How might an Agentic AI-driven project management tool support Business Analysts in their work

Master Thesis
Strategic Product Design,
Delft University of Technology

Cas Gratama

Author
Cas J.W. Gratama
Student Number: 4884833

Submitted in partial fulfilment of the requirements for the degree of:

Master of Science
in Strategic Product Design
Faculty of Industrial Design Engineering

At Delft University of Technology, to be defended publicly on 2nd of April, 2026

Thesis Committee:
Chair: **Giulia Calabretta**
Mentor: **Jiwon Jung**

Company Accenture Song:
First mentor: **Najla Barikzai**

Abstract

Consulting firms such as Accenture face growing pressure to deliver complex digital projects more efficiently, and at the same time Agile ways of working and composable commerce architectures increase coordination complexity and place significant pressure on BAs. In this context, Agentic AI offers potential as a more proactive and embedded form of workflow support. This thesis explores how an Agentic AI-driven project management tool might support Accenture Song BAs in increasing their work efficiency during the Design & Build phases of Agile composable commerce delivery projects.

Using the Double Diamond framework, the study applies a mixed-method approach combining literature review, unstructured and semi-structured interviews, observations, survey data, and internal documentation analysis. This research shows that BA work in this context is characterised by fragmented information, high coordination complexity, frequent replanning, and a strong dependence on system understanding and cross-team alignment. From the identified improvement opportunities, dependency management emerged as the most relevant focus area, as it was both highly desired by BAs and highly consequential for delivery progress.

To address this opportunity, two concept directions were explored and prototyped, after which Risko was selected as the primary concept. Risko is an Agentic AI-driven prototype that supports BAs in maintaining dependency and future sprint planning-status awareness by analysing project data, surfacing potential risks and opportunities, and guiding more targeted check-ins and planning adjustments. Built in n8n using mock project data and stand-ins for tools such as Jira, Confluence, and Excel, the prototype demonstrates how Agentic AI capabilities such as data retrieval, tool use, and text generation can be translated into concrete workflow support while retaining human-in-the-loop control.

The project concludes that Agentic AI has clear potential to support BAs not by replacing their work, but by augmenting it through more proactive, context-aware assistance. At the same time, the thesis shows that such support must be carefully designed around context provision, low hallucination risk, deterministic logic where appropriate, and seamless integration into existing workflows. Although Risko appears promising in terms of desirability, feasibility, and viability, further validation is still needed before it can be considered a sufficiently validated solution for Accenture.

Abbreviations

AI - Artificial Intelligence
API - Application Programming Interface
ART - Agile release train
ASC - Accenture Song Commerce
BA - Business analyst
BR - Business requirement
FR - Functional requirement
GenAI - Generative Artificial Intelligence
HITL - Human in the loop
LLM - Large Language Model
MACH - Microservices, API-first, Cloud-native, Headless
MAS - Multi-agent system
MRQ - Main research question
NFR - Non-functional requirement
PBI - Product Backlog Item
PI - Product Increment
Pn - Participant (*n* = number)
PO - Product Owner
QA - Quality and Assurance
RAG - Retrieval-Augmented Generation
Rn - Requirement (*n* = number)
SAFe - Scaled Agile Framework
SBI - Sprint Backlog Item
SPD - Strategic Product Design
SQn - Sub-question (*n* = number)
TR - Technical requirement
UI - User interface
UX - User experience
VP - Value proposition

Table of Contents

1. Introduction

1.1 Project client: Accenture Song Commerce

This graduation project is conducted in collaboration with Accenture, a global consulting services company that supports organizations with strategy, consulting, technology, and operations to drive business transformation (Accenture, n.d.). Within Accenture, Accenture Song is the business unit focused on the user-facing technology of Accenture's clients, meaning anything that the employees, customers or partners of the clients of Accenture use. The Commerce practice within Accenture Song specializes in helping large, complex organisations that operate at scale (across multiple markets/countries, brands, product lines, operating companies, and stakeholders) whose commercial performance depends on the quality, reliability, and continuous evolution of their commerce capabilities.

In this project, Accenture Song Commerce (ASC) acts as the project client and main stakeholder, providing relevant real-world project context insights. The team supports the project by aligning the scope with practical consulting needs and constraints, offering insights, and validating (interim) outputs (such as: Assumptions, claims, concepts, and prototypes) for relevance and applicability, to ensure the final proposed solution is as usable and credible within its intended environment as possible.

Figure 1.1: Accenture Song logo.

1.2 Project context

The Dutch management consulting market size is about €5.5 billion in 2026, and it is expected to increase by 4.12% each year from 2026 to 2031 (Mordor Intelligence, 2026), which shows that businesses continue to need help with strategy, operations, and digital technology. However, this growth faces challenges. The number of workers for every retiree worldwide is predicted to drop from 8:1 to 4:1 by 2050 (World Economic Forum, 2017). At the same time, productivity growth has slowed significantly in the Netherlands, increasing by just 0.2% per year on average over the last ten years (CBS, 2025). These problems mean that Dutch consulting firms such as Accenture need to find new, more efficient ways to do their work.

In many contexts, delivering the same output in less time can appear to create a risk of revenue cannibalisation. However, efficiency gains do not necessarily translate into lower revenues. Jevons paradox (Jevons, 1865) suggests that when efficiency improvements reduce the cost of producing output, demand can increase as the service becomes more attractive and accessible, potentially increasing demand for the service. Moreover, in a competitive consulting market, not pursuing productivity improvements carries a strategic risk: If Accenture cannot deliver outcomes more efficiently, competitors that do will be able to undercut pricing and capture market share, with more severe long-term implications than internal cannibalisation. This makes the topic strategically important for Accenture.

Agentic AI, with capabilities such as advanced reasoning and tool-use (McKinsey & Company, 2025), is a promising technology for consultant efficiency support in the management of complex projects. By for example retrieving, analysing, and consolidating information that is currently fragmented across multiple systems and tools, Agentic AI could reduce the time consultants spend on administrative and routine work. This, in turn, may improve overall work efficiency by freeing capacity for tasks that rely more strongly on strategic or creative thinking (Radwan et al., 2024). Moreover, looking into how Accenture's consultants could apply Agentic AI technology in their own workflows aligns with Accenture's strategy as it claims it has hired over 77.000 AI professionals, positions itself as leader in AI-business implementation (Accenture, n.d.), and well-designed human-AI workflows are considered the most significant success factor of AI implementation (McKinsey & Company, 2025). By improving its own workflows with Agentic AI, Accenture can strengthen its credibility as an AI transformation partner, demonstrate practical expertise, and lead clients by example through its own application of the technology.

1.3 Project scope and goal

The project is scoped down to the 'Design & Build' phases (Global, 2025) of 'composable commerce platform delivery' projects, that are managed using 'Agile', as preliminary client interviews suggest firstly that many current ASC projects involve composable commerce platform delivery, enabling opportunities for observation and testing. Its prevalence also indicates client demand for these services, making the topic relevant from a business perspective for ASC. Secondly, Agile is the way-of-working used in these projects, and thirdly the Design & Build phases contain a high concentration of day-to-day project management inefficiencies. 'Agile' and 'composable commerce platform delivery' are further explained in chapters 2.1.3 and 2.1.4 of this report.

The project further focuses on the BA role and not other roles because activities and responsibilities might vary too much, and this project's timeframe is limited to 6 months. Moreover, preliminary interviews indicated that BAs are directly involved in the Design & Build phases of these composable commerce delivery projects, and they are also often in office, which enables observation and validation.

The main research question (**MRQ**) of this project is therefore:
"How might an Agentic AI-driven project management tool support Accenture Song Business Analysts in increasing their work efficiency during the Design & Build phases of Agile composable commerce delivery projects?".

The goal is to **design and validate an Agentic AI-driven project management tool, delivered as a working prototype, that enables AS BAs to work more efficiently during Design & Build phases of composable commerce delivery projects that are managed using Agile ways of working**.

In this project, efficiency is interpreted broadly and may include: Increasing output quality, quantity, and speed; decreasing costs and errors; reducing product delivery delay time; and decreasing cognitive load and stress among BAs. The prototype therefore targets not only measurable delivery outputs, but also the day-to-day experience of BAs (such as feeling more "on top of tasks" and less stressed).

1.4 Project approach

To answer the MRQ, this project is structured using the Double Diamond framework (Design Council, n.d.). The Double Diamond is considered a good fit because it allows the researcher to build a strong understanding of the project context, including Accenture and consulting practices, Agile, composable commerce platform architecture, and Agentic AI. The framework explicitly supports an initial divergence phase to explore and build understanding of the complex problem space, followed by convergence to define a focused opportunity. This progression reduces the risk of prematurely developing a solution before the underlying workflow challenges, domain constraints, and relevant AI capabilities are sufficiently understood. The research is set up in 4 phases:

Phase 1 - Discover (context exploration)
The aim of this phase is to build BA empathy and establish a foundational understanding of Agentic AI, to enable the identification of potential efficiency improvement opportunities in the BA workflow. This is guided by three sub-questions:
SQ1: What activities and responsibilities do BAs have, and which tools and deliverables are involved in their work during composable commerce delivery projects?
SQ2: What is Agentic AI, how does it work, and what are its capabilities?
These insights are then combined to explore:
SQ3: What improvement opportunities might exist when looking at BA activities and Agentic AI's capabilities?

Phase 2 - Define (opportunity selection)
During the Define phase, the broad set of opportunities is narrowed down into a single improvement opportunity to guide solution development. This is done by answering:
SQ4: What 3 improvement opportunities have the highest BA desirability?
SQ5: What are the challenges and impacts on project success of the 3 improvement opportunities?
SQ6: What is the #1 improvement opportunity?

SQ7: What objectives, requirements, and design principles can be derived to guide the solution?
This phase produces a design brief with an improvement opportunity that the solution will address and acts as the starting point of phase 3.

Phase 3 - Develop (solution creation)
In the third phase, solution ideas are generated, evaluated, and translated into a working prototype. Rapid prototyping is used as an additional guiding approach within this phase of the project, to accelerate learning through making and testing. Rapid prototyping supports fast iterations on interaction concepts, Agentic AI behaviour, and workflow integration choices, enabling early assumptions to be surfaced and refined before committing to a final prototype direction. This phase is guided by the questions:
SQ8: How might Agentic AI mitigate the elicited challenge(s) and improve BA efficiency?
SQ9: What potential solution(s) is considered most desirable, viable and feasible?

Phase 4 - Deliver (validation and future direction)
In the final phase, the prioritized prototype is finalized and assessed in terms of its contribution to BA efficiency, as well as its broader implications and future evolution. This is addressed through answering the questions:
SQ10: How could this tool be improved, and where do additional opportunities exist for AI implementation to further enhance the tool?
SQ11: How could the tool be deployed and implemented by ASC?
SQ12: What effect does this tool (in its future form) have on Accenture and the BA role?

Together, these four phases ensure the project first converges on a viable improvement opportunity, and then develops and delivers a prototype that can be evaluated both in immediate workflow impact and in longer-term organisational and societal implications.

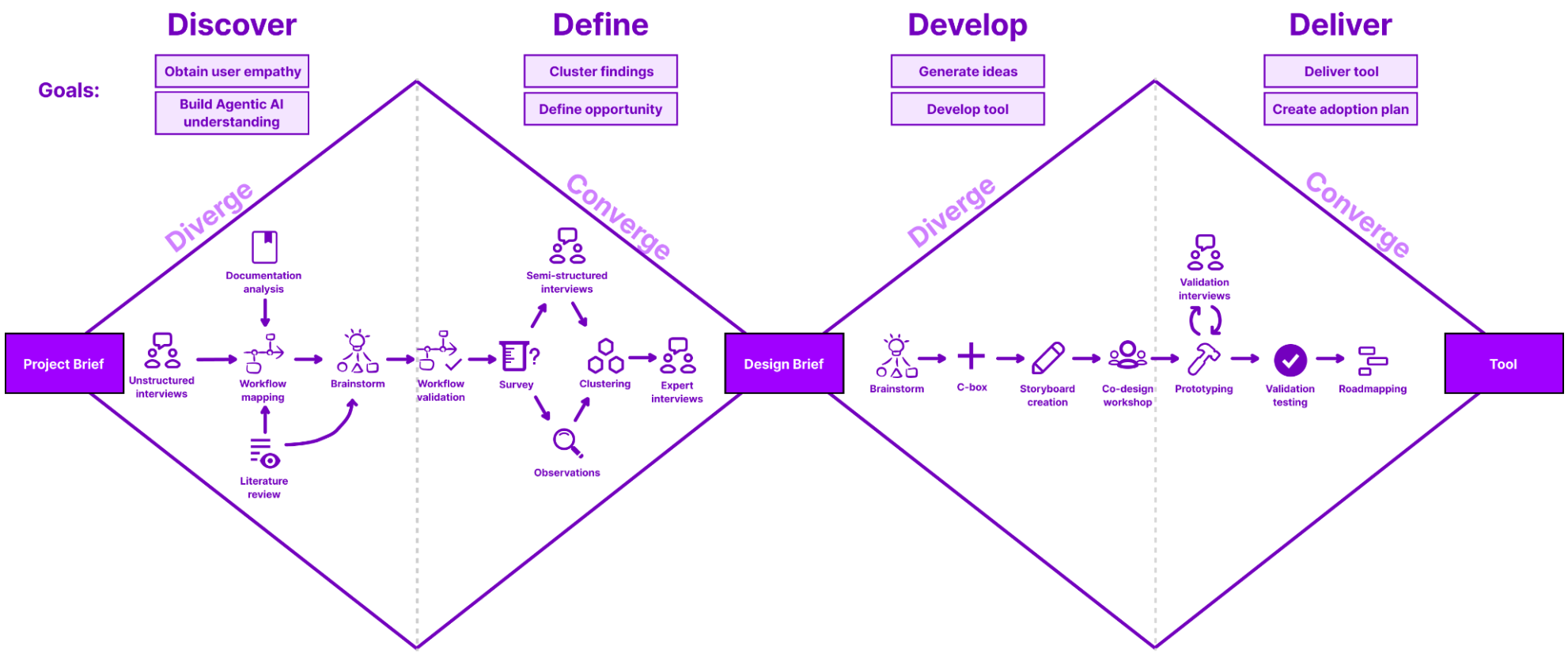


Figure 1.2: Project approach

2. Discover

In this chapter, the Discover phase of the project is described. First, a literature review is conducted to collect secondary data and establish a foundational understanding of the key concepts within the project context. This is necessary both to inform the focus of subsequent primary data collection and to generate insights that would be difficult to obtain through primary research alone within the limited project time frame. The literature review serves in particular as the main source for building an understanding of Agentic AI.

Second, primary research methods are applied to collect data on the work context, experienced challenges, and potential improvement opportunities of Business Analysts in Agile composable commerce delivery projects at Accenture Song.

Together, these methods build the understanding needed to define the problem space and identify relevant directions for design.

AI used in this chapter:

- Consensus AI and NotebookLM to support finding additional sources and analyzing sources in literature review.
- ChatGPT to support formatting text, improve text and translate interview transcriptions.

2.1 Literature Review

2.1.1 Introduction

In order to create an effective project management tool in this context, a literature review is conducted to build a foundational understanding of both the project context and the technology explored in this thesis. First, composable commerce delivery is studied to understand what these projects entail, how they are structured, and which challenges arise in practice. Second, Agile project management is studied to understand how these delivery projects are managed and how BAs operate within them. Third, Agentic AI is studied to develop an understanding of how it works, which components it consists of, how it can be applied in Agile delivery contexts, and which capabilities and challenges it brings. Together, the literature review provides the theoretical foundation for designing an effective project management tool for this context.

2.1.2 Understanding the BA's role

BAs operate within a stakeholder-rich environment, working closely with managerial project stakeholders (such as POs and Scrum Masters), development teams, clients, end-users, and often external vendors such as system providers (Dragos, 2021). They act as a bridge between all these stakeholders and facilitate effective communication. Moreover, BAs translate business needs into solution requirements in the form of **user stories**, which are brief textual descriptions of software functionality written from a stakeholder perspective and serving as boundary objects between business, design, and technical stakeholders to support a shared understanding of what needs to be built (Sporsem et al., 2025). As a result, they play a central role in eliciting, documenting, refining, validating, and prioritizing requirements, while also supporting project planning and maintaining development task lists through continuous interaction with stakeholders (Eyeregba et al., 2024; Wagner, 2010; Ndlela & Tanner, 2023; Adeleke et al., 2024).

