starting from full autonomy and then restricting behaviour, a next iteration could offer only a small, carefully chosen to set of assistant actions that are easy to specify, test and verify based on the usage patterns from the test environment.
- Promote common actions to first-class features**.** Once common patterns are identified, they should be turned into direct features in the UI (buttons, wizards, sidebars) rather than remaining implicit in free-text prompts. The chatbot could then become a secondary feature or eventually phase out altogether.
- Gradually broaden capability as reliability is demonstrated. As specific AI actions prove reliable in testing and pilots, new actions can be added and refined. This incremental approach supports higher assistant consistency, clearer expectations for users, and a more controlled path towards a production-ready product.

This “start small, build up” strategy aligns the evolution of capabilities with evidence of reliability and makes it easier to reason about the assistant’s behaviour and risks.

6.7 Reflection

In summary, this chapter has shown that the thesis delivers three main contributions: a guided process modelling assistant embedded in OrangeSpot’s workflow, empirical evidence from a realistic onboarding-like task, and a mixed evaluation approach suited for settings without a clear ground-truth model. Together, the results suggest that the assistant acts less as a uniform performance boost and more as an amplifier of user behaviour, increasing variance in expert-rated model quality while consistently improving perceived ease and efficiency. Theoretical implications centre on this amplifier role, the usability-expertise trade-off introduced by shared responsibility between user and assistant, and the observed gap between expert- and user-rated quality, which highlights the need to treat AI-assisted modelling also as an educational challenge. Practically, the chapter argues that OrangeSpot should position the assistant as a source of inspiration and structure rather than a consultant replacement, potentially work with a hybrid onboarding workflow, and prioritise improving consistency and guardrails over adding new features. At the same time, the discussion emphasises clear limitations related to the small, homogeneous sample, use-case scenario, timeboxed between-subject design and open-ended interaction style, which restrict the generalisability of the findings. Building on these constraints, the proposed future research agenda calls for stronger and more diverse evaluations, exploration of alternative technical setups (including better grounding and newer models), a more incremental prototyping strategy, and richer measurement and monitoring of real-world usage to progressively strengthen the assistant’s reliability and impact.

REFERENCES

1. ACKERMANN, L., & VOLZ, B. (2013). MODEL[NL]GENERATION: NATURAL LANGUAGE MODEL EXTRACTION. PROCEEDINGS OF THE 2013 ACM WORKSHOP ON DOMAIN-SPECIFIC MODELING, 45-50. HTTPS://DOI.ORG/10.1145/2541928.2541937
2. ALEXANDER, I. (2007). A TAXONOMY OF STAKEHOLDERS. INTERNATIONAL JOURNAL OF TECHNOLOGY AND HUMAN INTERACTION, 1, 23-59. HTTPS://DOI.ORG/10.4018/JTHI.2005010102
3. APPIAN. (2025). APPIAN: THE PROCESS COMPANY. HTTPS://APPIAN.COM/
4. AUGUSTO, A., CONFORTI, R., DUMAS, M., & LA ROSA, M. (2017). SPLIT MINER: DISCOVERING ACCURATE AND SIMPLE BUSINESS PROCESS MODELS FROM EVENT LOGS. 2017 IEEE INTERNATIONAL CONFERENCE ON DATA MINING (ICDM), 1-10. HTTPS://DOI.ORG/10.1109/ICDM.2017.9
5. BELLAN, P., DRAGONI, M., & GHIDINI, C. (2020). A QUALITATIVE ANALYSIS OF THE STATE OF THE ART IN PROCESS EXTRACTION FROM TEXT. PROCEEDINGS OF THE AIXIA 2020 DISCUSSION PAPERS WORKSHOP, 2776, 19-30. HTTPS://BIA.UNIBZ.IT/ESPLORO/OUTPUTS/CONFERENCEPROCEEDING/A-QUALITATIVE-ANALYSIS-OF-THE-STATE/991006694797001241
6. BEHESHTI, A., YANG, J., SHENG, Q. Z., BENATALLAH, B., CASATI, F., DUSTDAR, S., *ET AL*. (2023). *PROCESSGPT: TRANSFORMING BUSINESS PROCESS MANAGEMENT WITH GENERATIVE ARTIFICIAL INTELLIGENCE*. IN PROC. OF THE 2023 IEEE INTERNATIONAL CONFERENCE ON WEB SERVICES (ICWS). IEEE. (ARXIV PREPRINT ARXIV:2306.01771)
7. BELLAN, P., DRAGONI, M., & GHIDINI, C. (2022). EXTRACTING BUSINESS PROCESS ENTITIES AND RELATIONS FROM TEXT USING PRE-TRAINED LANGUAGE MODELS AND IN-CONTEXT LEARNING. IN J. P. A. ALMEIDA, D. KARASTOYANOVA, G. GUIZZARDI, M. MONTALI, F. M. MAGGI, & C. M. FONSECA (EDS.), ENTERPRISE DESIGN, OPERATIONS, AND COMPUTING (PP. 182-199). SPRINGER INTERNATIONAL PUBLISHING. HTTPS://DOI.ORG/10.1007/978-3-031-17604-3_11
8. BERTI, A., SCHUSTER, D., & VAN DER AALST, W. M. P. (2024). ABSTRACTIONS, SCENARIOS, AND PROMPT DEFINITIONS FOR PROCESS MINING WITH LLMS: A CASE STUDY. IN J. DE WEERDT & L. PUFAHL (EDS.), BUSINESS PROCESS MANAGEMENT WORKSHOPS (PP. 427-439). SPRINGER NATURE SWITZERLAND. HTTPS://DOI.ORG/10.1007/978-3-031-50974-2_32
9. BROWN, T. B., MANN, B., RYDER, N., SUBBIAH, M., KAPLAN, J., DHARIWAL, P., NEELAKANTAN, A., SHYAM, P., SASTRY, G., ASKELL, A., AGARWAL, S., HERBERT-VOSS, A., KRUEGER, G., HENIGHAN, T., CHILD, R., RAMESH, A., ZIEGLER, D. M., WU, J., WINTER, C., ... AMODEI, D. (2020). LANGUAGE MODELS ARE FEW-SHOT LEARNERS (NO. ARXIV:2005.14165). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2005.14165
10. ANTRHOPIC (N.D.) BUILDING EFFECTIVE AI AGENTS. HTTPS://WWW.ANTHROPIC.COM/ENGINEERING/BUILDING-EFFECTIVE-AGENTS
11. BUIJS, J. C. A. M., VAN DONGEN, B. F., & VAN DER AALST, W. M. P. (2014). *QUALITY DIMENSIONS IN PROCESS DISCOVERY: THE IMPORTANCE OF FITNESS, PRECISION, GENERALIZATION AND SIMPLICITY*. INTERNATIONAL JOURNAL OF COOPERATIVE INFORMATION SYSTEMS, 23(1), ARTICLE 1440001. HTTPS://DOI.ORG/10.1142/S0218843014400012
12. BUSCH, K., ROCHLITZER, A., SOLA, D., & LEOPOLD, H. (2023). *"JUST TELL ME": PROMPT ENGINEERING IN BUSINESS PROCESS MANAGEMENT*. IN H. VAN DER AA, D. BORK, H. A. PROPER, & R. SCHMIDT (EDS.), ENTERPRISE, BUSINESS-PROCESS AND INFORMATION SYSTEMS MODELING - BPMDS 2023 AND EMMSAD 2023 (LNBIP, VOL. 479, PP. 3-11). SPRINGER, CHAM
13. CELONIS. (N.D.). *CELONIS: PROCESS INTELLIGENCE AND PROCESS MINING*. HTTPS://WWW.CELONIS.COM/