2.1.3 Understanding composable commerce

A commerce platform is software through which organizations sell products or services online by managing storefronts, inventory, payments, and customer data. 'Composable' commerce platforms are powered by their MACH architecture, which is a modular software-architecture approach in which digital platforms are assembled from multiple independent services rather than a single tightly coupled system. This modularity is associated with faster iteration because components can be developed and deployed in parallel, and integrations can be adjusted to changing needs in near real time (Bathina, 2025; Wang, 2025). Reported benefits of MACH include substantially faster feature deployment (up to 53%), improved developer productivity and customer experience, independent scaling of components, and reduced downtime (up to 55% reduction) (Bathina, 2025; Wang, 2025). Additional advantages include simpler maintenance due to smaller, decoupled codebases (Fávero et al., 2025), and reported business impacts such as up to 20% cost savings and up to a 60% reduction in time-to-market for new features, supported by automation and cloud-native practices (Bathina, 2025; Wang, 2025).

Despite these advantages, MACH introduces project management challenges by increasing coordination complexity. More services create more integration points and a denser dependency network across teams and systems, which makes dependency management harder and increases coordination effort (Lercher et al., 2023; Lercher, 2024). A key source of complexity is evolving APIs, where versioning, backward compatibility, and inter-team alignment require continuous governance and communication to avoid issues such as stakeholder reluctance to change or degraded API design (Lercher et al., 2023; Lercher, 2024). Finally, MACH delivery often requires upskilling, as teams must learn to work with multiple services and architectural patterns, which can increase onboarding effort and initially slow delivery when knowledge is unevenly distributed (Bathina, 2025; Wang et al., 2021).

Synthesis

While composable commerce platforms offer important performance and flexibility benefits, they also make delivery environments harder to manage. Because work is distributed across multiple specialized teams and systems, information becomes more scattered, cross-team dependencies increase, and alignment requires more continuous coordination. In addition, composable architecture introduces greater technical complexity, meaning that BAs need a stronger understanding of different backend systems and their dependencies in order to effectively manage requirements, planning, and delivery progress.

2.1.4 Understanding Agile product delivery

Agile is a project management approach with principles of iterative and incremental development, customer collaboration, responsiveness to change, and regular inspection and adaptation. It is best suited for projects that have unclear or evolving end goals at the start, a need for changing/evolving requirements during the project, and therefore benefit from rapid feedback cycles, close stakeholder collaboration, and continuous reprioritization during delivery (Al-Saqqa et al., 2020; Appoh et al., 2022; Ghimire & Charters, 2022). Agile includes various frameworks. While they share a consistent underlying logic of iterative and incremental delivery, they mostly vary in meeting formats and frequencies (Waja et al., 2021; John & Sharma, 2024; Singh, 2021; Mishra et al., 2023; Alrabaiah & Medina-Medina, 2021). In this project, Scrum and SAFe are treated as in-scope frameworks as client discussions indicate that over 90% of ASC composable commerce delivery projects are run using Scrum and/or SAFe, and that projects initially set up as more structured approaches (such as Waterfall) often transition toward Agile due to changing requirements and the need for iteration during the project.

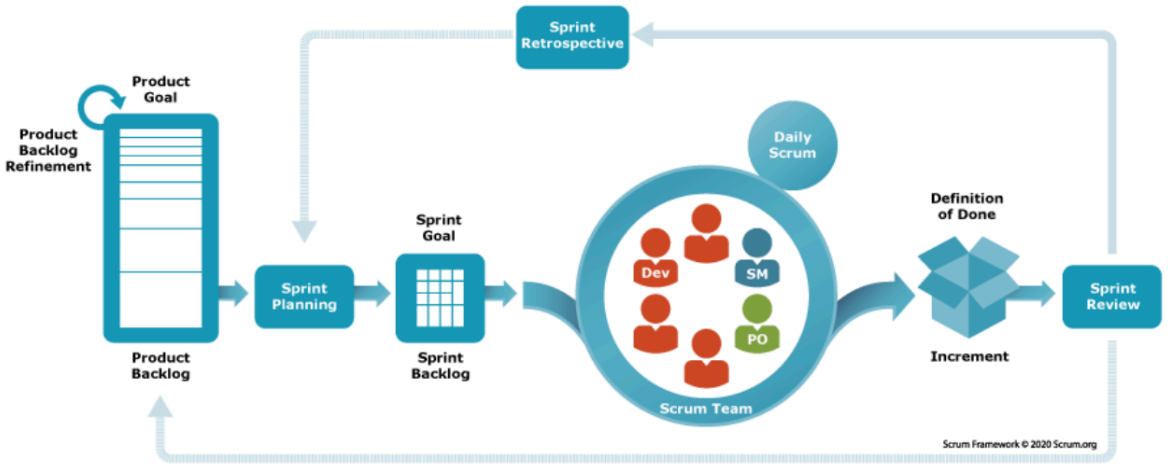


Figure 2.1: Overview of Agile Scrum (Yahya & Maidin, 2023)

Conceptually, Scrum provides the team-level operating model as a foundation for SAFe, which extends coordination and alignment mechanisms across multiple teams (Truss & Rhein, 2020). Scrum divides project work into time-boxed iterations called **sprints**, typically lasting two weeks and aiming to deliver a “potentially shippable increment” each sprint

(Kotaiah & Khalil, 2017; Alrabaiah & Medina-Medina, 2021; Al-Saqqa et al., 2020), and includes an initial setup phase (iteration 0) where objectives, teams, tools, and resources are established (Al-Saqqa et al., 2020). Its scope is managed through three key artefacts: The **product backlog** (prioritized list of PBIs), the **sprint backlog** (subset selected for a sprint), and the **increment** (the sum of completed backlog items across sprints) (Kotaiah & Khalil, 2017; Lei et al., 2017; González Moyano et al., 2022). Core roles include the PO (responsible for backlog ownership and customer alignment) and the Scrum Master (facilitator who supports Agile adherence and removes blockers) (Appoh et al., 2022; Mishra et al., 2023). Key meetings structure the sprint: **Sprint planning**, in which the sprint backlog is presented and discussed, and the sprint is initiated (Truss & Rhein, 2020; Waja et al., 2021), **Daily Scrum**, which takes 15 minutes and occurs every day during sprints in which the team discusses the project progress (Singh, 2021; Lei et al., 2017), **sprint review**, which occur at the end of each sprint in which the sprint increment is being discussed and assessed (Al-Saqqa et al., 2020), and **sprint retrospective**, which is another meeting at the end of the sprint in which the team discusses how to improve their work (Kotaiah & Khalil, 2017; Al-Saqqa et al., 2020).

SAFe is used to operate Scrum at scale: Coordinating multiple Scrum teams to deliver in large and complex environments (Itzik et al., 2023; Verwijs et al., 2024; Marinho et al., 2021). Cross-team interactions are managed through periodic alignment events: **PI planning**, where teams within an ART, which is the “Team of Agile teams” (Marinho et al., 2021), align on objectives and plan work for an upcoming 4-6 sprints (Bajpai et al., 2020), including explicit discussion of risks, progress checks, and dependency coordination (Camara et al., 2024). Moreover, **Inspect & Adapt** events function as a scaled improvement loop analogous to Scrum retrospectives, aimed at closing gaps and improving quality across the ART (Truss & Rhein, 2020).

The literature highlights recurring challenges that become more pronounced in scaled environments, which are shown in figure 2.2. First, requirement gathering and clarification can fail when stakeholders are unavailable or unable to clearly articulate their needs, resulting in ambiguous or incomplete user stories. Ill-defined user stories can lead to inaccurate work effort estimation and development of useless increments, which in turn can result in over -/under committed sprint backlogs, delays, unmet expectations (Rasheed et al., 2021; Fawzy et al., 2025; Hoda & Murugesan, 2016; Shameem et al., 2020), and eventually lower client satisfaction. Second, in scaled environments, the larger number of teams makes dependency management and planning coordination more complex, while also increasing the time required for status checks and cross-team alignment. This complicates knowledge sharing and increases the likelihood of dependency misunderstandings and planning errors (Amajuoyi et al., 2024; Shah, 2023; Hoda & Murugesan, 2016). Fragmented data and limited documentation further hinder knowledge sharing and increase the risk of planning errors, as relevant information is difficult to retrieve for effective planning and decision-making (Fawzy et al., 2025; Fissalma et al., 2024). At the

business level, these operational challenges can translate into delays, cost overruns, and reduced ROI due to slower releases and project inefficiencies (Geetha et al., 2025).

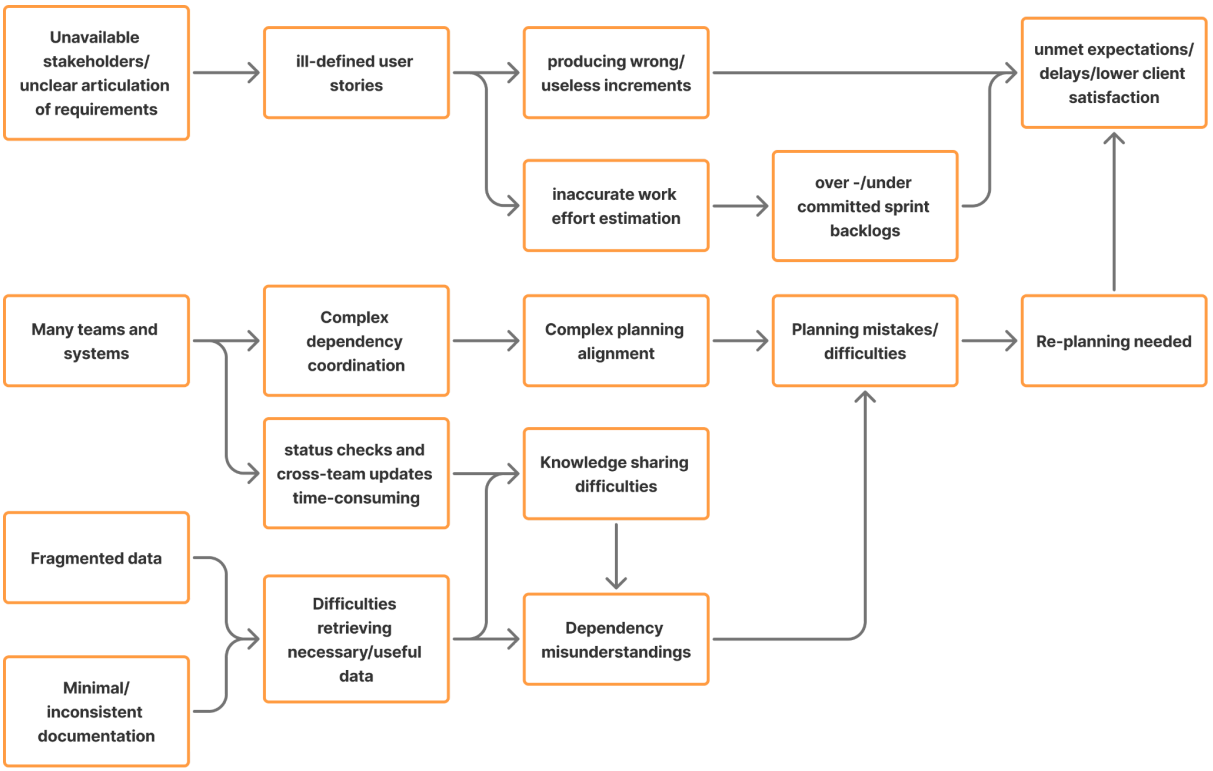


Figure 2.2: How the different challenges connect (arrows mean: Leads to …)

Synthesis

Agile intensifies managerial complexity. Its iterative nature requires frequent replanning, ongoing communication, and continuous alignment across teams and stakeholders. In projects where many teams are already involved, this makes it more difficult for BAs to maintain oversight, align plannings, track dependency impacts, and ensure that the right knowledge is shared at the right moment. These challenges are further intensified when project information is fragmented across tools and documentation is incomplete or difficult to access/retrieve.

2.1.5 Foundations of Agentic AI

To understand what Agentic AI is, traditional AI, LLMs, and GenAI have to be explained as together they form successive layers of AI capability. Traditional AI approaches (such as ML and NLP) are often built for narrow purposes and typically require large, high-quality datasets, making data collection and preparation a major cost driver (Joshi, 2025; Schneider, 2025). In contrast, LLMs are pretrained on massive and diverse datasets, which gives them broad general knowledge and strong performance across many tasks (Schneider, 2025; Xi et al., 2023; He et al., 2025). Organizations can access these capabilities through providers such as OpenAI, without training models themselves (OpenAI, n.d.). **GenAI** refers to systems that generate content (such as text, images, code) based on learned patterns, most often via LLMs (Schneider, 2025; Joshi, 2025; Narajala &

Narayan, 2025). **Agentic AI** builds on GenAI by adding stronger reasoning, memory, and interaction capabilities that enable more autonomous, goal-directed behavior over time, including decision-making, planning, and tool integration (He et al., 2025; Xi et al., 2023; Cinkusz & Chudziak, 2025). Because LLMs may lack domain-specific knowledge (Garcia et al., 2025), two adaptation mechanisms are frequently used. **Fine-tuning** adjusts a model's behavior using additional training examples for better alignment with a domain, task, or style (Xi et al., 2023), and **RAG** connects a model to external knowledge sources at run time, enabling the use of up-to-date and proprietary information without adjusting the AI model (Bharti, 2025), which thereby could make a tool deliver greater value to BAs.

Building on these foundations, **Agentic AI** moves from single-turn generation to autonomous multi-step goal pursuit by planning, taking actions, using tools, and iterating until objectives are reached (Xi et al., 2023; He et al., 2025), creating new opportunities for automation and efficiency improvements.

Synthesis

GenAI tools such as ChatGPT can be valuable, but they are relatively passive and limited in both functionality and embeddedness within existing project-management environments. As a result, BAs may need to spend considerable time manually providing the necessary context, for example by typing explanations or retrieving and uploading relevant documents. In contrast, Agentic AI can be integrated with existing project-management tools, allowing it to retrieve documentation and perform actions directly within those systems, thereby offering greater automation possibilities and practical value to BAs. Moreover, Agentic AI can operate more proactively, for example by running analyses at predefined times or in response to specific triggers, reducing the need for BAs to manually prompt the tool each time.

2.1.6 Agentic AI fundamental components and concepts

Agentic AI has several components and concepts that need to be understood when creating an Agentic AI-driven project management tool. The main component is what is called an **AI agent** which can be described as an autonomous system that interacts with an environment in iterative cycles: It perceives its surroundings in multi-modal formats via its **perception** module (its eyes and ears), reasons and plans based on those perceptions via its **brain** module (an **LLM**) (He et al., 2025), and then takes actions that change the environment or produce outputs via its **action** module (its hands and feet) (Xi et al., 2023).

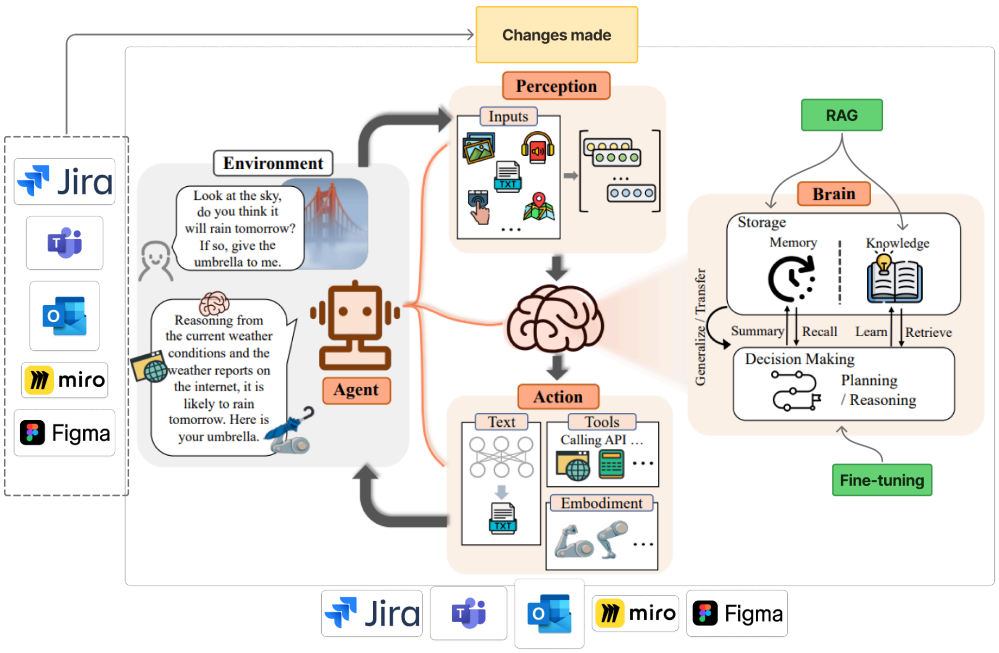


Figure 2.3: Conceptual overview of an AI agent (Xi et al., 2023)

Within the brain module, the **memory** component stores information from past interactions, so the agent can maintain context and continuity across tasks and sessions (Xi et al., 2023; He et al., 2025). An agent's memory can store its objective: The goal the agent should achieve. Objectives define the agent's constraints, role, goals, and operating instructions (Cinkusz & Chudziak, 2025; He et al., 2025). They are expressed as textual prompts (system messages) provided by the creator. The ability to make **use of tools** extend the agent's capabilities beyond just text generation, enabling them to use APIs to create or update records in databases, triggering automation workflows, or performing actions in external (project management) applications (Xi et al., 2023).