14. CHOWDHARY, K.R. (2020). NATURAL LANGUAGE PROCESSING. IN: FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE. SPRINGER, NEW DELHI. HTTPS://DOI-ORG.EZPROXY2.UTWENTE.NL/10.1007/978-81-322-3972-7_19
15. DACLIN N., MALLEK-DACLIN S., ZACHAREWICZ G. (2024). GENERATIVE AI FOR BUSINESS MODEL GENERATION (GAI4BM): FROM TEXTUAL DESCRIPTION TO BUSINESS PROCESS MODEL. PROCEEDINGS OF THE 10TH INTERNATIONAL FOOD & OPERATIONS SIMULATION WORKSHOP (FOODOPS 2024), 013. DOI: HTTPS://DOI.ORG/10.46354/I3M.2024.FOODOPS.013
16. DIKICI, A., TURETKEN, O., & DEMIRORS, O. (2018). FACTORS INFLUENCING THE UNDERSTANDABILITY OF PROCESS MODELS: A SYSTEMATIC LITERATURE REVIEW. INFORMATION AND SOFTWARE TECHNOLOGY, 93, 112-129. HTTPS://DOI.ORG/10.1016/J.INFSOF.2017.09.001
17. DMITRIEVNA, Y. (2025, APRIL 24). AI AGENTS EXPLAINED: FROM THEORY TO PRACTICAL DEPLOYMENT. N8N BLOG. HTTPS://BLOG.N8N.IO/AI-AGENTS/#WHAT-ARE-AI-AGENTS
18. DRAW.IO. (N.D.). ABOUT DRAW.IO. RETRIEVED APRIL 4, 2025, FROM HTTPS://WWW.DRAWIO.COM/ABOUT
19. EDGE, D., TRINH, H., CHENG, N., BRADLEY, J., CHAO, A., MODY, A., TRUITT, S., METROPOLITANSKY, D., NESS, R. O., & LARSON, J. (2025). FROM LOCAL TO GLOBAL: A GRAPH RAG APPROACH TO QUERY-FOCUSED SUMMARIZATION (NO. ARXIV:2404.16130). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2404.16130
20. ES, S., JAMES, J., ESPINOSA ANKE, L., & SCHOCKAERT, S. (2024). RAGAS: AUTOMATED EVALUATION OF RETRIEVAL AUGMENTED GENERATION. IN N. ALETRAS & O. DE CLERCQ (EDS.), PROCEEDINGS OF THE 18TH CONFERENCE OF THE EUROPEAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS: SYSTEM DEMONSTRATIONS (PP. 150-158). ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. HTTPS://ACLANTHOLOGY.ORG/2024.EACL-DEMO.16/
21. FERREIRA, R. C. B., THOM, L. H., & FANTINATO, M. (2017). A SEMI-AUTOMATIC APPROACH TO IDENTIFY BUSINESS PROCESS ELEMENTS IN NATURAL LANGUAGE TEXTS: PROCEEDINGS OF THE 19TH INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS, 250-261. HTTPS://DOI.ORG/10.5220/0006305902500261
22. FEUERRIEGEL, S., HARTMANN, J., JANIESCH, C., & ZSCHECH, P. (2024). GENERATIVE AI. BUSINESS & INFORMATION SYSTEMS ENGINEERING, 66(1), 111-126. HTTPS://DOI.ORG/10.1007/S12599-023-00834-7
23. FILL, H.-G., FETTKE, P., & KÖPKE, J. (2023). *CONCEPTUAL MODELING AND LARGE LANGUAGE MODELS: IMPRESSIONS FROM FIRST EXPERIMENTS WITH CHATGPT*. ENTERPRISE MODELLING AND INFORMATION SYSTEMS ARCHITECTURES, 18, 1-15.
24. FINE-TUNING | HOW-TO GUIDES. (N.D.). HTTPS://WWW.LLAMA.COM/DOCS/HOW-TO-GUIDES/FINE-TUNING/
25. FLOWISEAI. (N.D.-A). FLOWISE [COMPUTER SOFTWARE]. GITHUB. RETRIEVED AUGUST 14, 2025, FROM HTTPS://GITHUB.COM/FLOWISEAI/FLOWISE
26. FLOWISEAI. (N.D.-B). INTRODUCTION. FLOWISE DOCUMENTATION. RETRIEVED AUGUST 14, 2025, FROM HTTPS://DOCS.FLOWISEAI.COM/
27. FRIEDRICH, F., MENDLING, J., & PUHLMANN, F. (2011). PROCESS MODEL GENERATION FROM NATURAL LANGUAGE TEXT. LECTURE NOTES IN COMPUTER SCIENCE (INCLUDING SUBSERIES LECTURE NOTES IN ARTIFICIAL INTELLIGENCE AND LECTURE NOTES IN BIOINFORMATICS), 6741 LNCS, 482-496. SCOPUS. HTTPS://DOI.ORG/10.1007/978-3-642-21640-4_36
28. GBTEC. (N.D.). END-TO-END PROCESS MANAGEMENT: FROM ISOLATED DOCUMENTATION TO INTEGRATED PROCESS MODELS. HTTPS://WWW.GBTEC.COM/RESOURCES/END-TO-END-PROCESS/
29. GHOSE, ADITYA & KOLIADIS, GEORGE & CHUENG, ARTHUR. (2007). PROCESS DISCOVERY FROM MODEL AND TEXT ARTEFACTS. FACULTY OF INFORMATICS - PAPERS. 167-174. 10.1109/SERVICES.2007.52.

30. GREINER, L. E. (1998). EVOLUTION AND REVOLUTION AS ORGANIZATIONS GROW. *HARVARD BUSINESS REVIEW, 76*(3), 55-68. HTTPS://HBR.ORG/1998/05/EVOLUTION-AND-REVOLUTION-AS-ORGANIZATIONS-GROW
31. GROHS, M., ABB, L., ELSAYED, N., & REHSE, J.-R. (2024). LARGE LANGUAGE MODELS CAN ACCOMPLISH BUSINESS PROCESS MANAGEMENT TASKS. IN J. DE WEERDT & L. PUFAHL (EDS.), BUSINESS PROCESS MANAGEMENT WORKSHOPS (PP. 453-465). SPRINGER NATURE SWITZERLAND. HTTPS://DOI.ORG/10.1007/978-3-031-50974-2_34
32. HAN, Y., *ET AL*. (2024). *MAO: A FRAMEWORK FOR PROCESS MODEL GENERATION WITH MULTI-AGENT ORCHESTRATION*. (PREPRINT, ARXIV:2408.01916)
33. HERBST, J., & KARAGIANNIS, D. (1999). AN INDUCTIVE APPROACH TO THE ACQUISITION AND ADAPTATION OF WORKFLOW MODELS. HTTPS://WWW.SEMANTICSCHOLAR.ORG/PAPER/AN-INDUCTIVE-APPROACH-TO-THE-ACQUISITION-AND-OF-HERBST-KARAGIANNIS/03D2B48EF0585906618778295297 70DE93F30D51
34. INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, & INTERNATIONAL ELECTROTECHNICAL COMMISSION. (2023). *SYSTEMS AND SOFTWARE ENGINEERING—SYSTEMS AND SOFTWARE QUALITY REQUIREMENTS AND EVALUATION (SQUARE)—PRODUCT QUALITY MODEL* (ISO/IEC STANDARD NO. 25010:2023, 2ND ED.). HTTPS://WWW.ISO.ORG/STANDARD/78176.HTML
35. JI, Z., LEE, N., FRIESKE, R., YU, T., SU, D., XU, Y., ISHII, E., BANG, Y. J., MADOTTO, A., & FUNG, P. (2023). SURVEY OF HALLUCINATION IN NATURAL LANGUAGE GENERATION. ACM COMPUTING SURVEYS, 55(12), 1-38. HTTPS://DOI.ORG/10.1145/3571730
36. KLIEVTSOVA, N., BENZIN, J.-V., KAMPIK, T., MANGLER, J., & RINDERLE-MA, S. (2023). *CONVERSATIONAL PROCESS MODELING: CAN GENERATIVE AI EMPOWER DOMAIN EXPERTS IN CREATING AND REDESIGNING PROCESS MODELS?* ARXIV PREPRINT ARXIV:2304.11065
37. KLIEVTSOVA, N., BENZIN, J.-V., MANGLER, J., KAMPIK, T., & RINDERLE-MA, S. (2025). *PROCESS MODELER VS. CHATBOT: IS GENERATIVE AI TAKING OVER PROCESS MODELING?* IN A. DELGADO & T. SLAATS (EDS.), PROCESS MINING WORKSHOPS - ICPM 2024 (LECTURE NOTES IN BUSINESS INFORMATION PROCESSING, VOL. ???, PP. 637-649). SPRINGER, CHAM. HTTPS://DOI.ORG/10.1007/978-3-031-82225-4_47
38. KOURANI, H., BERTI, A., SCHUSTER, D., & AALST, W. M. P. VAN DER. (2024). PROMOAI: PROCESS MODELING WITH GENERATIVE AI. 9, 8708-8712. HTTPS://DOI.ORG/10.24963/IJCAI.2024/1014
39. KOURANI, H., & VAN ZELST, S. J. (2023). POWL: PARTIALLY ORDERED WORKFLOW LANGUAGE. BUSINESS PROCESS MANAGEMENT: 21ST INTERNATIONAL CONFERENCE, BPM 2023, UTRECHT, THE NETHERLANDS, SEPTEMBER 11-15, 2023, PROCEEDINGS, 92-108. HTTPS://DOI.ORG/10.1007/978-3-031-41620-0_6
40. KHURANA, D., KOLI, A., KHATTER, K., & SINGH, S. (2023). NATURAL LANGUAGE PROCESSING: STATE OF THE ART, CURRENT TRENDS AND CHALLENGES. MULTIMEDIA TOOLS AND APPLICATIONS, 82(3), 3713-3744. HTTPS://DOI.ORG/10.1007/S11042-022-13428-4
41. KÖPKE, J., SAFAN, A., & UNIVERSITY OF KLAGENFURT, DEPARTMENT OF INFORMATICS SYSTEMS. (2024). INTRODUCING THE BPMN-CHATBOT FOR EFFICIENT LLM-BASED PROCESS MODELING. IN *CEUR WORKSHOP PROCEEDINGS* [CONFERENCE-PROCEEDING]. HTTPS://CEUR-WS.ORG/VOL-3758/PAPER-15.PDF
42. LANGCHAIN. (N.D.). INTRODUCTION. LANGCHAIN DOCUMENTATION. RETRIEVED AUGUST 14, 2025, FROM HTTPS://PYTHON.LANGCHAIN.COM/DOCS/INTRODUCTION/
43. LEE, J. D., & SEE, K. A. (2004). TRUST IN AUTOMATION: DESIGNING FOR APPROPRIATE RELIANCE. *HUMAN FACTORS, 46*(1), 50-80. HTTPS://DOI.ORG/10.1518/HFES.46.1.50_30392
44. LIN, L., JIN, Y., ZHOU, Y., CHEN, W., & QIAN, C. (2024). MAO: A FRAMEWORK FOR PROCESS MODEL GENERATION WITH MULTI-AGENT ORCHESTRATION (NO. ARXIV:2408.01916). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2408.01916

45. LIN, S., HILTON, J., & EVANS, O. (2022). TRUTHFULQA: MEASURING HOW MODELS MIMIC HUMAN FALSEHOODS. IN S. MURESAN, P. NAKOV, & A. VILLAVICENCIO (EDS.), PROCEEDINGS OF THE 60TH ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (VOLUME 1: LONG PAPERS) (PP. 3214-3252). ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. HTTPS://DOI.ORG/10.18653/V1/2022.ACL-LONG.229
46. W, P., YUAN, W., FU, J., JIANG, Z., HAYASHI, H., & NEUBIG, G. (2023). PRE-TRAIN, PROMPT, AND PREDICT: A SYSTEMATIC SURVEY OF PROMPTING METHODS IN NATURAL LANGUAGE PROCESSING. ACM COMPUT. SURV., 55(9), 195:1-195:35. HTTPS://DOI.ORG/10.1145/3560815
47. MARTINEAU, K. (2024, NOVEMBER 13). WHAT IS RETRIEVAL-AUGMENTED GENERATION? IBM RESEARCH. HTTPS://RESEARCH.IBM.COM/BLOG/RETRIEVAL-AUGMENTED-GENERATION-RAG
48. MENDLING, JAN & DUMAS, MARLON & LA ROSA, MARCELLO & REIJERS, HAJO. (2019). STRUCTURING BUSINESS PROCESS MANAGEMENT: BRIDGING THE GAP BETWEEN INFORMATION SYSTEMS RESEARCH AND PRACTICE. 10.1007/978-3-030-06234-7_19.
49. MIRO. (N.D.). OUR MISSION. RETRIEVED APRIL 4, 2025, FROM HTTPS://MIRO.COM/ABOUT/
50. NAVVIA. (N.D.). *NAVVIA | PROCESS MODELING & OPERATIONAL RESILIENCE PLATFORM*. HTTPS://NAVVIA.COM/
51. NOTION LABS, INC. (N.D.). WHY WE BUILT NOTION. RETRIEVED APRIL 4, 2025, FROM HTTPS://WWW.NOTION.COM/ABOUT
52. NOUR ELDIN, A., ASSY, N., ANESINI, O., DALMAS, B., & GAALOUL, W. (2025). NALA2BPMN: AUTOMATING BPMN MODEL GENERATION WITH LARGE LANGUAGE MODELS. COOPERATIVE INFORMATION SYSTEMS, 398-404. HTTPS://DOI.ORG/10.1007/978-3-031-81375-7_27
53. OPENAI. (N.D.). PRICING | OPENAI API. RETRIEVED DECEMBER 22, 2025, FROM HTTPS://PLATFORM.OPENAI.COM/DOCS/PRICING
54. OPENAI. (2025A, APRIL 14). INTRODUCING GPT-4.1 IN THE API. OPENAI. RETRIEVED AUGUST 14, 2025, FROM HTTPS://OPENAI.COM/INDEX/GPT-4-1/
55. OPENAI. (2025B, AUGUST 7). INTRODUCING GPT-5 FOR DEVELOPERS. OPENAI. RETRIEVED AUGUST 14, 2025, FROM HTTPS://OPENAI.COM/INDEX/INTRODUCING-GPT-5-FOR-DEVELOPERS/
56. PARASURAMAN, R., & RILEY, V. (1997). HUMANS AND AUTOMATION: USE, MISUSE, DISUSE, ABUSE. *HUMAN FACTORS, 39*(2), 230-253. HTTPS://DOI.ORG/10.1518/001872097778543886
57. PEGA. (N.D.). *PEGA: BUILD FOR CHANGE*. HTTPS://WWW.PEGA.COM/
58. PETRONI, F., ROCKTÄSCHEL, T., RIEDEL, S., LEWIS, P., BAKHTIN, A., WU, Y., & MILLER, A. (2019). *LANGUAGE MODELS AS KNOWLEDGE BASES?* IN PROCEEDINGS OF THE 2019 CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING AND THE 9TH INTERNATIONAL JOINT CONFERENCE ON NATURAL LANGUAGE PROCESSING (EMNLP-IJCNLP) (PP. 2463-2473). ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. HTTPS://DOI.ORG/10.18653/V1/D19-1250
59. PIS (N.D.) *PROCESS INTELLIGENCE SOLUTIONS (P.I.S.)* HTTPS://PROCESSINTELLIGENCE.SOLUTIONS/PM4PY
60. POTTS, B. (2024, APRIL 2). GRAPHRAG: UNLOCKING LLM DISCOVERY ON NARRATIVE PRIVATE DATA - MICROSOFT RESEARCH. MICROSOFT RESEARCH. HTTPS://WWW.MICROSOFT.COM/EN-US/RESEARCH/BLOG/GRAPHRAG-UNLOCKING-LLM-DISCOVERY-ON-NARRATIVE-PRIVATE-DATA/
61. PROCESS DISCOVERY—PROCESS MINING. (N.D.). RETRIEVED FEBRUARY 26, 2025, FROM HTTPS://WWW.PROCESSMINING.ORG/PROCESS-DISCOVERY.HTML
62. PROCESSMAKER. (N.D.). *PROCESSMAKER | BUSINESS PROCESS AUTOMATION & WORKFLOW SOFTWARE*. HTTPS://WWW.PROCESSMAKER.COM/