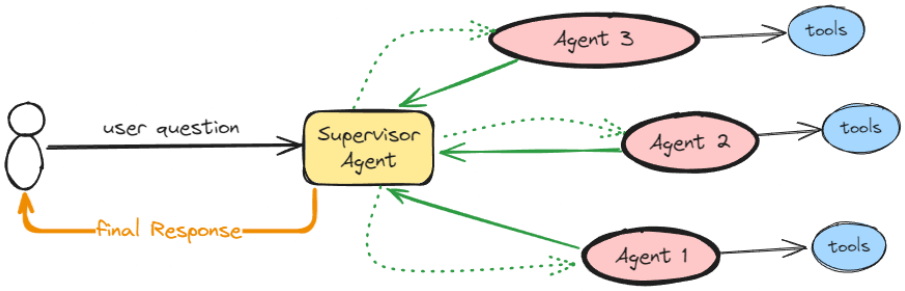


Figure 2.4: Conceptual overview of a MAS (Show, 2025)

MASs combine multiple agents that collaborate to perform a complex task. Complex tasks are often hard to perform by a single agent, and MAS allow these complex tasks to be divided into manageable ones that various agents can pick up in the system (Cinkusz & Chudziak, 2025, Xi et al., 2023). Each agent may hold a distinct role such as analysis, research, coding, compliance, or orchestrating for example (Cinkusz & Chudziak, 2025). Through coordination and collaboration mechanisms, the MAS can achieve higher

accuracy, robustness, and scalability than a single agent operating alone (Cinkusz et al., 2025). MASs are particularly valuable in environments that require expert-level reasoning, cross-validation, or complex workflows (Bharti, 2025), making it a promising candidate for an Agentic AI-driven tool for composable commerce projects.

When designing an Agentic AI tool, different MAS **coordination models** can be used to structure how AI agents collaborate in order to achieve a specific purpose. Cooperative models have agents work toward shared objectives by exchanging information and refining outputs (Xi et al., 2023). This is often organized as division of labor, where heterogeneous agents take specialized roles, or homogeneous agents share similar skills (He et al., 2025; Xi et al., 2023). Cooperation can be disordered (parallel contributions) or ordered, such as a leader–follower hierarchy (Xi et al., 2023; He et al., 2025), and adversarial models encourage agents to debate and challenge each other to improve evaluation and decision-making (Xi et al., 2023).

To enable collaboration between humans and AI, **human-Agentic AI collaboration frameworks** are developed. These describe how humans collaborate with Agentic AI and are often framed as deciding where to keep **HITL** (He et al., 2025). HITL is considered essential because human involvement provides guidance and oversight to ensure agent actions align with user requirements, objectives, and relevant safety, legal, and ethical standards (Xi et al., 2023). Because AI systems can hallucinate, their outputs may contain factual errors or fabricated claims. If these issues are not caught through HITL for example, they can propagate into deliverables and lead to serious consequences in consulting, such as misguided recommendations, loss of client trust, and reputational damage, as illustrated by a Deloitte case in which a report included fabricated sources (The Guardian, 2025). In the Instructor–Executor framework, humans act as supervisors who assign tasks, evaluate agent outputs, and provide corrective feedback, while agents function primarily as executors, making this framework suitable when reliability, personalization, and safety are critical (Xi et al., 2023). In the Equal Partnership framework, humans and agents collaborate as peers: They jointly plan, reason, and execute tasks, with agents actively contributing insights and recommendations rather than only following instructions (Xi et al., 2023), essentially giving the MAS more autonomy.

Lastly, MAS can be static (predefined roles and workflows) or dynamic (agents instantiated on-demand), trading predictability for adaptability and scalability at the cost of higher coordination and compute complexity (Wang et al., 2024; Hong et al., 2024; Xi et al., 2023).

Synthesis

AI agents (with memory and objectives), RAG, tool use, and MAS with appropriate coordination models, are considered highly relevant for designing the Agentic AI-driven tool and feasible within this project's timeframe. In addition, when designing the tool, one of the human–AI collaboration frameworks should be considered. As the choice of framework is a fundamental part of how the MAS is structured and how the BA interacts with it, selecting

the appropriate framework will therefore likely emerge naturally during the MAS design process. In contrast, more advanced topics such as fine-tuning and dynamic agent/MAS scaling introduce substantial orchestration, computational complexity and technical knowledge. Therefore, they are considered out of scope for deeper investigation in this project.

2.1.7 AI use cases in Agile delivery

Across the reviewed literature, AI is found to be supportive in Agile delivery by reducing manual effort in requirements gathering, planning, monitoring, and documentation, with increasing emphasis on agentic and multi-agent approaches. GenAI and LLM-based tools can generate requirement specifications, user stories, and acceptance criteria from prompts, accelerating early backlog creation (Bahi et al., 2024; Sami et al., 2024). NLP and embedding techniques are used to extract requirements from unstructured documents and to detect relationships between backlog items (dependencies) (Radwan et al., 2025). Agentic, MAS-based systems replicate Agile role responsibilities: For example, agents generate and refine backlog items, negotiate sprint feasibility, and produce sprint artefacts and reports (Nguyen et al., 2024; Cinkusz & Chudziak, 2025).

AI is also applied to backlog prioritization and structuring at scale. Approaches combine semantic similarity, dependency graphs, fuzzy logic, clustering, and optimization to rank, group, and reprioritize requirements, often with integrations into tools such as Jira (Radwan et al., 2024; Radwan et al., 2025). Moreover, decision-support systems further extend these capabilities by monitoring sprint metrics, predicting risks, balancing workloads, and analyzing stakeholder sentiment in communications (Almalki, 2025). In large-scale Agile settings, AI assistants are described as useful for cross-team alignment and dependency visibility during PI planning and coordination (Saklamaeva & Pavlič, 2024).

Synthesis

The AI-driven decision-support system presented by Almalki (2025) shows significant efficiency optimization results (resolution time reduction of 37,5%, 25% improved workload balance, 30% decrease in idle time, 18% improvement in sprint backlog completion rate), showing that AI tools have significant potential to positively impact work efficiency in Agile delivery. These use cases demonstrate that AI can support a wide range of activities and stages within Agile delivery, and that different techniques can be applied to serve different purposes.

2.1.8 AI challenges in Agile delivery

Although AI shows strong potential across Agile delivery, the literature also highlights major limitations. A recurring limitation is weak organizational and domain context understanding: AI can produce well-structured requirements and plans that still misalign with stakeholder needs, project constraints, or feasibility, particularly when user stories are brief and lack context (Bahi et al., 2024; Barcaui & Monat, 2023; Sporsem et al., 2025). Human trust and

repeatability are further undermined by inconsistent outputs and hallucinations (Bahi et al., 2024).

Moreover, dependency modeling remains difficult: Approaches ignore dependencies or capture them only partially, limiting usefulness in complex Agile ecosystems where sequencing and integration risks matter (Radwan et al., 2024; Radwan et al., 2025). Also, data dependency constrains ML-based methods, which often require large, clean historical datasets and are sensitive to noisy textual inputs (Radwan et al., 2024; Radwan et al., 2025; Almalki, 2025).

While MASs composed of many smaller, specialized agents may perform better on complex tasks, increasing the number of agents can also raise computational costs and intensify LLM context-limit issues, which may ultimately reduce accuracy, especially when agents debate (Nguyen et al., 2024; Xi et al., 2023; He et al., 2025). In addition, a larger number of agents increases the risk that faulty intermediate outputs or hallucinations produced by individual agents propagate through the system and lead to unreliable final results (Nguyen et al., 2024; He et al., 2025).

Finally, the literature highlights limited industrial validation and significant adoption barriers. Many approaches are evaluated in controlled or synthetic settings rather than in live Agile organizations, while methodological and orchestration complexity can limit practitioner uptake (Radwan et al., 2024; He et al., 2025; Nguyen et al., 2024).

Synthesis

For an AI agent to perform effectively and deliver meaningful value, the LLM that is used as its brain requires RAG to obtain sufficient organizational and domain-specific context. Additionally, when designing the tool, it is important to account for the challenges associated with increasing the number of agents within the tool, particularly higher computational costs and a greater risk of hallucinations. To mitigate these risks, the tool should incorporate appropriate control mechanisms, such as an AI agent that evaluates output for example, and deterministic code and/or HITL checks, before any final action is executed.

2.1.9 Key takeaways

Composable architecture and Agile amplify BA pressure and managerial complexity

Together, composable architecture and Agile delivery can be seen as amplifiers of BA pressure and managerial complexity. They increase the need for system understanding, dependency awareness, status tracking, and cross-team communication, while simultaneously making these activities more difficult and time-consuming. The challenges visualized in figure 2.2 point to clear opportunity areas for BA support the tool could address, such as helping BAs retrieve scattered information, improve documentation, maintain status awareness, and coordinate dependencies more efficiently.

Design specialized agents to improve tool performance and scalability

Adopting specialized AI agents with a single, clearly defined, assigned task is preferable because it makes the overall tool easier to manage, improve, and scale. When 1 AI agent is responsible for many different tasks, such as retrieving project data, analysing risks, generating documentation, and sending updates, changes to improve 1 capability can unintentionally reduce performance in another. This makes the system harder to control and more difficult to adapt over time. It is similar to applying composable architecture principles in MAS design: Each agent functions as an independent building block that can be developed and maintained separately, while still contributing to the overall tool. This makes the tool more flexible, robust, and future-proof, as individual AI agents can be optimized or swapped when project needs change without harming the full functionality of the tool.

Design for minimal AI, maximum BA benefit

Because AI agent tasks and API tool calls increase token consumption and computational cost, the design should deliberately balance AI functionality with efficiency. In practice, this means aiming for a minimal set of AI-driven actions that still deliver clear value, rather than maximizing automation or utilizing AI for its own sake.

Design for efficient context capture to prevent misalignment

The tool should be designed to capture sufficient organizational and domain context efficiently through strategies such as RAG, as limited context can cause AI outputs to misalign with BA needs, stakeholder priorities, or project constraints. This implies that mechanisms for structuring and supplying context are a core requirement rather than an afterthought.

Agentic AI capabilities as reference and building blocks for tool design

The literature implies a set of Agentic AI capabilities that are directly relevant as reference points for designing the tool. They are shown in table 2.1 below.

#	Agentic AI capability
1	Perceive, receive and get triggered (such as detecting new messages, artefact changes, or sprint events).
2	Reason and plan (such as interpreting context, form a multi-step approach).
3	Simulate scenarios and stakeholders (such as generating alternative interpretations, anticipate stakeholder concerns).
4	Discuss and evaluate (such as critiquing outputs through multi-agent review patterns).
5	Decide which option to take (decision making) (such as selecting recommendations based on constraints and evidence).
6	Perform tasks and use tools autonomously, including: 1. Generate multi-modal content (such as producing structured artefacts and, where relevant, visuals).

	2. Analyze data, run calculations, and more (such as computing metrics, compare scenarios, detect risks).
7	Reflect (such as self-checking outputs, identify uncertainty or missing inputs).
8	Learn and adapt (such as adjusting behavior based on feedback and recurring patterns).

Table 2.1: Discovered Agentic AI capabilities

These capabilities are used as building blocks for the design of the tool. They serve as reference points for identifying which forms of Agentic AI support could be relevant to BAs, which is further explained in chapter 4 of this report. Rather than being adopted as a fixed blueprint, these capabilities provide a starting structure for exploring how the solution could deliver value in practice, which capabilities are available, and how they could be combined within the tool.

2.2 Research methods

A mixed-method research strategy is used to build a grounded understanding of BA work at ASC in composable commerce, Agile delivery contexts and to identify workflow inefficiencies, frustrations, tool use, and AI-enabled improvement opportunities. Combining various methods enables **methodological triangulation** (Bekhet & Zauszniewski, 2012): insights from different sources and methods can be cross-checked rather than relying on a single perspective. This triangulation is considered good practice because it increases the credibility and validity of the research findings, reduces method-specific bias, and strengthens confidence in the conclusions. Importantly, data collection does not follow a pre-defined linear sequence. Instead, it is conducted iteratively: Defined and synthesized findings from earlier methods inform the focus, questions, and priorities of subsequent activities, allowing the research to progressively narrow in on the most relevant workflow steps and challenges.

2.2.1 Methods' justification & purpose

Unstructured interviews are used at the start of the project to quickly develop a broad, exploratory understanding of (1) what BAs do in practice in this context and (2) what BAs deliver. This early exploration supports the initial construction of a BA workflow map and helps define which topics to examine further through more structured methods later in the study.

Internal and Jira documentation analysis is used to better understand how BA work is scheduled and which tools they use. Jira is one of the main project-management tools BAs work with; therefore, analyzing Jira documentation offers valuable insights and leads to a better understanding of how BAs work. This method supports iterative improvements of the workflow map (appendix F).

A **survey** is used to collect input efficiently from multiple BAs and to quantify which workflow areas and AI-driven improvement opportunities, derived from the How Might We brainstorm explained in chapter 2.2.5, are most valuable to investigate further. The survey also includes qualitative questions to capture concrete examples of current AI usage, desired support, and perceived AI limitations.

Observations are conducted to capture what BAs actually do in context, including tool usage, collaboration patterns, and the practical small things that are often hard to recall or articulate accurately in interviews. This method is used to validate and enrich the workflow map and to identify friction points that may not emerge through interviews alone. In addition, shadowing enables the researcher to build stronger user empathy. Building empathy is widely considered essential in user-centred design because it improves the ability to design interventions that fit user needs and real-life contexts.

Semi-structured interviews are used to deepen understanding of the 3 main improvement opportunities identified within the BA workflow: Dependency management, meeting processing, and requirements gathering and user-story creation. These interviews support comparison across participants while still leaving room to ask follow-up questions and capture unexpected insights.

2.2.2 Sampling

A purposive sampling strategy is applied to recruit participants who can provide the most relevant insights into BA work efficiency in Agile composable commerce delivery projects.

Participant selection is guided by two criteria:

- Experience at Accenture: Participants need sufficient organisational familiarity to comment on how work is executed in practice at ASC, including routines, collaboration patterns, and constraints relevant to project delivery.

- Experience as a BA: Participants need direct exposure to BA responsibilities to ensure that data reflects real task ownership, day-to-day decision making, and operational challenges.

Within this purposive strategy, sampling is tailored per method to match the objective of each method:

For the survey, the intention is to collect a variety of perspectives rather than a statistically representative sample. Participants are therefore selected to capture variation in relevant backgrounds, such as different levels of Accenture experience and BA experience, enabling the survey to surface a broad range of perceived challenges and improvement opportunities in BA work.

For interviews, participants are selected to ensure minimum relevant experience and domain fit. BAs are included only if they have at least one year of experience, ensuring they can reflect on BA work beyond initial onboarding. In addition, interviewees need experience in composable commerce delivery projects, or in projects that the client considers highly similar, to ensure that insights are grounded in the project context.

To complement BA perspectives, experts and higher-level consultants are also interviewed. Their inclusion serves to triangulate findings by capturing how more senior roles perceive BA work, expectations, and recurring challenges, thereby enriching the interpretation of BA pain points and opportunities for improvement.

Overall, purposive sampling enables the selection of participants with direct and context-relevant experience, while balancing breadth (survey variety) and depth (experienced BAs, senior perspectives, and intensive observation) to support the study's design and validation goals.