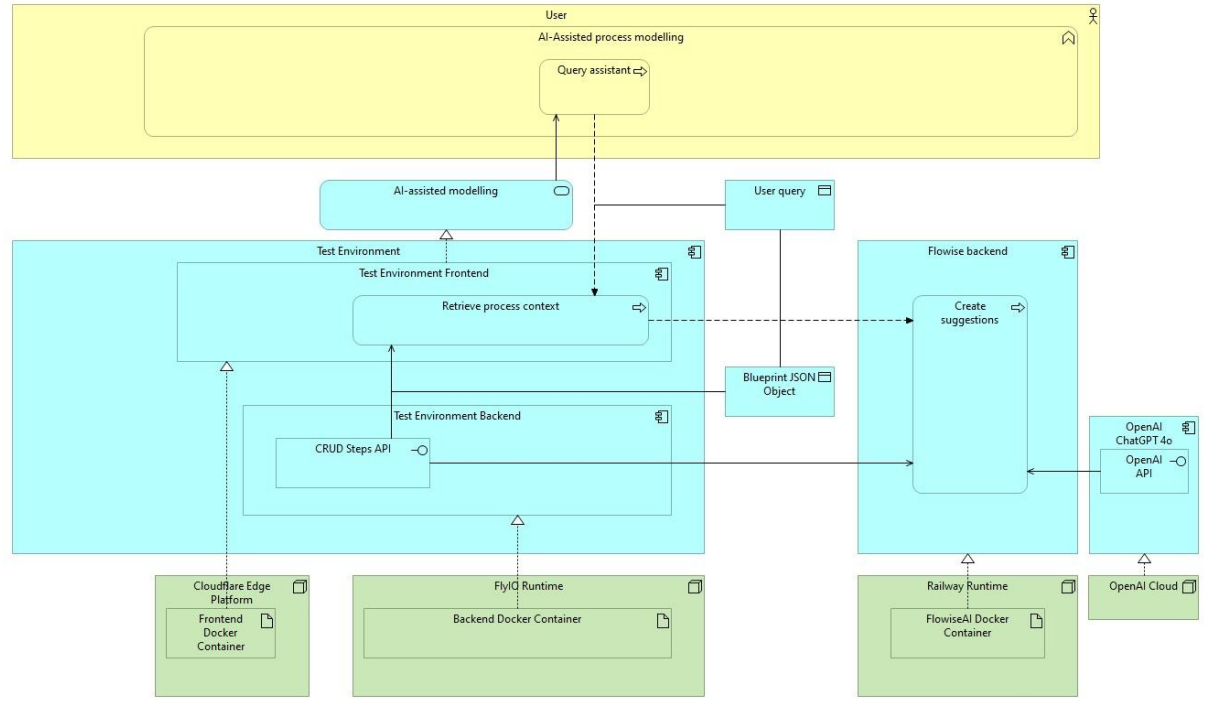
63. PROCESS MINING GROUP. (N.D.). *CONFORMANCE CHECKING*. EINDHOVEN UNIVERSITY OF TECHNOLOGY. RETRIEVED APRIL 2ND, FROM HTTPS://WWW.PROCESSMINING.ORG/CONFORMANCE.HTML
64. PROM TOOLS. (2024). *PROM 6.14* HTTPS://PROMTOOLS.ORG/PROM-6-14/
65. REIJERS, H. A., & MENDLING, J. (2011). A STUDY INTO THE FACTORS THAT INFLUENCE THE UNDERSTANDABILITY OF BUSINESS PROCESS MODELS. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS - PART A: SYSTEMS AND HUMANS, 41(3), 449-462. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS - PART A: SYSTEMS AND HUMANS. HTTPS://DOI.ORG/10.1109/TSMCA.2010.2087017
66. REIJERS, H. A. (2021). BUSINESS PROCESS MANAGEMENT: THE EVOLUTION OF A DISCIPLINE. COMPUTERS IN INDUSTRY, 126, 103404. HTTPS://DOI.ORG/10.1016/J.COMPIND.2021.103404
67. RETRIEVAL - OPENAI API. (N.D.). HTTPS://PLATFORM.OPENAI.COM/DOCS/GUIDES/RETRIEVAL
68. ROSEMANN, M. (2006). POTENTIAL PITFALLS OF PROCESS MODELING: PART A. *BUSINESS PROCESS MANAGEMENT JOURNAL, 12*(3), 377-384. HTTPS://DOI.ORG/10.1108/14637150610668024
69. SCHÜLER, S., & ALPERS, S. (2024). *STATE OF THE ART: AUTOMATIC GENERATION OF BUSINESS PROCESS MODELS*. IN J. DE WEERDT (ED.), BUSINESS PROCESS MANAGEMENT WORKSHOPS, BPM 2023 (LECTURE NOTES IN BUSINESS INFORMATION PROCESSING, VOL. 492, PP. 161-173). SPRINGER, CHAM. HTTPS://DOI.ORG/10.1007/978-3-031-50974-2_13
70. SONBOL, R., REBDAWI, G. & GHNEIM, N. A MACHINE TRANSLATION LIKE APPROACH TO GENERATE BUSINESS PROCESS MODEL FROM TEXTUAL DESCRIPTION. *SN COMPUT. SCI.* 4, 291 (2023). HTTPS://DOI-ORG.EZPROXY2.UTWENTE.NL/10.1007/S42979-023-01742-Z
71. SHUSTER, K., POFF, S., CHEN, M., KIELA, D., & WESTON, J. (2021). RETRIEVAL AUGMENTATION REDUCES HALLUCINATION IN CONVERSATION (NO. ARXIV:2104.07567). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2104.07567
72. SKOBTSOV, A., & KALENKOVA, A. (2019). *EFFICIENT COMPARISON OF PROCESS MODELS USING TABU SEARCH ALGORITHM*. IN PROC. OF MACSPRO 2019 - MODELING AND ANALYSIS OF COMPLEX SYSTEMS AND PROCESSES (PP. 51-61). CEUR WORKSHOP PROC., VOL. 2478
73. TAN, S. A. (2025, MAY 5). WHAT IS AGENTIC AI? MAKE. HTTPS://WWW.MAKE.COM/EN/BLOG/WHAT-IS-AGENTIC-AI
74. UIPATH. (N.D.). UIPATH AUTOMATION PLATFORM: DRIVE AI TRANSFORMATION WITH AGENTIC AUTOMATION | UIPATH. HTTPS://WWW.UIPATH.COM/
75. VAN DER AALST, W. M. P., ADRIANSYAH, A., ALVES DE MEDEIROS, A. K., ARCIERI, F., BAIER, T., BLICKLE, T., ... & VAN DER WERF, J. M. (2012). PROCESS MINING MANIFESTO. IN F. DANIEL, K. BARKAOUI, & S. DUSTDAR (EDS.), *BPM 2011 WORKSHOPS, PART I* (PP. 169-194). SPRINGER-VERLAG. HTTPS://WWW.TF-PM.ORG/RESOURCES/MANIFESTO
76. VAN DER AALST (2016). *PROCESS MINING : DATA SCIENCE IN ACTION* (SECOND EDITION). SPRINGER. HTTPS://DOI.ORG/10.1007/978-3-662-49851-4
77. VAN DER AALST (2011). *PROCESS MINING : DISCOVERY, CONFORMANCE AND ENHANCEMENT OF BUSINESS PROCESSES*. SPRINGER. HTTPS://DOI.ORG/10.1007/978-3-642-19345-3
78. VAN DER AALST, W., WEIJTERS, T., & MARUSTER, L. (2004). WORKFLOW MINING: DISCOVERING PROCESS MODELS FROM EVENT LOGS. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 16(9), 1128-1142. SCOPUS. HTTPS://DOI.ORG/10.1109/TKDE.2004.47
79. VATSAL, S., & DUBEY, H. (2024). A SURVEY OF PROMPT ENGINEERING METHODS IN LARGE LANGUAGE MODELS FOR DIFFERENT NLP TASKS (NO. ARXIV:2407.12994). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2407.12994
80. VENKATESH, V., MORRIS, M. G., DAVIS, G. B., & DAVIS, F. D. (2003). *USER ACCEPTANCE OF INFORMATION TECHNOLOGY: TOWARD A UNIFIED VIEW*. MIS QUARTERLY, 27(3), 425-478. HTTPS://DOI.ORG/10.2307/30036540