Participant	Job title/role	Years at Accenture	Years as BA
P1 (BA)	Management Consultant	3	3
P2	Senior Manager	4	0 but manages many

Table 2.2: Unstructured interview participants

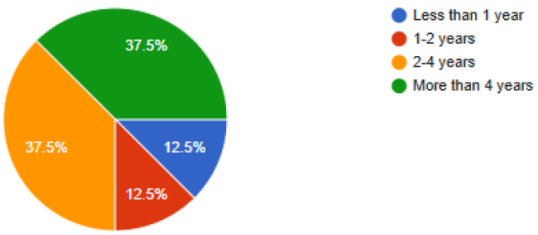
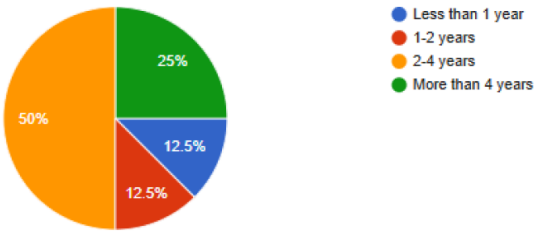
Years at Accenture	Years as a BA
	

Table 2.3: Survey participants (n = 8)

Participant	Job title/role	Years at Accenture	Years as BA
P4 (BA)	Management Consultant	3	3 mels

Table 2.4: Observation participant

Participant	Job title/role	Years at Accenture	Years as BA
P3 (BA)	AI & Data Consultant (niet)	4	4
P4 (BA)	Management Consultant	3	3
P5 (BA)	Digital Business Analyst	1	1

P6 (BA)	Digital Integration Consultant	4	4
E1 (expert)	Digital Business Consultant	4	n/a
E2 (expert)	AI/ML Computational Science Specialist	4	n/a
E3 (expert)	Digital Business Senior Manager	9	n/a

Table 2.5: Semi-structured interview participants

2.2.3 Data collection

Interviews (unstructured and semi-structured)

Unstructured interviews lasted approximately 30 minutes; semi-structured interviews lasted approximately 60 minutes. All interviews are audio-recorded with voluntary informed consent.

For the semi-structured interviews, a peer-reviewed interview script (Appendix A) is developed to maximise clarity, neutrality, and openness. The script is reviewed by two SPD master students and one SPD alumnus. Interviews are structured around the 3 improvement opportunities, and within each opportunity covered: (1) workflow steps, tools and artifacts, (2) challenges and inefficiencies, and (3) current tool/AI usage and limitations, followed by a final section on general AI-related issues.

Semi-structured interview questions are explicitly designed to be:

- Open and neutral (non-leading): Leading language is avoided by phrasing questions to invite description rather than confirmation (such as replacing “Do you struggle with gathering requirements?” with “What is your experience with requirements gathering?”).
- Clear and singular: Each question addresses one concept to reduce ambiguity and cognitive load for participants.

All recordings are transcribed using Microsoft Word’s transcription functionality and subsequently checked and corrected by the researcher. When interviews are conducted in Dutch, transcripts are translated using ChatGPT and then reviewed again by the researcher to ensure translation accuracy.

Internal and Jira documentation analysis

Relevant items are captured through screenshots of internal slides and Jira web pages/examples. The documentation analyses can be found in Appendix B.

Survey

The survey combines quantitative and qualitative sections. Quantitatively, respondents rate their interest in receiving support across workflow improvement opportunities using 5-point Likert scales (1 = not interested, 5 = very interested), which is used to capture BA desirability towards the improvement opportunities. Qualitatively, open questions capture

respondents' current AI usage, other desired support, and experienced challenges. The complete survey instrument is included in Appendix C.

Observations
Meetings and work sessions are observed and structured notes are taken on tasks performed, stakeholders involved, tools and artifacts used, and time spent per activity. Notes are recorded using a structured note-taking guide (Appendix M).

2.2.4 Data analysis

Exploratory interviews analysis
The unstructured interview transcripts are used directly to construct the **first version of the BA workflow map**. Insights are synthesized into workflow steps, deliverables, tools, and stakeholder interactions. At this stage, **no formal coding procedure** is applied; analysis serves an exploratory purpose to establish an initial representation of BA work and to support subsequent data collection.

Semi-structured interviews and observations analysis
Semi-structured interview transcripts are analyzed using a hybrid thematic analysis approach (Braun & Clarke, 2006). This combined:

- Deductive coding, using pre-defined themes aligned with the study focus (such as activities, challenges, deliverables, stakeholder interactions, tool/AI use), and
- Inductive coding, to capture emergent themes not covered by initial categories.

Transcripts are checked against recordings to ensure accuracy, coded in FigJam, and codes are iteratively grouped into themes. These themes are organised into a thematic map to support comparison across participants and to derive cross-cutting insights.

Observation notes are consolidated and analysed using the same thematic analysis approach applied to the interview data. Notes are coded with a focus on observed activities, tools, artifacts, stakeholder interactions, and friction points. Findings are used to complement and contextualise interview and survey insights by adding behavioural and situational evidence from day-to-day delivery work.

Documentation analysis
Screenshots of internal training slides and Jira artifacts are analyzed to identify BA activities, delivery artifacts, terminology, and workflow patterns/structure. Concepts from documentation are mapped and linked in FigJam to iteratively refine the workflow map. This also enables cross-checking whether workflow steps derived from interviews are reflected in actual project tooling and examples.

Survey analysis
Quantitative survey responses are analysed by calculating the mean score per item to compare interest levels across identified workflow improvement opportunities. Qualitative

responses are incorporated into the thematic analysis, contributing additional examples of experienced challenges, current AI usage, and other desired support.

2.2.5 Data synthesis (key findings)

All the derived quotes, themes, tools, artifacts and other findings can be found in Appendix D and E. The report contains lists of key recognized BA activities:

Name	What the BA does	Source
PI planning session	1. Attend the meeting. 2. Create notes for your team(s)	Literature review, interviews
Requirements gathering	1. Receive BRs from business stakeholders/PO 2. Translate BRs to FRs, NFRs and TRs by meeting with stakeholders and mapping user flows.	Observations, interviews, doc. analysis
Spotting dependencies	1. During meetings with stakeholders, while presenting your work, ask stakeholders for input on dependencies. 2. Write these dependencies down, and insert them in the Jira PBIs: PBI 1 “blocks” PBI 2. 3. Map these dependencies in Jira/Miro to create a visual overview of each cascading dependency	Interviews
Design PBI creation	1. Translate these requirements into epics and features, meet with stakeholders if unsure to receive their input 2. Translate these features to design PBIs 3. Check in with PO to assign priority levels to design PBIs	Interviews, doc. analysis
Design research & user testing	1. Provide designers with prioritized design PBIs, so they know what to work on. 2. Support designers during design research	Doc. analysis, interviews
Development PBI creation	1. Translate UX/UI designs created by designers into features, components and user stories containing happy flow (when software works) and unhappy flow (if there is an error or if software doesn’t work for example). 2. Create PBIs of features and user stories 3. Ask stakeholders for input if you need it in order to finish a PBI 4. Label PBIs as “Refinement ready”	Doc. analysis
Perform pre refinement meeting tasks	1. Create and gather “Refinement ready” PBIs^ 2. Check which stakeholders should be in the meeting. 3. Send refinement meeting agenda to all meeting attendees (stakeholders) and ensure that everyone has read them. 4. Make sure the Figma/Miro is ready and contains all the designs and user flows you’re going to present	Doc. analysis, observations
Refinement meetings	**1.** Present and walk through the Figma/Miro and explain the functionalities and details of the features/components etc. 2. Ask if developers have any questions. 3. Present the Jira PBIs. 4. Ask if developers have any questions. 5. Start story poker to estimate the time/effort needed to finish each PBI. 6. Place PBIs that have an estimation in the “Ready for planning” board in Jira.	interviews, doc. analysis
Perform pre sprint planning session tasks	1. Check in with each developer and ask for status updates of each PBI (which are done? Which spill to the next sprint (delayed)?). 2. Check if additional PBIs are needed to be created and assign other PBIs to that developer if there is enough capacity left in the active sprint. 3. Create additional/follow-up PBIs if needed. 4. Check team capacity and align with Scrum master stakeholder to confirm capacity. 5. Check in with PO to assign PBIs to the next sprint you are going to plan.	Doc analysis, interviews,

Sprint planning session meeting	1. Go over the active sprint board: Are these PBIs done? Which spill to the next sprint (delayed)? 2. Check sprint goal: Did we achieve the goal? 3. Close the active sprint. 4. Check/confirm team capacity: Is everyone available? 5. Go over new sprint PBIs per stakeholder: Does stakeholder understand and agree with PBIs? 6. Set the sprint goal. 7. Start the sprint.	Doc analysis
Daily standups	1. Attend the meeting. 2. Present any important topics (progress, questions, tasks). 3. Take notes of important topics mentioned by stakeholders. 4. Turn on CoPilot to automate summarization and action item generation. 5. Create action items. 6. Update documentation in Confluence. 7. Update stakeholders by sending them messages. 8. Perform action items.	Interviews, doc analysis, observations
Retrospective meetings	1. Attend the meeting. 2. Take notes of important topics (What do we need to do differently to improve the process?). 3. Turn on CoPilot to automate summarization and action item generation. 4. Create action items. 5. Update documentation in Confluence. 6. Update stakeholders by sending them messages. 7. Perform action items.	Observations, doc. analysis

Table 2.6: List of BA activities

The collected primary data shown in table 2.6 and Appendix E, is used to iteratively map the BAs activities, artifacts and tools used in a sequential order, to create the BA workflow map (figure 2.5), which is validated during a workflow validation session with participants P4, P5 and P6. During this session, the workflow map, together with the initial generated ideas (table 2.7), are presented and feedback is gathered, after which adjustments are made and the improvement opportunities are formulated. Together with the recognized Agentic AI capabilities found in table 2.1, the workflow map acts as a point of reference for the How Might We brainstorm (Appendix G), of which the generated ideas are translated to identify **8 potential improvement opportunities** in the BA workflow (table 2.7).

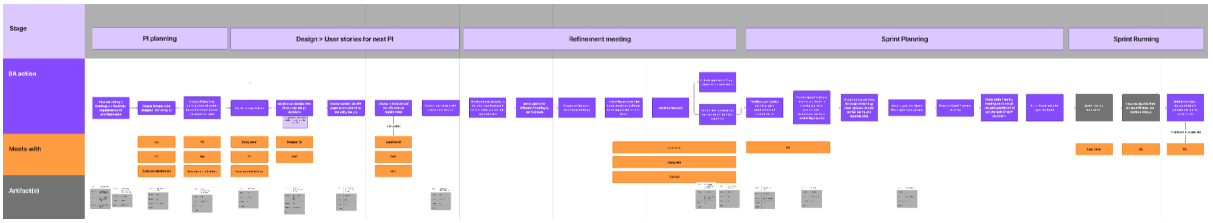


Figure 2.5: Detailed workflow map. A large version can be found in appendix F.

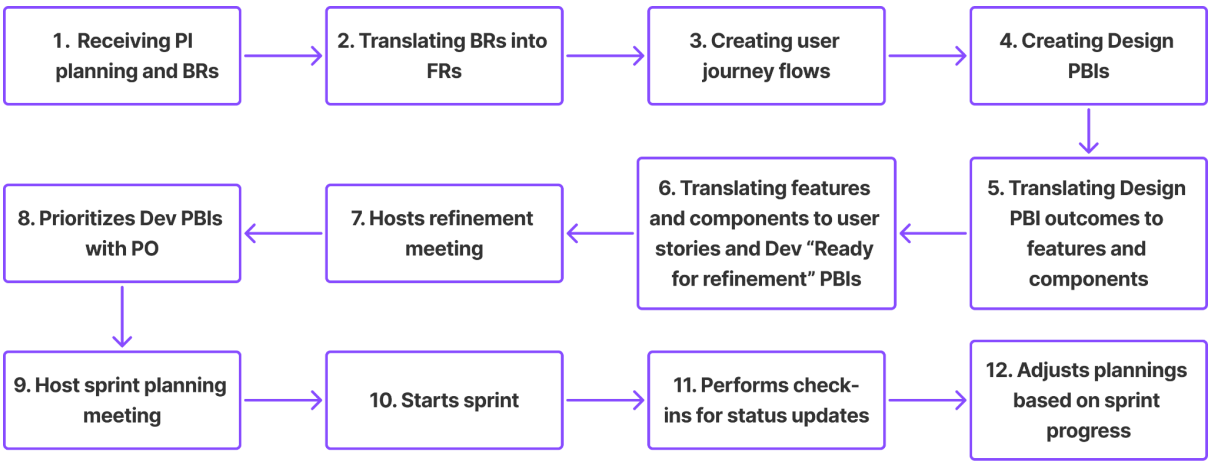


Figure 2.6: Simplified version of BA workflow map

Idea explanation	Opportunity area
A tool that automates the translation of strategic roadmaps/list of epics/BRs to Dev PBIs. --- opportunity area 1 1. BA feeds an excel or image containing the list of BRs to the tool. 2. Tool translates it to FRs, NFRs, TRs 3. BA checks, tweaks and confirms 4. Tool translates to design PBIs 5. BA checks, tweaks and confirms --- 6. Designer creates UI/UX --- opportunity area 2 7. BA feeds UI/UX to the tool 8. Tool translates to features & components 9. BA checks, tweaks and confirms 10. Tool creates user stories 11. BA checks, tweaks and confirms 12. Tool adds acceptance criteria 13. BA checks, tweaks and confirms 14. Tool creates Dev PBIs 15. BA checks, tweaks and confirms ---	**1. User journey/flow creation (translating BRs into design PBIs)** **2. User story creation (decomposing UX/UI designs to features and components and creating user stories)**
A tool that simulates usage of feature/software/product to collect (synthetic) user insights	**3. Receiving feedback on designs and user stories ("Have you thought about …?")**
A tool that collects, analyzes, summarizes and shows important data to BAs, such as a dashboard that shows dependencies and next steps that should be taken to manage/spot these dependencies	**4. Managing dependencies**
A tool that analyzes data and runs predictive calculations to provide dynamic suggestions on PBI prioritization and estimation	**5. PBI prioritization and estimation**
A tool that simulates workflow scenarios to find the best planning/task sequence approach	**6. Sprint planning (how many and which PBIs in this sprint?)**
A tool that schedules meetings based on PBIs that are in the "Ready for refinement" or "Ready for planning" Jira boards	**7. Planning meetings (who needs to be in this meeting?)**
A tool that transcribes and summarizes meetings and gathers insights and action points + information such as: Conflicts, questions, dependencies, low team morale, user feedback, etc.	**8. Processing meetings (generating action items, updating stakeholders and documentation)**

Table 2.7: Generated ideas translated into improvement opportunities

3. Define

In this Define phase, the improvement opportunities identified in the Discover phase are narrowed down to one focused design direction. First, the most promising opportunities are compared based on BA desirability and explored in more depth through clustered findings from the interviews, observations, and survey. These insights are then used to determine which opportunity offers the strongest potential. The chapter concludes with the selection of managing dependencies as the central focus of the project and translates this into a design brief that guides the Develop phase.

AI used in this chapter:

- ChatGPT to support formatting text and improving text

3.1 Improvement opportunities

To decide where to further investigate the BA workflow and conduct more targeted interviews, BAs desirability towards the 8 improvement opportunities is captured via the quantitative part of the survey as described in chapter 2.2.3.

Improvement opportunity (a tool that supports BAs with …)	**BA desirability (1 to 5)**
User journey/flow creation (translating BRs into design PBIs)	3.75
User story creation (decomposing UX/UI designs to features and components and creating user stories)	**4.75**
Receiving feedback on designs and user stories (“Have you thought about …?”)	3.75
Managing dependencies	**4.63**
PBI prioritization and estimation	3.50
Sprint planning (how many and which PBIs in this sprint?)	3.50
Planning meetings (who needs to be in this meeting?)	3.50
Processing meetings (generating action items, updating stakeholders and documentation)	**4.75**

Table 3.1: BA desirability towards improvement opportunities

This shows that BAs have the highest desirability towards receiving support in 1) user story creation, 2) dependency management and 3) processing meetings. The opportunity “user story creation” is broadened to “Requirements gathering and user story creation,” as client discussions indicate that requirements gathering is an ongoing activity throughout the creation of user stories, including after design tickets are produced.

These 3 opportunities are selected for deeper investigation through semi-structured interviews and observations. The resulting insights are clustered, and key challenges within each opportunity are identified.

3.2 Clusters

In this section, each of the 3 improvement opportunities are described in more detail, diving into what the opportunity area really is: What the BA does and is responsible for, what challenges exist in the opportunity area, and how AI might offer support.

3.2.1 Processing meetings

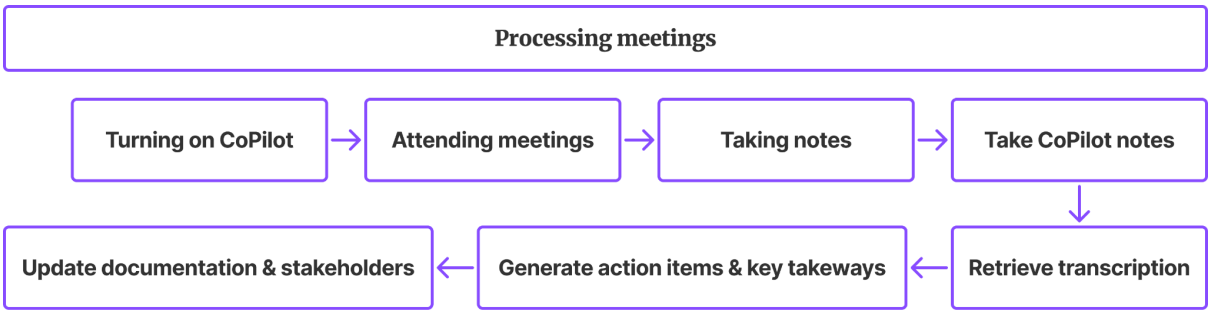


Figure 3.1: Processing meetings workflow

Processing meetings is an activity that occurs across the entire BA workflow, as the BA continuously meets to receive input from stakeholders on the artifact it is working on. Processing meetings begins by attending meetings. Although individual meetings vary in purpose and content, they typically involve presenting topics and discussing decisions and next steps. The BA may present in some sessions, but often acts as a participant and coordinator. During meetings, notes are usually captured manually, often by the BA, while transcription tools and Microsoft Copilot may also be used to record the conversation. After the meeting, the BA is responsible for translating the discussion into tangible outputs. This includes creating or updating project documentation, adjusting or creating product backlog items (PBIs), formulating action items for relevant stakeholders, and communicating updates to those stakeholders. To complete these tasks, BAs may revisit the meeting transcript to extract decisions, requirements, and follow-up actions.

Much cognitive effort required to extract actionable outcomes

Processing meetings is cognitively demanding because BAs must separate relevant decisions and next steps from extensive discussion. Rather than documenting everything that was said, the BA needs to distill the conversation into concise, actionable outcomes. The challenge is to *“Really capture everything that is needed… but don't overcomplicate it… there's a lot of discussions going on… you really don’t need all these discussions… but really the sense conclusion.” P3*. This filtering effort requires the BA to continuously interpret what was agreed, what remains open, and what information is essential for documentation and follow-up. BAs describe this as extracting direction and context: *“I think the hardest*

part is more like: OK, how do we move forward, and what's the context? Making sure you extract everything properly." P4.

Limited accessibility of meeting transcripts

In addition to the cognitive workload, observations showed transcript availability can be a practical bottleneck. Meeting transcripts are not always directly accessible to BAs, as client stakeholders may own the recording or transcription and must share it manually. This dependency can delay post-meeting work; in observed cases, BAs waited multiple days for transcripts because the client stakeholder responsible for sharing them was unavailable.

Inaccuracy and readability issues in transcripts

Finally, transcript quality can limit how effectively BAs can rely on them. Automated transcripts are sometimes incomplete or difficult to interpret due to tool limitations and meeting conditions: *"Sometimes it picks up half things, or not properly, and people speak unclear, speak fast, maybe not very good English or half English." P4*. Accuracy issues can also arise when multiple people speak from the same physical space but audio is captured inconsistently: *"... limitation of ... Teams ... What you also see is that people are in the same room, and one person has their mic off and the other doesn't, but there are essentially two people talking through one microphone. ... we don't have an issue with it, but the technology does ..." P4*. These limitations reduce trust in the transcript and increase the effort required to reconstruct what was actually said, which in turn slows down documentation and follow-up actions.

AI to increase efficiency by reducing cognitive load and automating transcript retrieval tasks

As the literature review shows, improving documentation could offer significant improvements to managing Agile composable platform delivery projects. Additionally, retrieving and analyzing transcripts, extracting important information, and generating (textual) documentation, are great tasks to be performed by AI as its textual language analysis and generation capabilities are proven to be very good, and those tasks are considered administrative tasks, which should be automated as much as possible according to interviewed expert E1. AI could very likely be used to offer support to BAs in this activity and increase BA efficiency by 1) lowering the necessary cognitive effort of BAs, and 2) decreasing the time BAs spent on processing meetings by supporting documentation generation and sending status updates to stakeholders for example.

3.2.2 Requirements gathering & user story creation

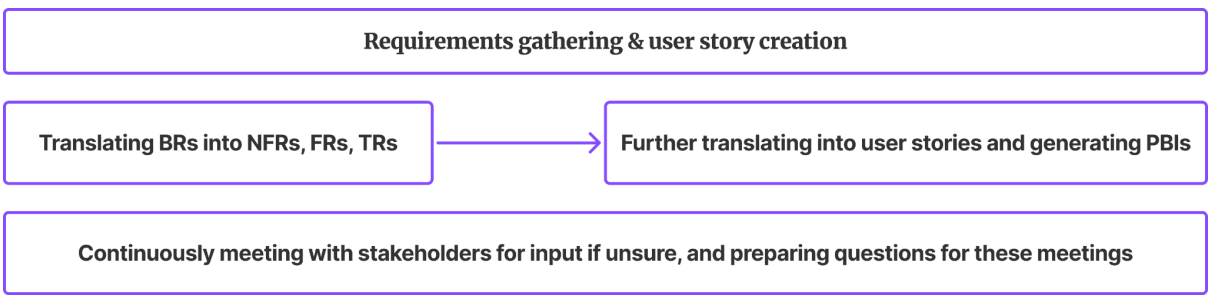


Figure 3.2: Requirements gathering & user story creation workflow

Requirements gathering refers to the activities through which BAs elicit, consolidate, and structure what needs to be built. This starts with meetings with business stakeholders, where requirements are captured by extracting relevant information from conversations and existing materials: *"You do requirement gathering by extracting information/data from meetings, etc." E1*. The BA then translates these BRs into more specific requirement types: FRs, NFRs, and TRs, so they can be used to guide design and development work: *"You get business requirements. Based on the business requirements, I create a design PBI" P3*. Because many requirements depend on technical feasibility and system constraints, BAs also meet with technical stakeholders to understand technical complexity and to formulate technical requirements: *"…we also need to have sometimes some discussions with the architect… to see how actually does it work… Is there any technical complexity…" P3*. User story creation is the process of translating requirements into implementable units of work for Agile delivery. It involves breaking down large tasks into smaller items that create a shared understanding of what to build and can be completed within a single sprint (Sporsem et al., 2025). For BAs, this requires producing user stories that are sufficiently explicit for developers, including detailed (technical) requirements and step-by-step descriptions: *"Really step by step details… developers… really need it detailed." P3*. BAs are responsible for gathering requirements and delivering complete user stories/PBIs.

Interpreting stakeholder intent and ambiguity is difficult

A central difficulty is translating what stakeholders mean into precise requirements, which is also recognized as a key challenge in the literature review (figure 2.2). Stakeholders may describe desired outcomes without specifying operational details, edge cases, or constraints. This makes it challenging to define how a feature should work in practice: *"To really define what someone from the business or IT actually means is a skill that makes this role quite challenging." E1*.

Limited system knowledge across complex back-end landscapes

To formulate requirements and user stories effectively, BAs need sufficient understanding of the relevant systems and their technical feasibility. However, acquiring and maintaining this knowledge is perceived as highly difficult, as it depends heavily on interactions with technical stakeholders to obtain the necessary information: *"You need some input from the business, but also input from the developers, from the architect… sometimes it gets really technical." (P3)* and considered difficult to fully understand and maintain due to the amount of different systems in composable platforms: *"There are so many different backend systems, we... need context about all backend systems... You can't know everything about every system..." P5*.

Dependency-driven user story information gaps

Even when a user story is well-defined, essential input may still be missing because it depends on other teams or stakeholders. This creates delays and prevents BAs from finalizing requirements and stories on time: *"You can have a super tight ticket, it's completely clear …, but the information still isn't available." P4*.

Gaps in refinement meetings as a quality control mechanism

Finally, refinement meetings are designed to flag ill-defined user stories, but they do not always function as an effective safeguard. When developers do not consistently challenge assumptions or ask clarifying questions, ill-defined PBIs can remain undetected and progress into development: *"What I often experience in Refinement Meetings … is that developers are supposed to ask me critical questions about the user stories and PBIs, but that doesn't always happen." P6*.

AI to increase efficiency by providing system knowledge and evaluating PBIs

If an AI tool has enough organizational and system context of the project, it is expected to be able to offer value to BAs by providing explanations of each system used in the project. This feature could improve BA efficiency by decreasing the time needed to learn about systems, and therefore enrich their understanding, which in turn could improve their ability to formulate requirements. Moreover, if the tool has that context, it could potentially also evaluate user stories and PBI that the BA has created based, and maybe even automate the creation of PBIs.

3.2.3 Managing dependencies

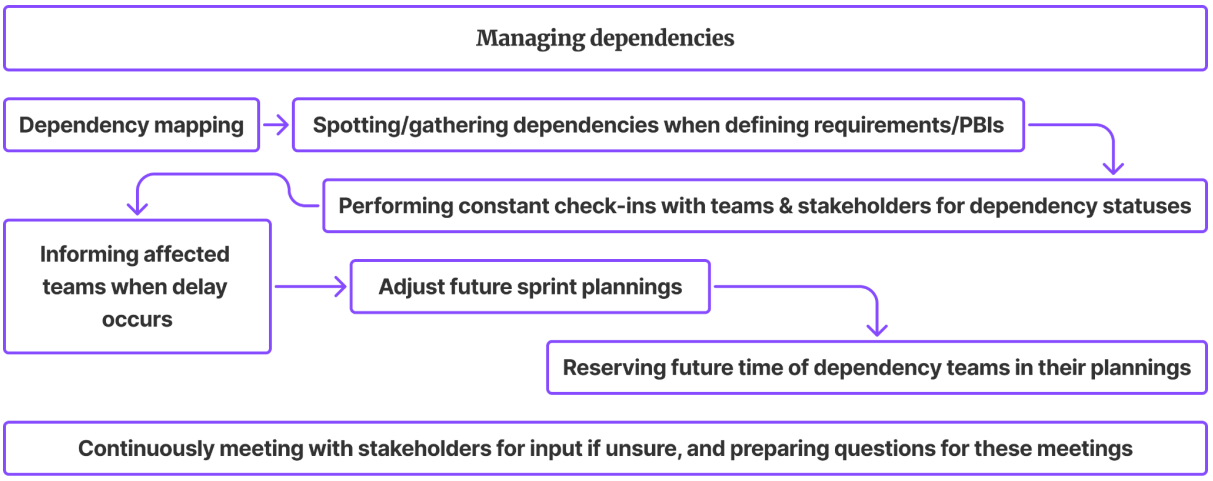


Figure 3.3 High-level managing dependencies workflow

Dependency management refers to the BA's responsibility to maintain oversight of interdependent work across teams and to prevent upstream delays from disrupting downstream planning. In practice, it involves identifying which PBIs depend on other PBIs, which BAs do by mapping out dependencies, understanding which teams own those PBIs, and continuously monitoring whether delays (are going to) occur. When delays are detected, the BA must inform the affected teams and clarify the expected impact on their planning. This early signalling enables timely re-prioritization and planning adjustments,

instead of last-minute changes once blocked work reaches the sprint. Dependency management also extends beyond the current sprint. BAs often support longer-term planning by mapping dependencies months in advance so that time of other teams can be reserved for upcoming work that is required by the BAs' teams.

To perform this effectively, BAs need sufficient system understanding to anticipate how components and back-end systems interact, and they must maintain PBI status awareness through frequent check-ins with teams. When delays occur, BAs are expected to rapidly update plans based on the latest information: *"must be able to gather enough information… to know when things are going to run into problems… and therefore have the ability to update the planning." E1*. This requires contingency thinking and the ability to formulate alternative options when dependencies slip: *"How do you deal with that dependency? And what are your plan B, C, D? ... A doesn't happen; what's your next plan? How do you fix it?" (P4)*.

Maintaining PBI status awareness is difficult

BAs mentioned they have trouble keeping track of who to meet with, especially in large-scale projects. Forgetfulness is also common among BAs (BAs feel they are not being able to keep track of to-do's): *"that was one of the struggles we faced: the more stakeholders you have, the easier it is to forget something." P5* and *"I have two development teams ... And per team there are 4–5 small mini-projects, and sometimes I just can't keep track anymore of what's going on" P6* and *"Sometimes it's not that you forget the entire dependency, but the devil is in the detail" P5*.

Difficult to obtain the necessary knowledge to be able to successfully flag dependencies

Composable commerce platforms typically consist of many interconnected back-end systems. BAs are expected to understand these interactions and systems to correctly identify and flag dependencies, yet participants indicated that fully maintaining this knowledge is unrealistic: *"There are so many different backend systems, we... need context about all backend systems... You can't know everything about every system..." P5*. As a result, dependency management is vulnerable to gaps in system-level context, particularly when dependencies hinge on detailed technical constraints, again reflecting that: *"the devil is in the detail" P5*.

PI plannings do not take room for downstream planning changes into account

As PI plannings are made when not all requirements are elicited, downstream re-planning is often needed: *"It's always just theory. That's basically the whole point of Agile ... you have to deviate from the plan again and re-prioritize." P4*. Although Agile suggests leaving room for adaptation, this flexibility is not always feasible in practice: *"You really need to be a bit flexible and sometimes you cannot really be flexible because you have tight timelines." P3*. Together, these factors make it difficult to adjust plannings in an effective manner, placing pressure on BAs.

Limited cross-team alignment and lacking ownership over their delay consequences

Dependency management is further hindered when teams operate with limited awareness of end-to-end project impact. Participants described siloed working patterns, which reduce shared accountability for dependencies and downstream consequences: *"I think the main problem is that the teams are working in silos." P3*. Misalignment can lead teams to invest effort at the wrong moment relative to overall sequencing: *"now you often see teams starting way too early on some 'options' and way too late on others" P5*. In these contexts, BAs can become perceived as the primary party responsible for delivery coordination, while other teams may not feel accountable for their dependency obligations: *"You are very often seen as the main person responsible for delivering something… and then the other teams don't take ownership to deliver their part on time." P5*. If sense of ownership can be increased among teams within the project, they might put more effort and focus on tasks that have an (stronger) effect on overall project delivery speed (i.e. PBIs that have more dependencies). This could lower delay effects, which in turn relieves the BA.

AI to support BAs in dependency management

AI has the potential to offer substantial support to BAs in addressing the challenges outlined above. First, AI could analyze project data and notify BAs which teams require attention, or even flag potential delay risks. This could strengthen BAs' sense of being in control by increasing their awareness of PBI status and emerging delays, enabling them to adjust planning earlier and helping to mitigate cascading effects on overall delivery speed. As discussed earlier, AI could also support BAs by providing relevant system knowledge, thereby improving their ability to identify and manage dependencies. In addition, AI could be used to evaluate and improve PI planning, for example by helping ensure sufficient flexibility remains for later planning adjustments. Finally, AI could support BAs by improving communication and messaging toward teams in ways that strengthen task ownership.

3.2.4 AI adoption bottlenecks and potential enablers

Collected primary data shows various AI adoption bottlenecks and potential enablers. First, data confidentiality limits possible use of open models (such as ChatGPT) which perform better than closed models (CoPilot): *"I noticed that the output from Copilot … actually weren't good enough." P4*. *"There's a limit to how much client data you can put into OpenAI… With designs, for example, I don't upload them" P4 & P5*. This issue is expected to be resolved in the near future, as Accenture is working to obtain enterprise licenses for closed OpenAI and n8n environments that employees will be able to use.

Second, BAs feel embarrassed when mistakes occur, which makes them particularly cautious about hallucinations. This concern persists even when context is provided, as hallucinated outputs can still occur: *"Sometimes ChatGPT just invents things ... I do have a chat where I write all my user stories, and it remembers the context ... even then it sometimes invents an API that doesn't exist" P6*. A tool with embedded guardrails, more clearly defined rules, and access to richer project context could reduce hallucinations and provide more reliable support: *"If there were an assistant that actually knew the scope of*

your project (which systems you work with, how things are connected, basically the full context), then it could probably recognize dependencies much better and ask critical questions." P5.

Third, many clients are not yet sufficiently prepared for AI adoption. In practice, this means they often lack both the foundational infrastructure and the C-level support needed to pursue AI initiatives and embed AI functionalities into their systems: *"If you want to go agentic AI, that also means a commitment to data … But if clients aren't there yet, you have that limitation." P4*. At an operational level, this results in access and integration issues, which were identified as a key challenge by 5 survey participants. These issues make it difficult for AI tools to obtain and work with the organizational and project context required to provide meaningful support.

Fourth, many interviewees indicate that they either lack the time, or do not prioritize making the time, to write effective prompts or configure agents that could improve their work efficiency: *"It's quicker if you just do it yourself, which means you don't make the investment because the client doesn't want it." P4. "…there I just don't have enough time." P3*. *"I ask something quickly ... it has no idea what kind of client I'm talking about ... I'm a fast, impatient prompter ... I think "okay let's do this quickly" and I give too little context ... output isn't always great." P5*. Interviewees indicate that lowering the barrier to providing the necessary context to an AI tool is highly desirable. Rather than requiring users to write extensive free-text input, they expressed a preference for more structured and accessible ways of supplying information: *"a bit like a questionnaire where you can just put in the things and then it helps you create the agents" P3*, and *"A framework where a BA can just enter certain things ... instead of a big text, that would be super chill" P4*. This suggests that structured input frameworks may make AI tools more accessible and easier for BAs to use effectively.