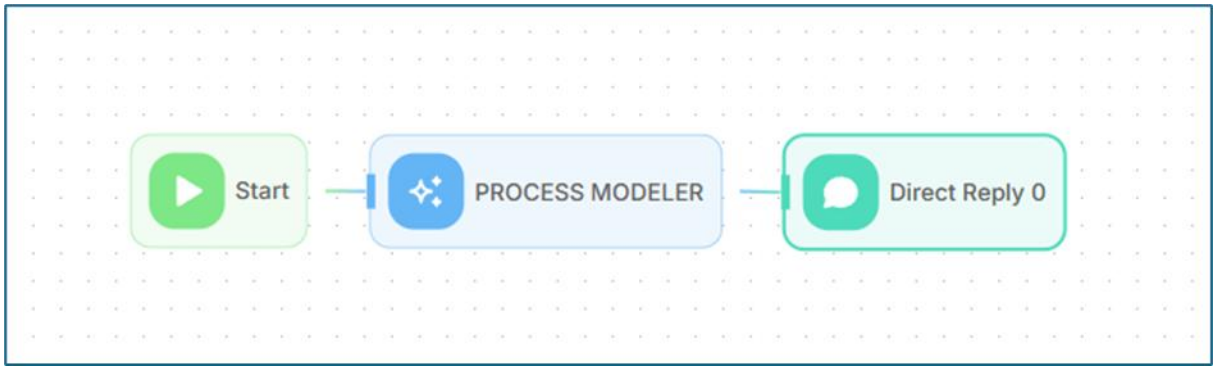
81. VIDGOF, M., BACHHOFNER, S., & MENDLING, J. (2023). LARGE LANGUAGE MODELS FOR BUSINESS PROCESS MANAGEMENT: OPPORTUNITIES AND CHALLENGES. IN C. DI FRANCESCOMARINO, A. BURATTIN, C. JANIESCH, & S. SADIQ (EDS.), BUSINESS PROCESS MANAGEMENT FORUM (PP. 107-123). SPRINGER NATURE SWITZERLAND. HTTPS://DOI.ORG/10.1007/978-3-031-41623-1_7
82. WASPADA, I., & SARNO, R. (2020). AN IMPROVED METHOD OF GRAPH EDIT DISTANCE FOR BUSINESS PROCESS MODEL SIMILARITY MEASUREMENT (P. 6). HTTPS://DOI.ORG/10.1109/ICICOS51170.2020.9299037
83. WEI, J., WANG, X., SCHUURMANS, D., BOSMA, M., ICHTER, B., XIA, F., CHI, E., LE, Q., & ZHOU, D. (2023). CHAIN-OF-THOUGHT PROMPTING ELICITS REASONING IN LARGE LANGUAGE MODELS (NO. ARXIV:2201.11903). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2201.11903
84. WESKE, M. (2024). INTRODUCTION. IN M. WESKE (ED.), BUSINESS PROCESS MANAGEMENT: CONCEPTS, LANGUAGES, ARCHITECTURES (PP. 3-23). SPRINGER. HTTPS://DOI.ORG/10.1007/978-3-662-69518-0_1
85. WIERINGA, R. J. (2014). *DESIGN SCIENCE METHODOLOGY FOR INFORMATION SYSTEMS AND SOFTWARE ENGINEERING*. SPRINGER. HTTPS://DOI.ORG/10.1007/978-3-662-43839-8
86. WIERINGA, R. J., & HEERKENS, J. M. G. (2007). *DESIGNING REQUIREMENTS ENGINEERING RESEARCH*. IN PROC. OF THE 5TH INTERNATIONAL WORKSHOP ON COMPARATIVE EVALUATION IN REQUIREMENTS ENGINEERING (CERE 2007) (PP. 3-7). ENSCHEDE, NL: UTWENTE/CTIT
87. WIŚNIEWSKI, P., KLUZA, K., JOBCZYK, K., STACHURA-TERLECKA, B., & LIGĘZA, A. (2019). *OVERVIEW OF GENERATION METHODS FOR BUSINESS PROCESS MODELS*. IN C. DOULIGERIS, D. KARAGIANNIS, & D. APOSTOLOU (EDS.), KNOWLEDGE SCIENCE, ENGINEERING AND MANAGEMENT - KSEM 2019 (LECTURE NOTES IN COMPUTER SCIENCE, VOL. 11776, PP. 55-60). SPRINGER, CHAM. HTTPS://DOI.ORG/10.1007/978-3-030-29563-9_6
88. IBM (2021, OCTOBER 15) *WHAT IS PROCESS MINING? | IBM*. HTTPS://WWW.IBM.COM/THINK/TOPICS/PROCESS-MINING
89. W&B. (2025, MAY 13). WEIGHTS & BIASES. W&B. HTTPS://WANDB.AI/BYYOUNG3/GENERATIVE-AI/REPORTS/GRAPHRAG-ENHANCING-LLMS-WITH-KNOWLEDGE-GRAPHS-FOR-SUPERIOR-RETRIEVAL--VMLLDZOXMDY0OTU0MA
90. XU, B., YANG, A., LIN, J., WANG, Q., ZHOU, C., ZHANG, Y., & MAO, Z. (2025). *EXPERTPROMPTING: INSTRUCTING LARGE LANGUAGE MODELS TO BE DISTINGUISHED EXPERTS*. ARXIV PREPRINT ARXIV:2305.14688
91. ZHOU, D., SCHÄRLI, N., HOU, L., WEI, J., SCALES, N., WANG, X., SCHUURMANS, D., CUI, C., BOUSQUET, O., LE, Q., & CHI, E. (2023). LEAST-TO-MOST PROMPTING ENABLES COMPLEX REASONING IN LARGE LANGUAGE MODELS (NO. ARXIV:2205.10625). ARXIV. HTTPS://DOI.ORG/10.48550/ARXIV.2205.10625

APPENDIX A: BLUEPRINT JSON FORMAT EXAMPLE

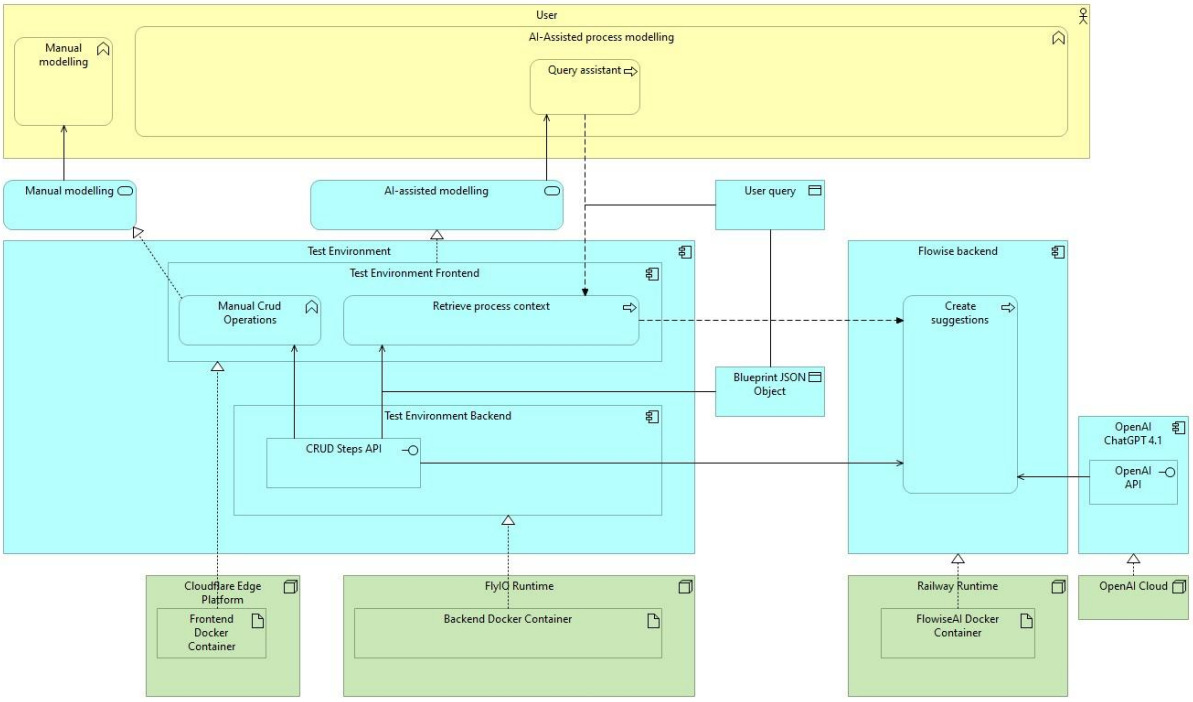
```
{
  "elements": {
    "byId": {
      "100": {
        "id": 100,
        "type": "BPProcess",
        "name": "Handle order",
        "parent": null,
        "children": [],
        "input_data": [200],
        "output_data": [],
        "apps": [300],
        "roles": [400]
      },
      "200": {
        "id": 200,
        "type": "BPData",
        "name": "Order form",
        "parent": null
      },
      "300": {
        "id": 300,
        "type": "BPApp",
        "name": "Order management system",
        "parent": null
      },
      "400": {
        "id": 400,
        "type": "BPRole",
        "name": "Order handler",
        "parent": null
      }
    }
  }
}
```