Fifth, BAs may struggle to provide the necessary context to AI systems when they do not yet possess that context themselves, making effective context sharing difficult. As one participant noted in response to whether providing context to AI is difficult: *"Yes ... you just need to know that from your own head" P5*. This challenge appears to be particularly pronounced during project onboarding, when BAs are still developing an understanding of the project context: *"Especially when you're new to a project ... You can't be expected to know all of that" P5*.

Lastly, BAs do not always possess the skills or understanding needed to design effective prompts or agents, nor to assess where AI can realistically add value. As a participant explains, using AI effectively requires an understanding of how to provide sufficient context, clearly describe the scenario, and sometimes even include an example to steer the output toward the intended result (P4). The same participant also emphasized that not every problem calls for Agentic AI, and that its value depends on applying it selectively and appropriately (P4). Another participant desires best-practice guidelines and example use

cases that illustrate how AI can be applied within BA workflows. This could include: *“some guidelines… sample prompts or maybe a bit of inspiration where you can still use it,”* as well as *“maybe a bit more use cases. How can we use it safely with the client” P3*. This suggests that, in addition to tooling, BAs may benefit from practical guidance that demonstrates where AI can add value and how it can be used responsibly in client contexts.

3.3 Decision: Managing dependencies

3.3.1 Weighted criteria

Expected viability: Number of tools already available refers to the extent to which suitable tools already exist for a given improvement opportunity. When few tools are currently available, this is considered positive, as it indicates a clear support gap in the area. Such a gap suggests more room for meaningful improvement and lowers the chance that the opportunity is already sufficiently addressed by existing solutions. As a result, the viability of the improvement opportunity is expected to be higher.

Expected viability: Potential impact on delivery efficiency refers to the extent to which improving a specific area is expected to contribute to more efficient project delivery. This includes its potential to positively affect factors such as delivery speed, coordination, and overall project success. Improvement areas with a greater positive influence on delivery efficiency are valued more highly, as they are expected to generate more meaningful practical benefits.

Expected feasibility refers to the extent to which it is practically achievable to design and develop a tool that can effectively support a particular improvement opportunity. It considers how easy, realistic, and technically manageable it is to create a solution for that area. Improvement opportunities for which a supporting tool can be developed more easily and with fewer barriers are assessed as more feasible. Therefore, higher expected feasibility increases the attractiveness of the improvement opportunity.

Result

Cluster	Tools available (20%)	Project impact (45%)	Feasibility (35%)	Total
Processing meetings	2	3	3	2.80
Requirements gathering & user story creation	3	4	5	4.15
Managing dependencies	**5**	**5**	**3**	**4.30**

Table 3.2: Weighted criteria result

3.3.2 Justification

Meeting processing is considered relatively low in viability for 3 main reasons. First, Microsoft Copilot is already widely used for this purpose. Second, documentation generated from meetings is not always read or used by client and project stakeholders: *"Because there is so much text, that the client or people just don't read it. That they think: 'Yeah, whatever'" P4*. This can make BAs question the value of producing such documentation in the first place: *"Often you have the discussions and it seems like it's clear... then another team is doing something else than you do. So I think that's the biggest frustration" P3*. Similarly, another interviewee states: *"What's the value of such a transcript then? It's really documentation for the sake of documentation" P4*. Third, BAs do not often make mistakes when generating documentation and time spent on the activity is relatively low: *"... I don't think you'll quickly make big mistakes." P4*, *"I don't see it often… not really that you have a meeting… that you forget like an action point… That doesn't happen that often for us." P3*, and *[How much time do you spend on meetings?] "I would say medium." P3*. For these reasons, most interviewees ranked processing meetings 3rd in viability compared with other improvement opportunities.

Developing a tool for processing meetings is also considered less feasible within the context of this project due to data-privacy regulations that restrict the use of AI in client-related environments. As one expert interviewee explains: *"With Copilot ... the moment the meeting ends, you lose your transcription... You can't prompt after the meeting… this is due to data-privacy regulations" E1*. In addition, the limited accessibility and imperfect accuracy of transcripts create further usability bottlenecks, as such a tool would depend heavily on both transcript availability and quality.

The expected viability of a tool that supports requirements gathering and user story creation is higher than that of a meeting-processing tool, although not exceptionally high. One reason is that missed requirements appear to occur relatively often. As E1 explains, missed requirements are essentially about whether a solution has been considered from all relevant angles: *"'Missed requirements' are really about: 'did you look at this from all angles?' … A pitfall for BAs: ... 'Well, I spoke to three business owners, so now I've gathered the requirements for a good tool…' ... I do see missed requirements quite often" E1*. Eliciting requirements is also difficult because stakeholders do not always articulate all relevant needs explicitly: *"Maybe the requirement wasn't missed, maybe it was never mentioned" E1*. This is problematic because incomplete requirements can result in features that are useless: *"If there isn't someone in the supply chain who also says, 'Guys, that's not possible'… then you end up building a feature that's useless" E1*. This challenge is also identified in the literature review. The expected viability of such a tool is moderated by the fact that many BAs already use AI tools for this purpose, which reduces the novelty and potential added value of a new solution in this area: *"I'm already using Copilot for PBIs to create them" P3*, and, *"I'd built an agent for user stories and at some point I had fully automated it with n8n" P4*. In addition, ill-defined requirements and user stories are often

identified during refinement meetings, which already function as an important control mechanism, even if they are not always fully effective.

In terms of feasibility, a tool that supports requirements gathering and user story creation is considered relatively high in feasibility. Several BAs indicate that they have built tools for this purpose themselves, and experts E2 and E3 also identified it as a suitable use case for AI.

Viability for a tool that offers support in dependency management is high because no existent tools have been found during the research: *“AI is not used for dependency mapping as far as I know.” E1*, moreover, dependency management directly affects product delivery speed: If done well, teams can adapt task priority and planning to dependency delays occurrences, and optimal delivery is achieved: “Flagging dependencies on time ensures… optimal delivery by the team executing the work.” E1. If done bad, teams might take their own initiative and start tasks without being aligned with other teams’ plannings in the ART, potentially causing even more delays, or they might not pick up any task at all: *“now you often see teams starting way too early on some ‘options’ and way too late on others” P5*, and delivery delays directly impact costs, success and reputation of BAs and Accenture: *“[Not flagging dependencies correctly] immediately creates blockers, and that instantly causes a decrease in delivery speed. That’s important for both us as Accenture and for the client, because you’re judged on that…” E1*. However, feasibility of dependency management is low because it is considered dependent on a lot of data. Overall, dependency management has the highest combined score as can be seen in table 3.2, and therefore that opportunity area is considered the opportunity to design the tool for.

The expected viability of a tool that supports dependency management is considered high, as no existing tools for this purpose were identified during the research: *“AI is not used for dependency mapping as far as I know” E1*. This suggests a clear gap and, therefore, a strong opportunity for value creation. In addition, dependency management directly affects product delivery speed. When dependencies are identified and flagged in time, teams can adjust task priorities and planning in response to dependency-related delays, thereby supporting more optimal delivery: *“Flagging dependencies on time ensures… optimal delivery by the team executing the work” E1*. When dependency management is handled poorly, teams may start work on tasks without being properly aligned with the planning of other teams within the ART, which can create further delays and inefficiencies: *“now you often see teams starting way too early on some ‘options’ and way too late on others” P5*. In some cases, teams may also hesitate to start work at all because dependencies remain unclear. These delivery delays have direct consequences for project costs, delivery performance, and the reputation of both BAs and Accenture: *“[Not flagging dependencies correctly] immediately creates blockers, and that instantly causes a decrease in delivery speed. That’s important for both us as Accenture and for the client, because you’re judged on that…” E1*.

At the same time, the feasibility of a dependency-management tool is considered relatively low, as effective support in this area is considered to depend on access to large amounts of relevant and reliable project data according to interviewees. Nevertheless, dependency management receives the highest combined score, as shown in table 3.2. It is therefore selected as the most promising opportunity area for the design of the tool.

3.4 Design brief

3.4.1 Design goal

Design and validate an Agentic AI-driven project management tool in n8n that supports BAs by achieving one or more of the following objectives: (A) increasing team alignment and task ownership across project teams, (B) improving PI planning by ensuring sufficient room remains for downstream planning adjustments, giving BAs more room to create new plannings based on dependencies, relieving them from stress, (C) reducing the number of forgotten BA tasks and strengthening BAs' sense of being on track, (D) improving BAs' system and dependency understanding by making it easier to learn about relevant systems, (E) reducing the time required to adjust sprint planning in order to limit cascading delay effects, and (F) improving BA outputs, such as plannings, PBIs, and requirements, by evaluating them in relation to dependencies.

What it will be:

1. A working prototype developed iteratively through feedback sessions with BAs.
2. A survey accompanied by a prototype demo video distributed to assess BA desirability and expected viability using specific evaluation criteria based on the tool's exact value proposition and expected efficiency improvements: *"A lot of folks want to understand how is your agent better or what is your agent doing? Like what is your evaluation criteria." E3"*.
3. An adoption roadmap created to outline how Accenture could deploy and further develop the tool, as well as other Agentic AI-driven tools, over time to improve BA workflows.
4. A how-to-use guide created in the form of a scenario story board and demo video enabling BAs to understand how the tool should be used within their workflow.
5. A guidebook or framework created to capture and explain the key lessons learned during the prototyping of the Agentic AI tool, which BAs can use when designing Agentic AI-driven workflows.

3.4.2 Design requirements

R1: The tool should be designed to enable efficient project-context capture, so that using the tool does not become time-consuming for BAs.
R2: The tool should be designed around minimal AI use and maximum BA benefit, as API calls and token consumption increase computational costs and should therefore be limited. This implies aiming for the smallest possible set of AI-driven actions that still delivers clear value, rather than applying AI for its own sake. As E3 noted: *"These are all API calls. And*

each API call … If you think from a client point of view, and making it compete … means … 12 API calls between these 3 agents, … it's a huge cost".

R3: The tool should be designed to work primarily with tools and files that are relatively easy to integrate and that do not contain excessive sensitive or non-shareable company data. Examples include Jira, Confluence, and Excel files, as these are generally uploadable or accessible through APIs, making them more suitable for integration into the tool.

R4: The tool should be designed to keep the "AI chain" as short as possible in order to minimize hallucination risk and reduce the loss of important information across multiple AI-processing steps. As E1 explained: *"I also often see people … letting AI process the information 10 times, … important details get filtered out and aren't included anymore"*.

R5: The tool should be designed to augment BAs rather than fully automate their work, as excessive automation may reduce opportunities for BAs to develop essential domain and professional skills. As E1 noted: *"The biggest concern is that a BA no longer learns what a product is or does because everything becomes automated, and the BA themselves doesn't become a specialist in the job"*.

R6: The tool should be designed around specialized agents, each responsible for a specific task, as the literature indicates that this improves performance, scalability, and maintainability.

R7: The human–tool workflow should be designed such that a HITL check is always in place before any output is shared with client or project stakeholders. As E2 emphasized: "Let's take that as the big title: Human has a final say".

R8: The tool should be designed to reflect Accenture's brand tone. As E2 noted: "Sticking to brand tone, ... or brand guidelines are important to adhere to as well".

4. Develop

This chapter presents the Develop phase, in which the selected opportunity is translated into solution concepts and prototypes. Through exploration, conceptualisation, and rapid prototyping, two concept directions, Jumpstart and Risko, are developed and iteratively refined based on feedback. The chapter concludes with the main prototyping learnings and the justification for taking Risko forward into the Deliver phase.

AI used in this chapter:

- ChatGPT as a co-design companion to develop the prototype in n8n
- ChatGPT and Google Gemini for story board creation support
- ChatGPT to support formatting text and improving text

4.1 Exploring and ideating

4.1.1 How might we brainstorm

To explore potential tools, another “how might we …” Brainstorm is set up to stimulate divergent solution thinking. In total, 15 ideas are formulated and can be found in table 4.1.

Objective A:	A1: Show visualizations of dependency delay consequences to the respective teams by sending these to them or adding these to the PBIs in Jira.	A2: Give clearer prioritization or task sequence instructions to teams.	
Objective B:	B1: Evaluate created PI plannings and provide improvement suggestions based on dependencies	B2: Create PI planning drafts with dependencies in mind	B3: Support in PI planning by visually mapping dependencies of selected/provided tasks
Objective C:	C1: Remind BAs to perform check-ins with stakeholders based on PI or Sprint planning/requirements	C2: Remind BAs to instruct teams to work on particular tasks.	C3: Notify BAs when a delay is occurring or likely to occur.
Objective D:	D1: Educate BAs on the back-end systems within a platform: What does this back-end system do within this platform? What teams are the owner? How do they work? What is the documentation?	D2: Notify/educate BAs on the sprint velocity and team capacity of each of the teams within the project	
Objective E:	E1: Create sprint planning drafts with explanations	E2: Analyze PBIs and estimations and analyze sprint planning to provide suggestions to swap out particular tasks with other PBIs with explanations.	
Objective F:	F1: Evaluate user stories: Do	F2: Evaluate feature	F3: Evaluate sprint planning:

	these user stories cover all dependencies? What is this user story dependent on? --> Is that all correctly covered in this user story?	requirement lists: “So we are building this feature and we created this requirement list. Please evaluate this. Do we miss any dependencies (=requirements)?	This is the sprint planning we created. Is this good? How can it be improved? --> tool looks at team capacity, team velocity, task sizing and planning to check for mismatches and gives suggestions for improvements, lists potential risks, missed things etc., based on dependencies.

Table 4.1: 15 formulated tool functionality ideas

4.1.2 From ideas to concepts

To narrow down to ideas to create concepts of, the 15 ideas are positioned in a C-box matrix. The y-axis represents the researcher’s confidence in viability (i.e., likelihood of improving BA efficiency), and the x-axis represents the researcher’s confidence in feasibility (i.e., likelihood that the idea could be built within the remaining project period). Ideas are mapped based on the researcher’s best judgment, informed by the understanding and empathy developed during the Discover and Define phases of the project.

Selection focuses primarily on the upper-right quadrant (high viability, high feasibility). In addition, F2 is considered because it is aligned with D1: If D1 (educating on requirements and dependencies) could be implemented quickly, F2 (evaluating requirement lists using system and dependency knowledge) could serve as a feasible extension.

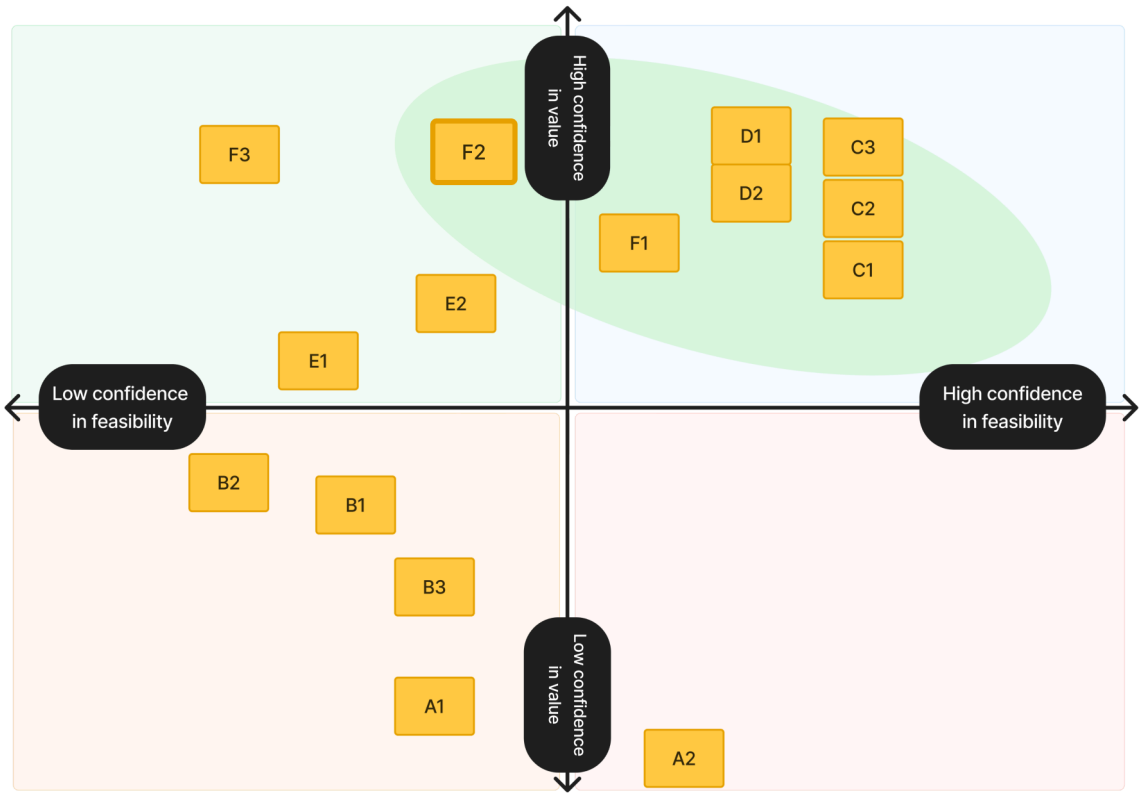


Figure 4.1: C-box used to select 7 ideas to develop further into concepts

As shown in figure 4.1, only ideas from objectives C, D, and F are assessed as both highly valuable and feasible. Objective A is not prioritized because the underlying causes of low PBI ownership and delays are not sufficiently understood. As a result, the researcher can not confidently argue that A-type solutions will address the root problem. Objective B is excluded because BAs are not primarily responsible for PI planning. In addition, the key stakeholders who shape PI plans (e.g., POs and business stakeholders) are not extensively interviewed or observed, making it difficult to assess the viability of objective B ideas. Objective E is deprioritized due to lower perceived value and feasibility. Generating full sprint-planning drafts is considered “overkill,” since BAs mainly need to swap or reprioritize a limited set of items when delays occur. While E2 (supporting targeted swaps) is viewed as more valuable and feasible than generating complete drafts, it still scores substantially lower than C, D, and F and is therefore not selected.

4.2 Conceptualization

4.2.1 Initial concepts

In total 7 concepts are created. Each concept has a description of its value proposition, a story board showing how the tool will work and what the experience for the BA will be when using the tool, as well as a high-level technical overview of what data/integrations the tool would need in order to function. Only 1 storyboard and technical overview is shown in figures 4.2 and 4.3 as an example. All of the concepts’ storyboards and overviews can be found in Appendix H. Only the value propositions of each of the 7 concepts is presented in the report.

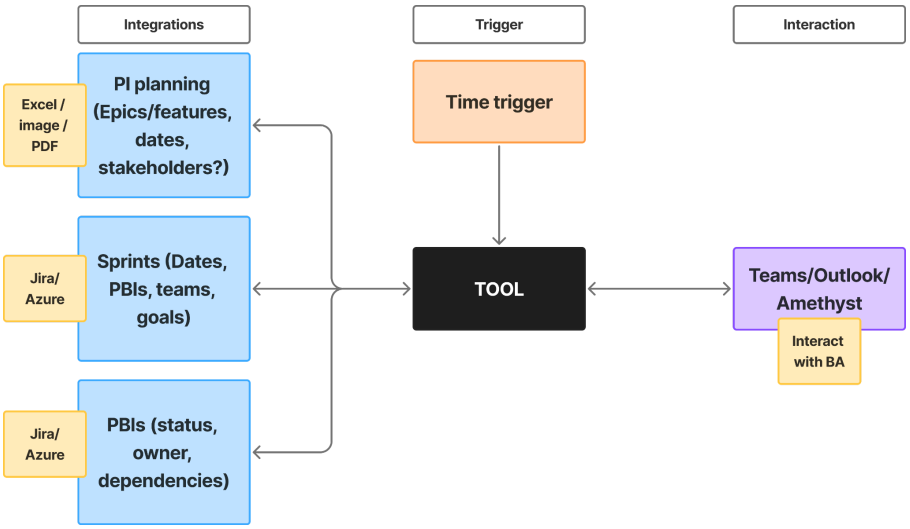


Figure 4.2: Example of technical overview of concept

Figure 4.3: Example of storyboard of concept

C1: Dependency Check-In Reminder

This proactive notifier alerts BAs when a relevant upcoming dependency, such as an epic or feature owned by another team, is scheduled in the PI planning and approaching in the coming months. It reminds the BA to initiate a timely check-in with the owning team to secure capacity and ensure that the required PBIs are planned on time. **VP:** By preventing these future-oriented check-ins from being overlooked, the tool strengthens the BA's sense of being on track, which may help reduce stress, while also improving dependency readiness and thereby helping to reduce delays and support faster delivery.

C2: Sprint-PI planning alignment Notifier

This sprint-alignment notifier alerts BAs when a sprint plan for one of their teams is misaligned with the PI planning, for example when expected PI-related tasks are missing from upcoming sprints. It prompts the BA to discuss the gap with the team and adjust future sprint planning where possible. **VP**: By identifying planning issues early and enabling earlier alignment between sprint and PI planning, the tool strengthens the BA's sense of being on track, which may help reduce stress, while also reducing the risk of late rework or last-minute replanning, thereby supporting faster delivery.

When comparing C1 and C2, C1 has an outside-in orientation, as it focuses on limiting the negative effects that other teams may have on the BA's teams. In contrast, C2 has an inside-out orientation, as it focuses on limiting the negative effects that the BA's teams may have on other teams.

C3: Delay Notifier
This delay-risk notifier alerts BAs when a dependency relevant to their team(s) is unlikely to be completed within the current sprint. It provides a prioritized overview of likely delayed items, enabling the BA to (1) check in with the responsible teams to ensure sufficient focus is placed on the dependency and/or (2) proactively reprioritize their own sprint planning to mitigate downstream impact. The same analysis can also be applied to the BA's own teams, generating a concise list of at-risk items that can be used as input for stand-ups or status discussions and to inform impacted teams. **VP**: By translating early delay signals into targeted follow-up actions, the tool strengthens the BA's PBI status awareness and sense of being on track, which may help reduce stress, while supporting faster adaptation to emerging delays and reducing missed check-ins.

D1: System Educator
This system educator assistant integrates with the client's Confluence space, where system, team, and feature documentation is stored, and allows BAs to ask questions through Microsoft Teams, such as "For feature X, which systems are likely involved?" or "How does the CRM system work?" It uses client documentation as RAG to return client-specific answers, including likely involved systems, the rationale behind them, and key open points or considerations per system. **VP**: By making client-specific system knowledge easier and faster to access, the tool helps BAs build system understanding with less effort, increasing confidence and reducing the time needed to get up to speed. In turn, this could support earlier dependency identification, better-prepared stakeholder conversations, and higher-quality requirements work.

D2: Velocity and Capacity Notifier/Educator
This sprint-planning assistant supports BAs by analysing future sprint risk on the basis of team capacity, historical velocity, and planned story points. With access to team-availability Excel files and Jira, it can run these analyses automatically on a time trigger or on demand when requested by the BA. In addition, BAs can query planning scenarios, such as "How long will Team X take to complete PBI X if planned in sprint Y?", after which the tool returns an estimated duration in days together with a transparent explanation of the underlying calculation. **VP**: By making the consequences of capacity constraints and planning choices more explicit, the tool helps BAs create more realistic sprint plans with greater confidence and less manual effort. In turn, this reduces planning errors and may support faster delivery.

F1: User Stories Evaluator
This user story reviewer evaluates BA-authored user stories and flags quality issues such as formatting problems, misspellings, and potential API inconsistencies. It also assesses

stories against known system and dependency information by examining whether key dependencies are addressed and what each story depends on. When a BA uploads user stories, the tool retrieves relevant system, team, and API documentation from Confluence and returns feedback with concrete improvement suggestions. **VP**: By helping BAs identify issues and strengthen their user stories with less manual checking, the tool increases confidence in the quality of their output and supports stronger system and dependency understanding. In turn, this can improve requirements quality and reduce rework.

F2: Requirements Evaluator
This requirements evaluator reviews a BA-created requirement list and can flag issues, identify open points, and assess the listed requirements in relation to dependencies, for example by evaluating whether relevant dependency-related requirements may be missing. When a BA uploads a requirement list, the tool analyzes related system, team, and API documentation from Confluence and returns targeted improvement suggestions. **VP**: By helping BAs check and improve their requirements with less manual effort, the tool increases confidence in the quality of their output and supports a stronger understanding of relevant systems and dependencies. In turn, this can improve BA output and reduce the likelihood of missed requirements.

4.2.2 Co-creation session

Next to the researcher's own conceptualization efforts, 6 BAs participated in a 1-hour co-creation workshop in which they developed their own concepts for an Agentic AI-driven project-management tool intended to support them in addressing challenges related to dependency management.

During the workshop, participants were presented with the challenges underlying objectives C, D, and F and were then invited to develop concepts by responding to a series of open-ended questions. The workshop was structured using elements of the 1-2-4-All framework (Liberating Structures): participants first worked individually, then in pairs, and finally shared their ideas with the full group. In addition, the workshop questions were designed to stimulate creativity through the use of "magic wand" scenarios, as well as prompts inspired by the 15% Solutions (Liberating Structures) and the Scamper framework. These were used to encourage participants to think beyond existing constraints and generate a broader range of ideas, particularly regarding the data sources and tool integrations required for such a tool to deliver value.

All participants provided informed consent for the use of the workshop material they created in Miro for the purposes of this project. The data is anonymized and images of the Miro board can be found in Appendix I.

Participant	Years at Accenture	Years as BA
P3 (BA)	4	4

P5 (BA)	1	1
P7 (BA)	1	1
P8 (BA)	5	8
P9 (BA)	0	0
P10 (BA)	4	4

Table 4.2: Participants of co-creation session - figure 4.4: Co-creation workshop at Accenture

During the workshop, four concepts are generated. (1) **Plani** supports BAs in creating sprint and PI plans, as well as PBIs, using inputs such as roadmaps, Excel files, and Jira data. BAs described this concept as “exciting.” However, the concept largely overlaps with idea E1 and was considered to have relatively low feasibility. In addition, it would require the BA to review and quality-assure an entire planning board, which would involve substantial effort. (2) **Risko** supports BAs by analysing sprint backlogs and PI planning to flag risks, dependencies, and relevant considerations. BAs expected this concept to make them feel “relieved.” Risko overlaps with concept C3 (Delay Notifier), but extends it by also including dependency detection. Implementing both dependency detection and risk flagging would require strong system-understanding and risk-calculation or simulation capabilities, which is considered challenging within the project timeframe. Excluding dependency detection, however, is considered more feasible. (3) **Jumpstart** is identical to concept D1 (System Educator) and is aimed at helping BAs better understand the system landscape. BAs indicate that this concept would make them feel more prepared and efficient. (4) **CAS** is a meeting assistant that joins sessions, produces summaries, runs analyses, and suggests actions during or after meetings. This concept is considered out of scope, as meeting processing had already been excluded earlier in the project.

4.2.3 Deciding on the concept

The co-creation session participants’ votes on their favorite concepts during the session indicate they desire the Risko (C3 + D2) and Jumpstart (D1) concepts most. While this influences the decision, additional criteria influence the eventual decision for which concept(s) to further develop into prototype(s).

Concept	Feasibility (1 to 5)	Viability (1 to 5)	Project-context accessibility (1 to 5)	Avg.
C1	3	3	5	3.67
C2	3	3	5	3.67
C3 (Risko)	**5**	**5**	**5**	**5.00**
D1 (Jumpstart)	**5**	**5**	**5**	**5.00**
D2 (RIsko)	**5**	**5**	**5**	**5.00**
F1 (Jumpstart+)	4	5	4	4.33

F2 (Jumpstart+)	4	5	4	4.33

Table 4.3: Concept scores

Consolidation into two main concept directions

C3 + D2 (+ potentially E2) → Risko

For C3 to function effectively, the ability to retrieve and calculate team capacities and historical sprint velocities is essential. To flag risks accurately, Risko must be able to estimate how much work a team is realistically able to complete within the remaining time of a sprint. This requires the tool to calculate available team capacity and compare it with historical velocity in order to identify PBIs or sprint plans that are likely to cause delays.

Once Risko flags a risky PBI or sprint, it could also analyse the broader product backlog and suggest a suitable PBI to remove, swap, or add to the sprint backlog. This functionality aligns with idea E2 and would further increase the concept's value for BAs. As such, the eventual Risko concept combines elements of C3, D2, and potentially E2.

To enable these functionalities, the required project context consists primarily of the Excel file in which team availability is captured and an integration with Jira to access sprint and backlog data. Both of these sources are considered relatively easy to share with and integrate into the tool.

D1 → Jumpstart (with F1/F2 as post-MVP extensions)

F1 and F2 build directly on D1's system-analysis capability, but additionally require strong interpretation of user stories and requirements, as well as high-quality examples to support reliable evaluation. This makes F1 and F2 less feasible than D1. It also makes the necessary project context more difficult to obtain, since not every project uses the same format for PBIs, user stories, or requirements. As a result, additional functionality would likely be needed to first structure these artefacts according to the project-specific format, which further increases complexity and reduces context accessibility. Therefore, F1 and F2 are treated as additional features or extensions built on top of the D1 concept. D1 itself is considered relatively feasible, as it primarily requires an integration with Confluence, where documentation, project descriptions, and team information are already available.

Why C1 and C2 are deprioritized

C1 and C2 are both considered attractive concepts. However, primary data indicates that PI plans are often relatively static and are not updated continuously, whereas requirements and sprint plans tend to change frequently during delivery. As a result, the concept would only be effective if PI plans were kept up to date on an ongoing basis. Since automating such updates is considered too complex within the project timeframe, the feasibility of these concepts is reduced.

Implication for prototyping

These scores and concept combinations point to two distinct and valuable concept candidates for the tool. At this stage, the relative feasibility and viability of Risko versus Jumpstart cannot yet be determined analytically and instead require prototyping to be assessed properly. Therefore, the first prototyping iteration focuses on developing initial versions of both concepts.

4.3 Prototyping

During prototyping, the rapid prototyping framework is used to quickly build, validate, and refine the concepts. In addition, a YouTube tutorial is consulted to support development in n8n (YouTube, n.d.). To formulate agent system messages, which define the role and task of an agent, and user messages, which are prompts sent by the user or another node or agent in n8n, the KERNEL framework is used (Reddit, n.d.). This framework is intended to improve first-try success rate, answer speed, consistency across prompts, and token efficiency.

n8n (n8n, n.d.) is selected as the primary prototyping platform because discussions with client-side stakeholders indicate that an Accenture n8n license is likely to become available in the near future, creating the possibility for reuse of the prototype beyond the thesis project. In addition, n8n offers a practical balance between ease of learning and technical flexibility. It supports transparent workflow-based automation, enabling the prototype to be built as a sequential workflow that executes the steps required to generate an output, such as answering a BA question in Jumpstart or composing a notification in Risko. n8n also provides built-in nodes for custom code, email delivery, and spreadsheet-data retrieval, making it suitable for implementing tool use and RAG-based behaviour within the prototype.

Because an enterprise n8n license and an Accenture LLM license are not available during the prototyping phase, the prototype is built using personal n8n and OpenAI accounts. As a result, client data cannot be used. To address this limitation, a mock project scenario is created involving a company (“Dyson”) developing a spare-parts B2B commerce platform for professional cleaning companies. Mock data inspired by real project structures is created and validated through discussions with participant E1. Moreover, to emulate project-tool integrations, the prototype uses a personal Google Sheets environment as a mock knowledge base and as a stand-in for Confluence, Jira, and Excel. The sheets are structured to mimic Jira and Confluence data, including PBIs, sprints, teams, and documentation references. Finally, the n8n chat interface is used as a stand-in for Microsoft Teams as the interaction channel.

During prototyping, 6 individuals provide feedback on 1 or both prototypes in sessions lasting 30 minutes to 1 hour. This feedback is used to refine the designs. For example, Risko is initially set to run daily, but BA feedback indicates that weekly notifications are more usable, as PBIs are often finalized near the end of the week. Notes from these sessions are included in Appendix J.