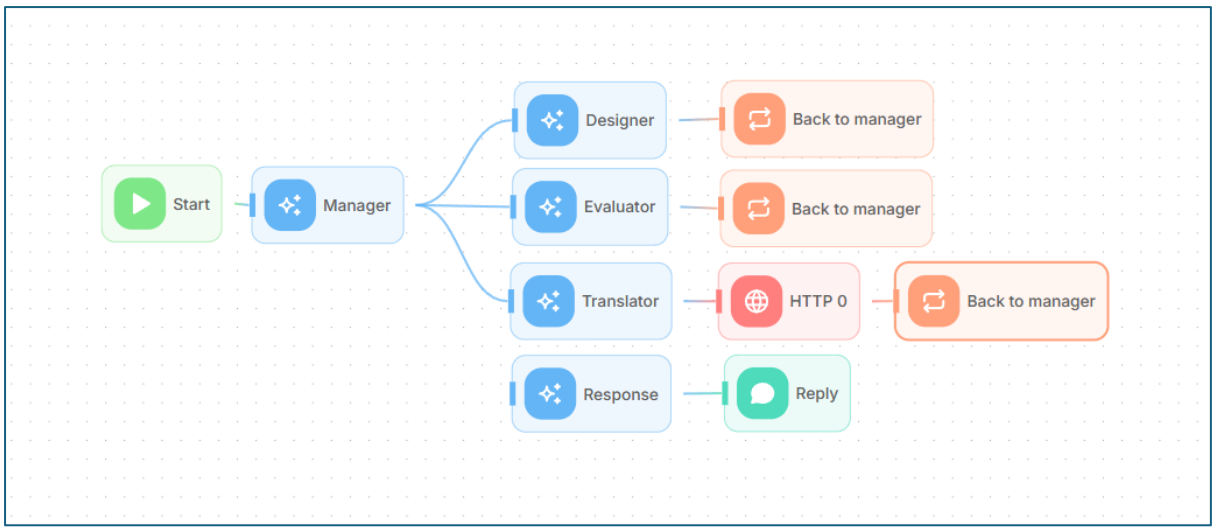
APPENDIX B: PROCESS ASSISTANT V1



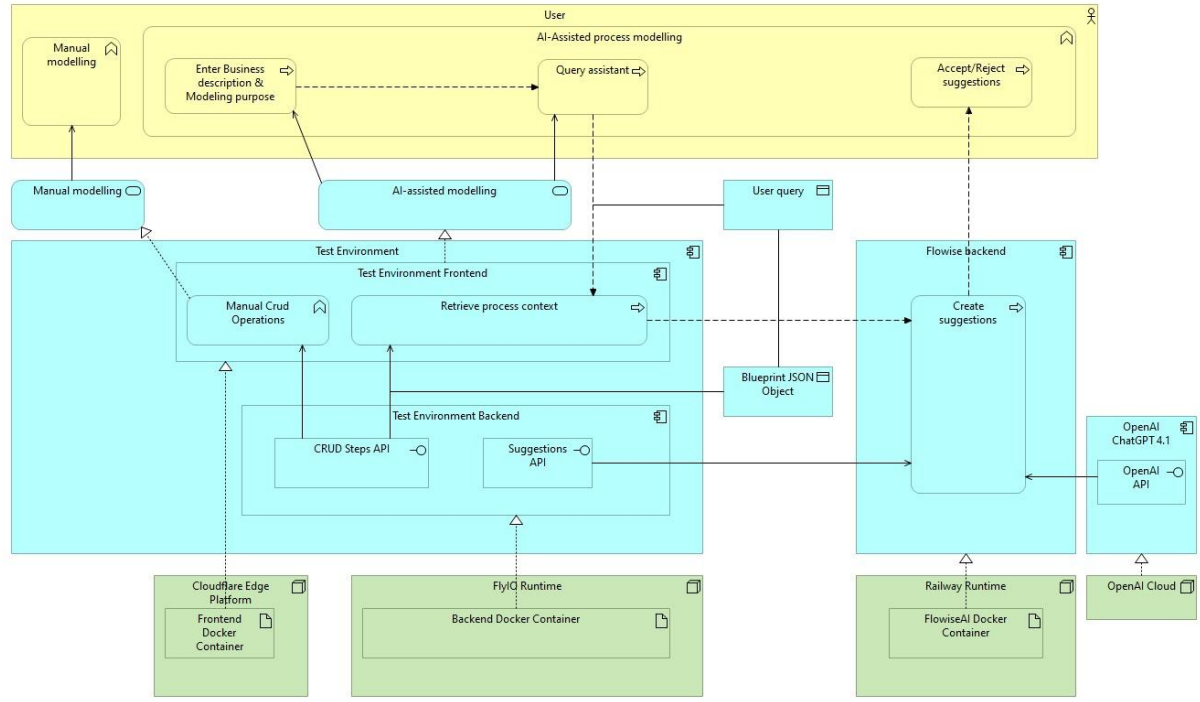


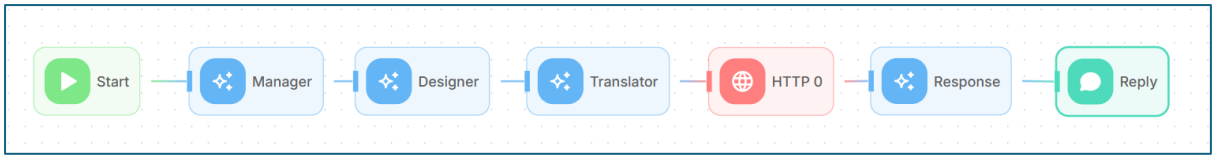
APPENDIX C: PROCESS ASSISTANT V2



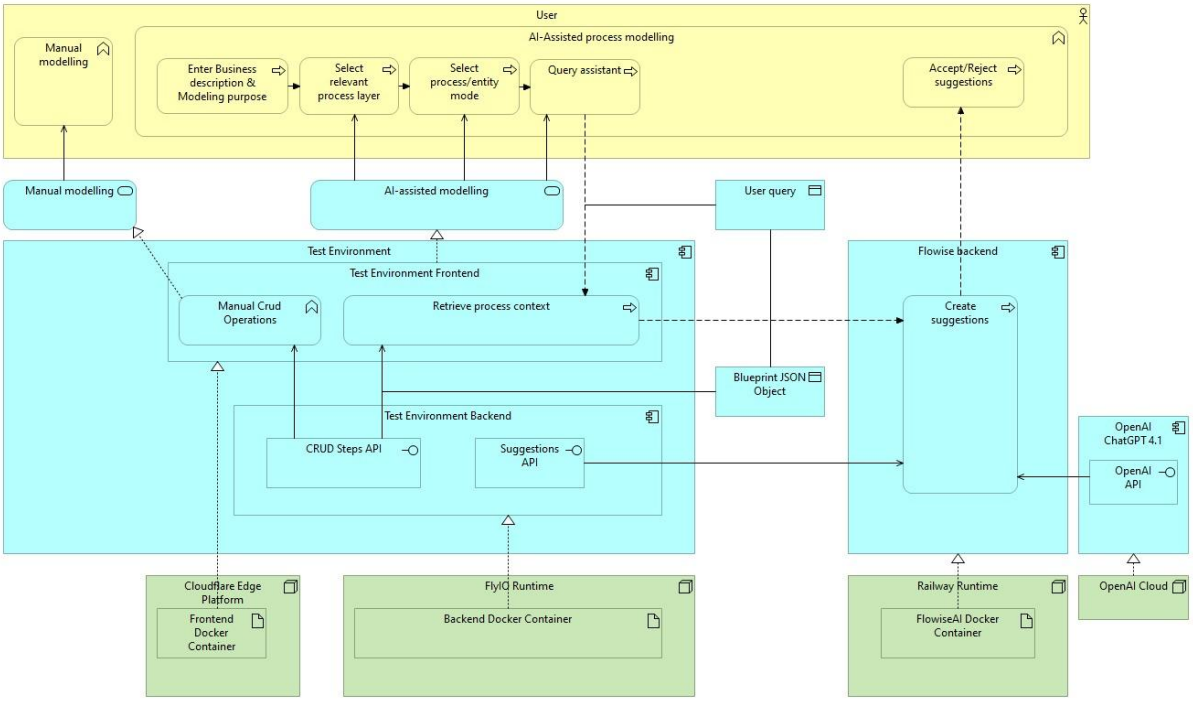


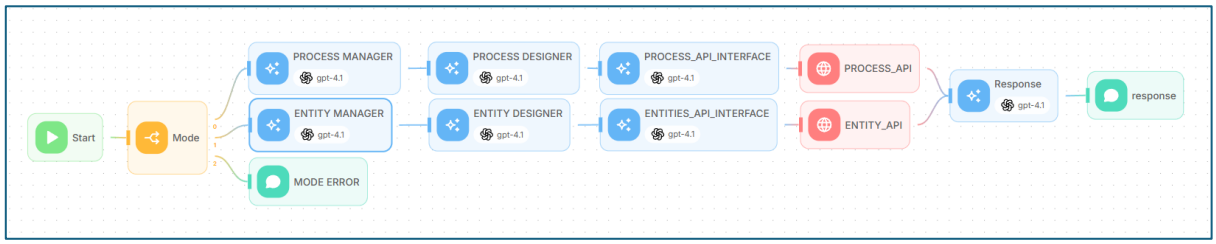
APPENDIX D: PROCESS ASSISTANT V3





APPENDIX E: PROCESS ASSISTANT V4





APPENDIX F: EXPERT-RUBRIC CORRELATION TABLE

Variable	Rubric Dimension	Correlation (ρ)	p-value	Interpretation
Group (Manual vs AI)	Clarity	-0.188	0.403	Negligible
	Usability	-0.119	0.599	Negligible
	Correctness	0.045	0.841	Negligible
	Completeness	-0.022	0.921	Negligible
	Specificity	0.015	0.948	Negligible
Supermarket Experience	Clarity	-0.317	0.151	Moderate negative
	Usability	-0.287	0.195	Moderate negative
	Correctness	-0.213	0.342	Negligible
	Specificity	-0.078	0.730	Negligible
	Completeness	0.009	0.969	Negligible
Academic Experience	Correctness	-0.293	0.186	Moderate negative
	Clarity	0.229	0.306	Moderate positive
	Completeness	0.087	0.699	Negligible
	Usability	-0.009	0.969	Negligible
	Specificity	0.017	0.939	Negligible
Process Modelling Experience	Usability	0.196	0.382	Weak positive
	Correctness	0.149	0.509	Negligible
	Clarity	-0.050	0.827	Negligible
	Specificity	-0.021	0.925	Negligible
	Completeness	-0.083	0.715	Negligible
AI Experience	Clarity	0.264	0.235	Weak positive
	Usability	0.106	0.639	Negligible
	Correctness	0.092	0.685	Negligible
	Specificity	0.027	0.905	Negligible
	Completeness	-0.020	0.929	Negligible
Age	Specificity	-0.172	0.445	Weak negative
	Completeness	-0.169	0.451	Weak negative
	Correctness	-0.066	0.770	Negligible
	Clarity	0.047	0.837	Negligible
	Usability	-0.012	0.958	Negligible
Total Time Spent	Completeness	0.433	0.044	More time → higher completeness
	Specificity	0.320	0.147	Moderate positive
	Usability	0.150	0.507	Negligible
	Clarity	0.101	0.654	Negligible
	Correctness	-0.156	0.487	Negligible
Total Tokens (manual group excluded)	Specificity	0.201	0.370	Weak positive
	Completeness	0.163	0.468	Negligible
	Correctness	0.037	0.870	Negligible
	Clarity	-0.140	0.534	Negligible
	Usability	0.003	0.990	Negligible
Total AI Calls (manual group excluded)	Specificity	0.190	0.397	Weak positive
	Completeness	0.150	0.505	Negligible
	Correctness	0.010	0.964	Negligible
	Clarity	-0.139	0.538	Negligible
	Usability	-0.011	0.960	Negligible

APPENDIX G: PERCEIVED QUALITY CORRELATION TABLE

Prior Metric	Question	Correlation	p-value
Age	Q1: Correct Models	0.215	0.337
	Q6: Sufficient Knowledge	0.186	0.408
	Q3: Efficiency	0.077	0.734
	Q4: Ease of Use	-0.159	0.481
	Q2: Complete Models	0.050	0.826
Academic Experience_encoded	Q6: Sufficient Knowledge	0.376	0.085
	Q3: Efficiency	0.267	0.230
	Q1: Correct Models	0.237	0.289
	Q4: Ease of Use	0.139	0.536
	Q2: Complete Models	0.122	0.590
AI Experience_encoded	Q4: Ease of Use	-0.225	0.313
	Q6: Sufficient Knowledge	-0.087	0.701
	Q1: Correct Models	-0.055	0.808
	Q3: Efficiency	0.005	0.981
	Q2: Complete Models	0.005	0.984
Group_encoded	Q4: Ease of Use	0.772	<0.001
	Q3: Efficiency	0.508	0.016
	Q6: Sufficient Knowledge	0.238	0.285
	Q1: Correct Models	0.218	0.330
	Q2: Complete Models	0.104	0.646
Process Modeling Exp._encoded	Q1: Correct Models	-0.283	0.203
	Q4: Ease of Use	0.251	0.260
	Q3: Efficiency	-0.178	0.429
	Q6: Sufficient Knowledge	-0.087	0.701
	Q2: Complete Models	0.007	0.975
Supermarket Exp._encoded	Q4: Ease of Use	0.226	0.311
	Q2: Complete Models	-0.208	0.352
	Q1: Correct Models	-0.104	0.645
	Q6: Sufficient Knowledge	-0.070	0.757
	Q3: Efficiency	-0.009	0.969
Total AI Calls	Q4: Ease of Use	0.742	<0.001
	Q3: Efficiency	0.431	0.045
	Q1: Correct Models	0.043	0.848
	Q6: Sufficient Knowledge	0.042	0.854
	Q2: Complete Models	0.020	0.931
Total Time Spent (s)	Q2: Complete Models	-0.372	0.088
	Q3: Efficiency	-0.184	0.413
	Q1: Correct Models	-0.183	0.415
	Q6: Sufficient Knowledge	-0.161	0.474
	Q4: Ease of Use	-0.123	0.585
Total Tokens	Q4: Ease of Use	0.767	<0.001
	Q3: Efficiency	0.463	0.030
	Q1: Correct Models	0.097	0.667
	Q2: Complete Models	0.059	0.793
	Q6: Sufficient Knowledge	0.029	0.897

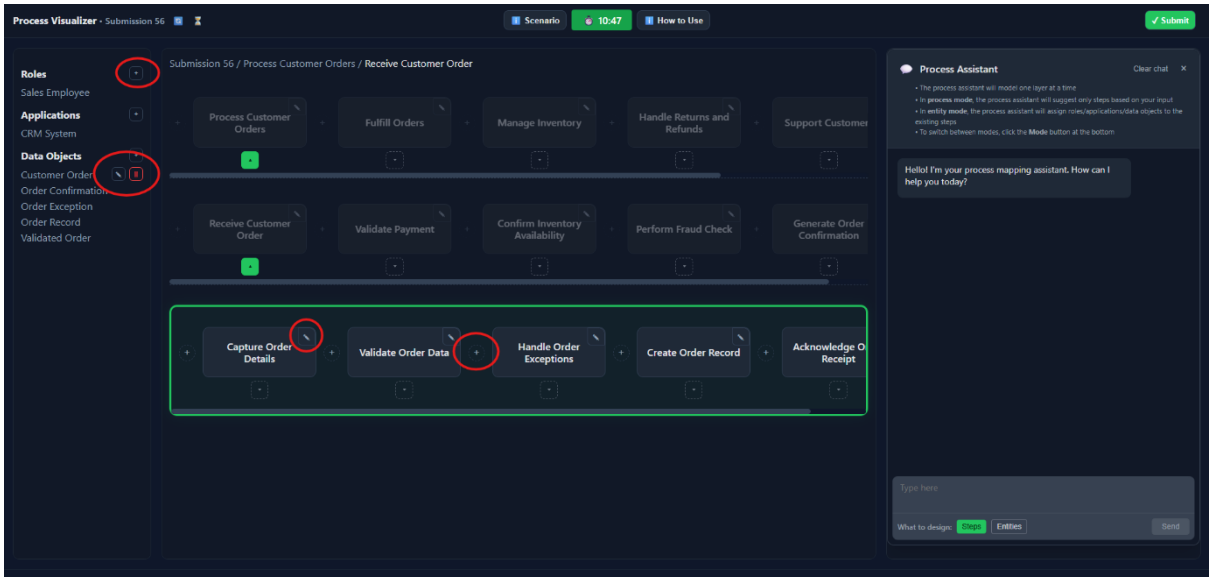
APPENDIX H: ASSISTANT QUALITY CORRELATION TABLE

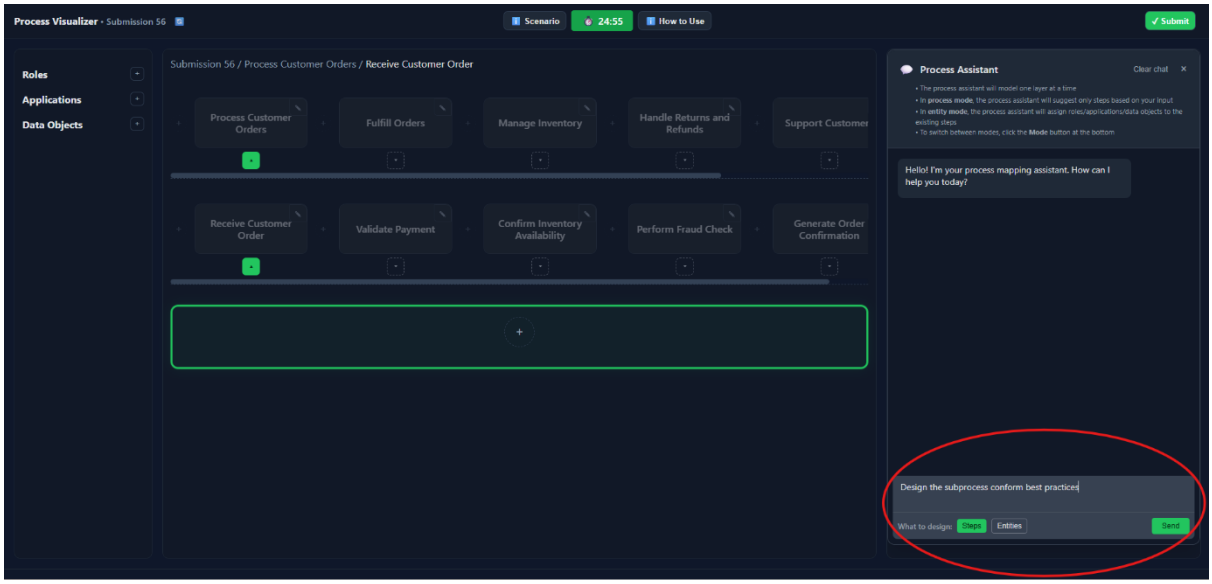
Prior Metric	Question	Correlation	p-value
Age	Q10: Intent to Use	-0.338	0.258
	Q9: Clear Motivations	-0.326	0.277
	Q11: AI Guidance	-0.233	0.444
	Q7: Clear Suggestions	-0.217	0.476
	Q8: Relevant Suggestions	-0.017	0.957
	Q5: Social Approval	0.005	0.988
Academic Experience_encoded	Q8: Relevant Suggestions	0.406	0.169
	Q5: Social Approval	-0.375	0.207
	Q9: Clear Motivations	-0.302	0.316
	Q7: Clear Suggestions	0.212	0.486
	Q11: AI Guidance	-0.176	0.565
	Q10: Intent to Use	0.091	0.768
Process Modeling Exp._encoded	Q7: Clear Suggestions	-0.562	0.046
	Q11: AI Guidance	-0.302	0.316
	Q8: Relevant Suggestions	-0.295	0.327
	Q5: Social Approval	-0.154	0.615
	Q10: Intent to Use	-0.149	0.626
	Q9: Clear Motivations	-0.124	0.686
Supermarket Exp._encoded	Q7: Clear Suggestions	0.308	0.305
	Q10: Intent to Use	0.260	0.391
	Q11: AI Guidance	0.229	0.451
	Q5: Social Approval	0.220	0.471
	Q9: Clear Motivations	-0.142	0.644
	Q8: Relevant Suggestions	0.073	0.812
Total AI Calls	Q8: Relevant Suggestions	0.434	0.139
	Q9: Clear Motivations	-0.210	0.492
	Q11: AI Guidance	0.147	0.632
	Q7: Clear Suggestions	0.122	0.691
	Q10: Intent to Use	0.122	0.692
	Q5: Social Approval	-0.100	0.745
Total Time Spent (s)	Q8: Relevant Suggestions	0.202	0.509
	Q9: Clear Motivations	-0.172	0.575
	Q7: Clear Suggestions	0.120	0.696
	Q10: Intent to Use	0.067	0.828
	Q11: AI Guidance	-0.059	0.847
	Q5: Social Approval	-0.018	0.953
Total Tokens	Q8: Relevant Suggestions	0.346	0.246
	Q11: AI Guidance	0.243	0.423
	Q10: Intent to Use	0.172	0.574
	Q9: Clear Motivations	-0.221	0.467
	Q5: Social Approval	-0.084	0.784
	Q7: Clear Suggestions	0.029	0.924

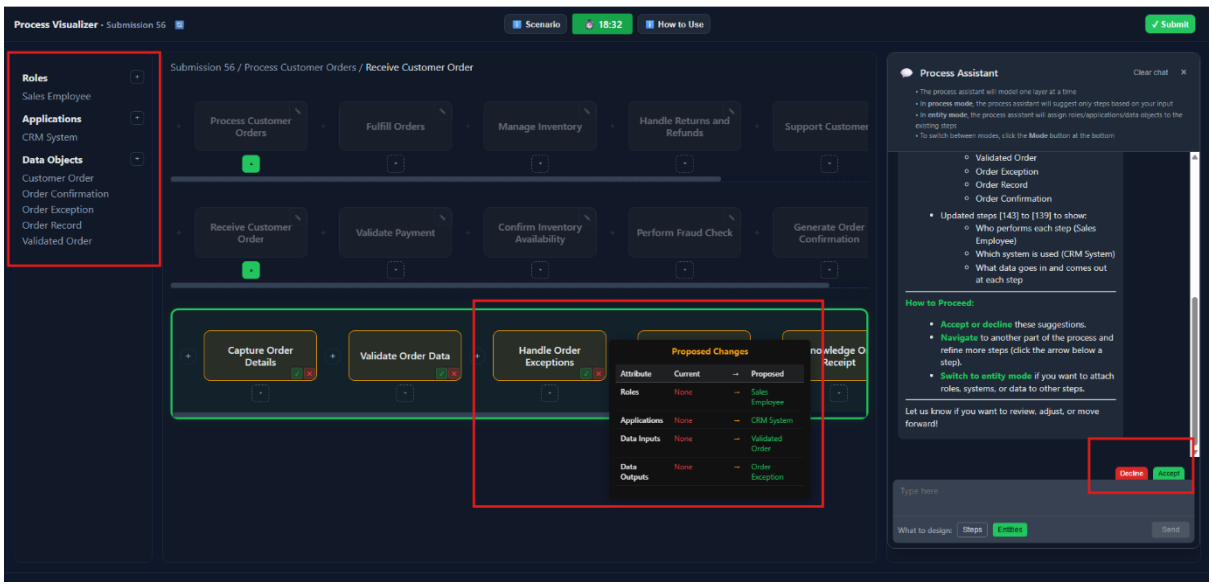
APPENDIX I: DEMOGRAPHIC CORRELATION TABLE

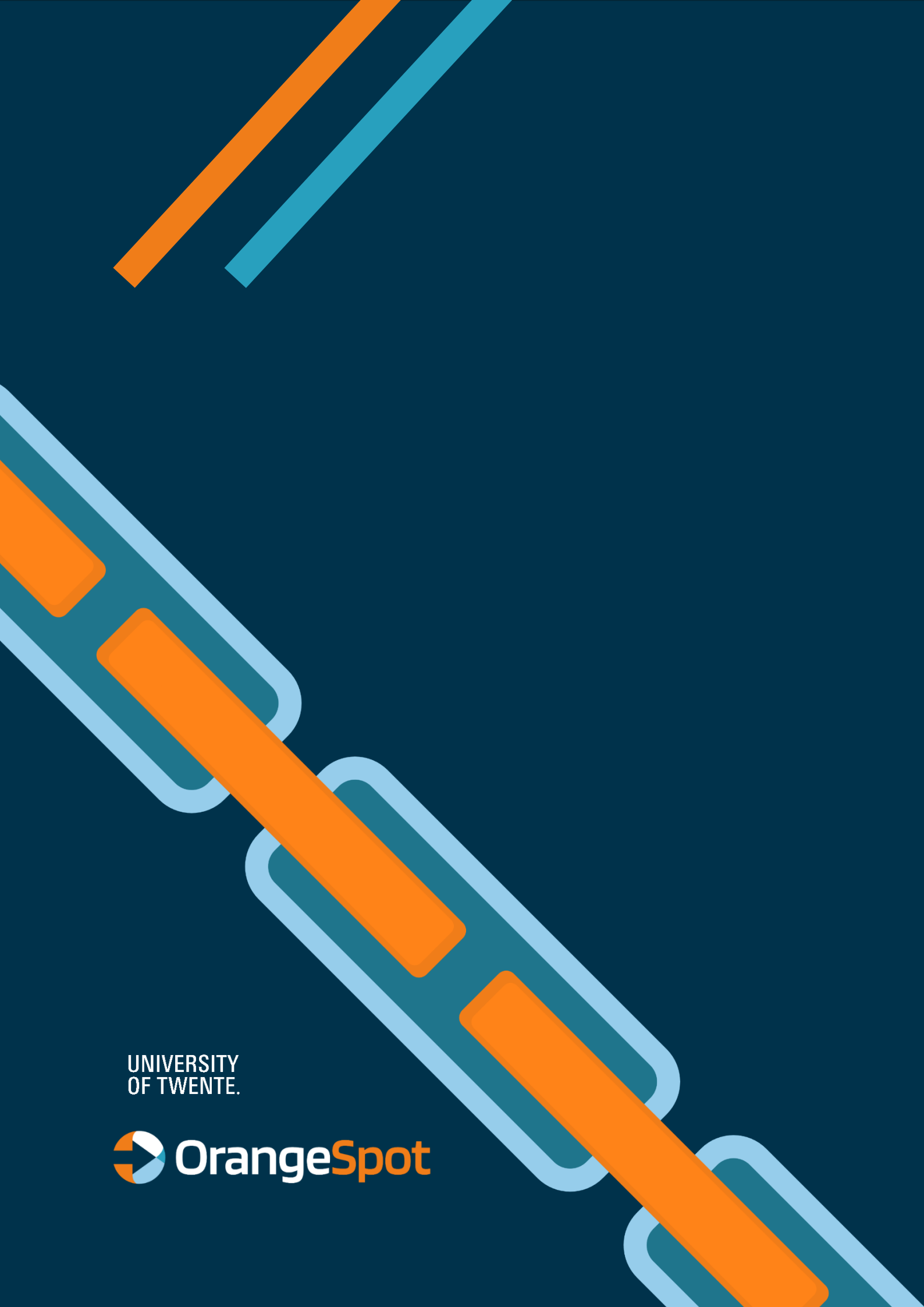
Metric 1	Metric 2	Correlation (ρ)	p-value
Educational level	Age	**0.508**	0.0158
Educational level	Process Modeling Experience	**-0.457**	0.0323
Group	AI Experience	-0.380	0.0814
Supermarket Experience	Age	-0.336	0.1265
AI Experience	Process Modeling Experience	0.265	0.2332
Group	Process Modeling Experience	0.255	0.2528
Group	Supermarket Experience	0.231	0.3018
Educational level	Supermarket Experience	-0.224	0.3173
Educational level	AI Experience	-0.223	0.3192
Educational level	Group	0.211	0.3469
Process Modeling Experience	Age	-0.204	0.3631
AI Experience	Supermarket Experience	-0.171	0.4459
Supermarket Experience	Process Modeling Experience	0.112	0.6197
Group	Age	-0.051	0.8204
AI Experience	Age	0.042	0.8541
Metric 1	Metric 2	Correlation (ρ)	p-value

APPENDIX J: TEST ENVIRONMENT INTERFACE








UNIVERSITY
OF TWENTE.
OrangeSpot