Participant	Role/expertise	Experience at Accenture
P4	Management Consultant	3 (considered an Agentic AI expert among ASC. Hosts Agentic AI training sessions)
P5	Digital Business Analyst	1
E1	Digital Business Consultant	4
P7	Management Consulting Analyst	1
P8	Digital Strategy Delivery Consultant	5
P11	AI & Data Accenture Song Intern	0 (builds a lot of tools in n8n)

Table 4.4: Participant list of prototype feedback discussions

4.4 Jumpstart

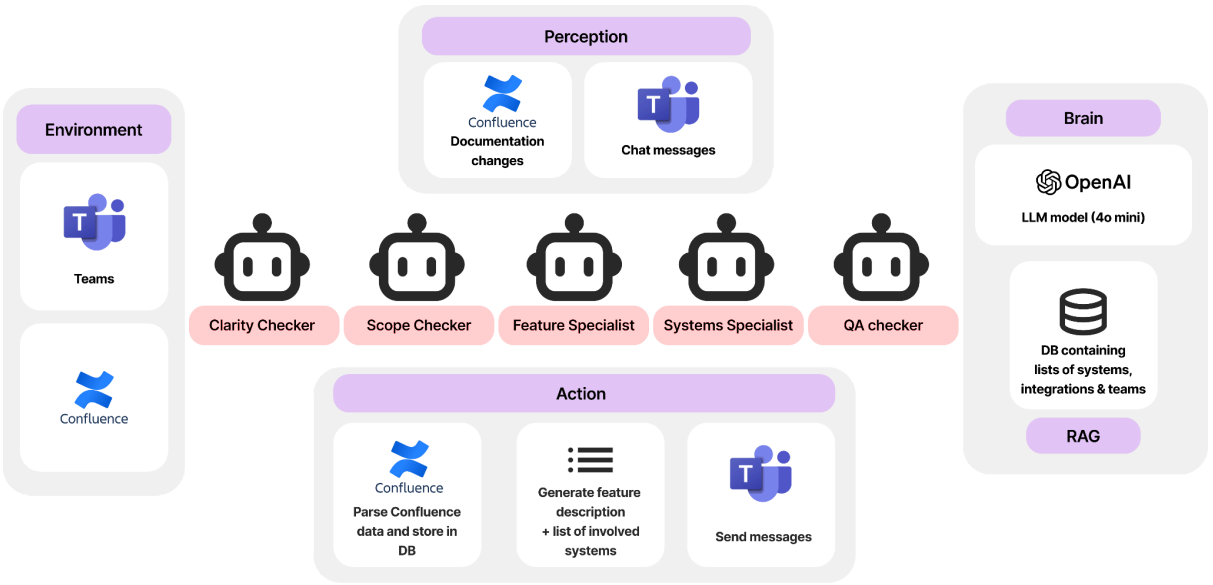


Figure 4.5: Conceptual overview of Jumpstart

Jumpstart is essentially a chatbot composed of 5 specialized AI agents: (1) Clarity Checker, (2) Scope Checker, (3) Feature Specialist, (4) Systems Specialist, and (5) QA Checker. Together, these agents answer BA questions in a usable format about systems, involved teams, and system integrations within the BA's project. To support these responses, Jumpstart uses a knowledge base as a form of RAG, containing lists of systems with relevant information about each system, project teams, and integrations between systems.

Jumpstart builds on the Agentic AI capabilities identified in table 2.1, including receiving messages, reasoning and planning actions through question scoping, and autonomously performing tasks and using tools, such as retrieving and analysing Confluence data, generating textual content and sending messages.

4.4.1 Jumpstart scenario

1. BA is working on a feature and needs to spot dependencies, but doesn't understand the different systems in the commerce platform

2. BA needs to check-in with technical stakeholders, but what questions to ask?

3. BA can search for documentation, but this is like searching for a needle in a haystack: Very difficult and time-consuming

4. BA feels stressed, overwhelmed, clueless

5. BA can now use Jumpstart

6. BA explains the feature and asks which systems are involved.

7. Jumpstart provides an answer containing open questions with regards to the feature, and a list of involved systems with explanations why

8. BA feels relieved and confident about finding the dependencies for this feature.

Figure 4.6: Jumpstart scenario story board

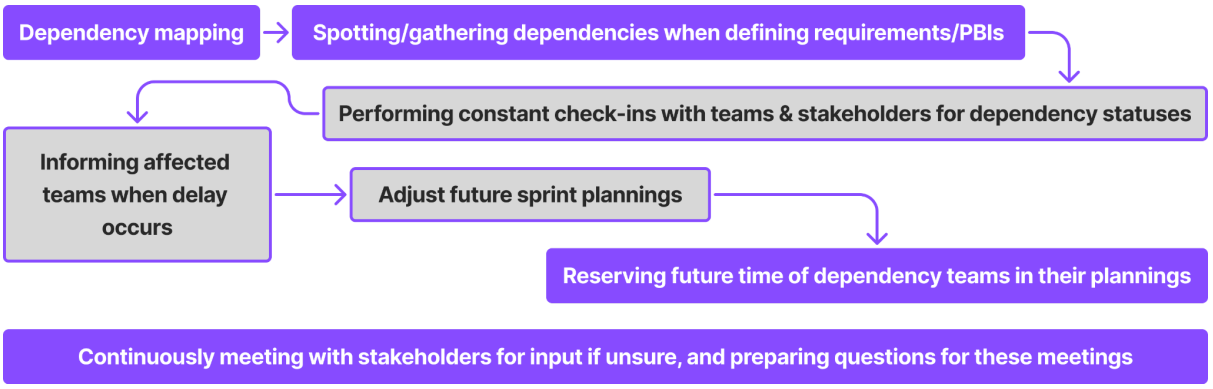


Figure 4.7: Jumpstart addresses the purple parts of the managing dependencies workflow

4.4.2 Jumpstart BA interaction

Jumpstart is built according to the Instructor–Executor human–AI collaboration paradigm (Xi et al., 2023), in which the BA instructs the tool through a chat message. In the prototype, this takes place via the n8n chat interface (figure 4.8), although the intended application is a dedicated Microsoft Teams chat. Whenever a BA is uncertain about a dependency or requirement, the tool can be queried.

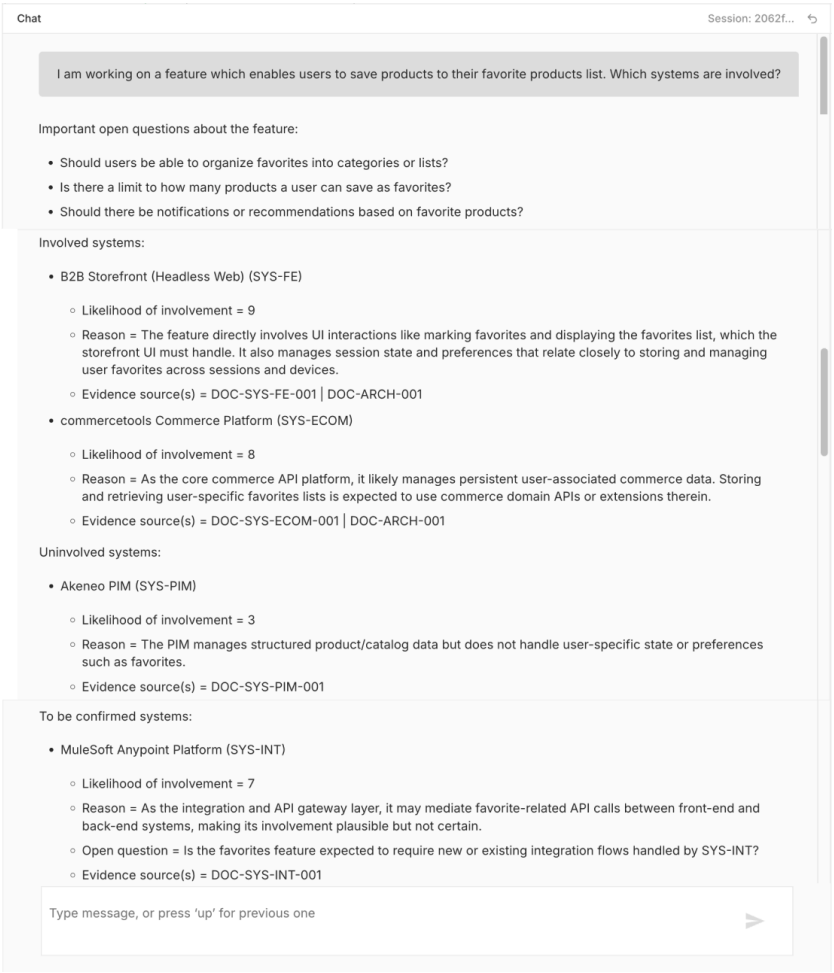


Figure 4.8: Jumpstart answer to BA question

4.4.3 How Jumpstart works

Data input	Tool tasks	BA-facing output
- **API documentation** (how API works, dependencies) - **Organization chart** (teams, systems, responsibilities, relations/dependencies) - **System architecture diagrams** (back-end systems, dependencies) - **BA question**	- **Parse Confluence data to create database** (containing lists of systems, integration's, teams and ownership) - **Analyze BA question** - **Use database as RAG** - **Generate answer** - **Send message via Teams**	- Answer to question containing: 1. **Explanations** 2. **Assumptions made by tool** 3. **Open questions** 4. **Source references**
Confluence	RAG, AI agents	(figure 4.8)

Figure 4.9: High level overview of how Jumpstart works

When a BA submits a question, the Clarity Checker first assesses whether the question is sufficiently clear for the tool to process. If not, the Clarity Checker generates a question that if answered by the BA, clarifies the question. If the question is clear, the Scope Checker determines which scope category the question falls into. At present, Jumpstart supports only one scope type: Feature → Systems. This means it can answer questions in which a

BA describes a feature and asks which client systems are likely to be involved in its development.

Next, the Feature Specialist analyses the user message and formulates its own interpretation of the feature. It generates assumptions and open questions, and produces a more detailed feature description. This description is passed to the Systems Specialist, which uses the knowledge base as a RAG source to identify which systems are likely to be involved. It returns a list of systems with a likelihood level, an explanation of why that level is assigned, and the supporting evidence source(s). These evidence references are intended to eventually become clickable Confluence links, allowing BAs to verify the output and explore the source material further. When the Systems Specialist is uncertain, it also generates a follow-up question for the BA to help confirm or rule out that system's involvement. To improve consistency and interpretability, the Systems Specialist is instructed to assign a likelihood level to each system and determine involvement accordingly. Table 4.5 shows the meaning of each likelihood level and the corresponding action.

Level	Meaning	Systems Specialist Action
1-3	System involvement is unlikely.	Do not consider this system.
4-7	System involvement is uncertain.	Generate 1 targeted question that if answered, would confirm/refute system involvement.
8-10	System involvement is likely.	Consider this system as involved.

Table 4.5: Likelihood levels of system involvement used by the Jumpstart Systems Specialist

Finally, the QA Checker verifies whether the generated feature description and system list together answer the BA's question and whether the mentioned systems and evidence sources actually exist in the knowledge base and Confluence. If not, it instructs the relevant agents to revise their output accordingly.

4.4.4 Jumpstart's expected impact (value proposition)

Jumpstart is designed to help BAs identify dependencies and requirements when defining features or working with specific systems, thereby addressing objective D. The Feature Specialist not only interprets the BA's question and prepares it for the Systems Specialist, but also acts as a sparring partner by producing a more detailed feature description together with assumptions and open questions. This gives the BA a useful starting point for reflection and for follow-up discussions with relevant stakeholders. In turn, the Systems Specialist educates the BA by explaining why systems are considered involved or not.

Through these features, Jumpstart supports more targeted and project-specific learning about systems and dependencies, while also allowing BAs to ask questions whenever needed instead of having to wait for input from (technical) stakeholders. By making BA

learning more efficient and improving system and dependency understanding, Jumpstart strengthens BAs' ability to identify dependencies and elicit requirements, thereby improving their dependency-management capability. This can positively affect project delivery, while also increasing BA confidence and, as indicated in the co-creation workshop, their sense of relief.

4.4.5 Jumpstart and the design requirements

R1 is not met (yet). Jumpstart is intended to integrate with Confluence to parse, normalize, and interpret documentation such as API descriptions, organizational charts, team descriptions, and system architecture diagrams, and use this to build its own knowledge base of systems, integrations, teams, and their relationships. The agents can then use this knowledge base as RAG when answering BA questions. In the current prototype, however, this knowledge base is pre-built for the mock project and is not generated or updated automatically, as the focus is on answering BA questions rather than on parsing and structuring project data. Although automatic knowledge-base creation is essential for real-world usability across projects, it was considered too complex to implement within the project timeframe and is therefore left for future development. As a result, Jumpstart cannot yet be concluded to meet requirement R1.

R2, R4, and R6 are met. Jumpstart uses specialized agents, each with a distinct and relevant task that does not overlap with the others in the system, and AI is applied only where it adds clear value. In addition, the "AI chain" is limited to two agents when generating an answer to a BA question, helping to keep API calls and token usage as low as possible.

R3 is met. Jumpstart requires only Confluence and Microsoft Teams as integrations, and since plugins or tools for these integrations are already available, this is considered feasible. However, Jumpstart does depend on the client having the necessary project data available in Confluence or in a comparable environment, such as OneDrive or SharePoint. Only under that condition can the tool function effectively.

R5 is met. Jumpstart is designed to augment rather than automate BA work by helping BAs improve their system understanding and giving them a "jumpstart" in identifying dependencies and defining technical requirements. It therefore supports BAs in performing these activities more effectively, rather than replacing them.

R7 is met. Jumpstart incorporates a HITL mechanism by generating BA-facing rather than client-facing output, ensuring that the BA remains in control of how the output is interpreted and used. In addition, the included evidence sources allow the BA to verify the tool's output.

R8 is currently not applicable. Because Jumpstart uses the n8n chat interface, which offers no room for visual customization, adherence to brand guidelines is not yet relevant.

Once Jumpstart is embedded in Microsoft Teams, however, brand alignment could be incorporated through the use of an appropriate profile image and Accenture-consistent visual elements. In addition, the agents could be instructed to generate responses in line with Accenture's tone of voice. Although this has not yet been implemented, it is considered feasible.

4.5 Risko

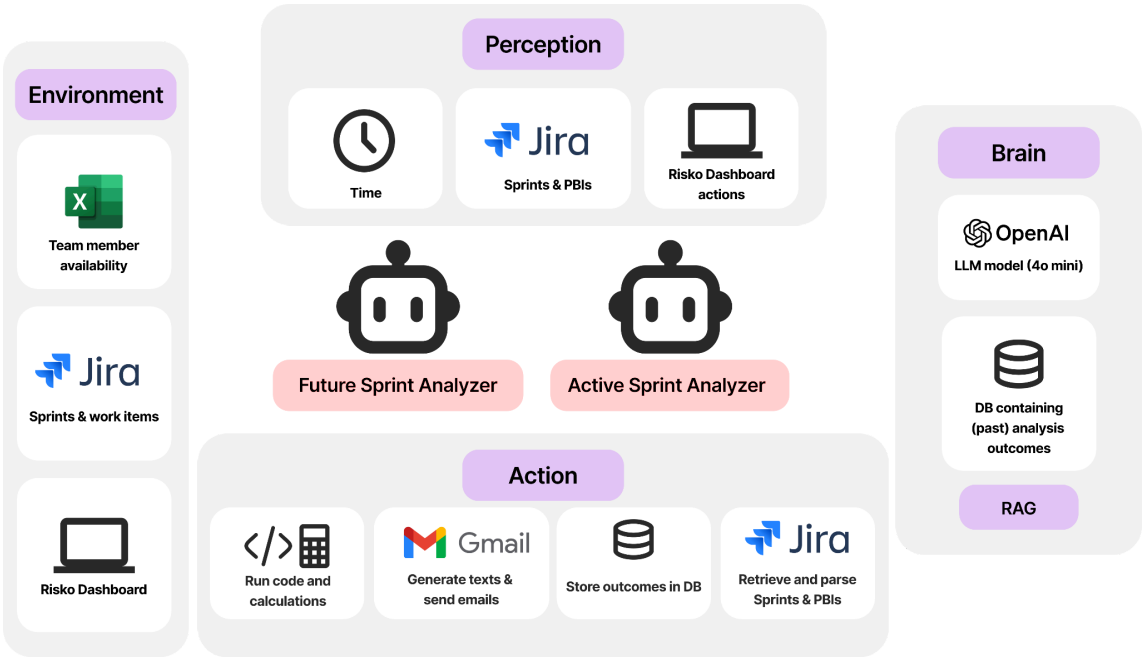


Figure 4.10: Conceptual overview of Risko

Risko is essentially a BA notifier powered by two connected n8n workflows: (1) the **Future Sprint Backlog Risk Analysis workflow** and (2) the **Active Sprint PBI Delay Likelihood Analysis workflow**. Together, these workflows and their AI agents send notification messages containing (1) lists of risky and opportunity future sprints, along with suggested add or remove actions that can serve as input for planning discussions and backlog refinements with teams, and (2) prioritized lists of PBIs that are likely to cause delays, both within the BA's own teams and in teams on which those teams depend, accompanied by recommended BA check-in actions. Together, these workflows enable BAs to initiate targeted check-ins with both their own teams and other teams within the project, encourage prioritization of high-importance PBIs, proactively inform relevant stakeholders, and adjust future sprint backlogs to mitigate cascading delay effects. To perform these analyses, the tool integrates with Jira, accesses team-availability Excel sheets, and is configured through its dashboard.

Risko builds on the Agentic AI capabilities identified in table 2.1, including perceiving time and getting triggered, and autonomously performing tasks and using tools, such as

retrieving and analysing Jira, Excel files, running calculations, generating textual content and sending emails.

4.5.1 Risko scenario

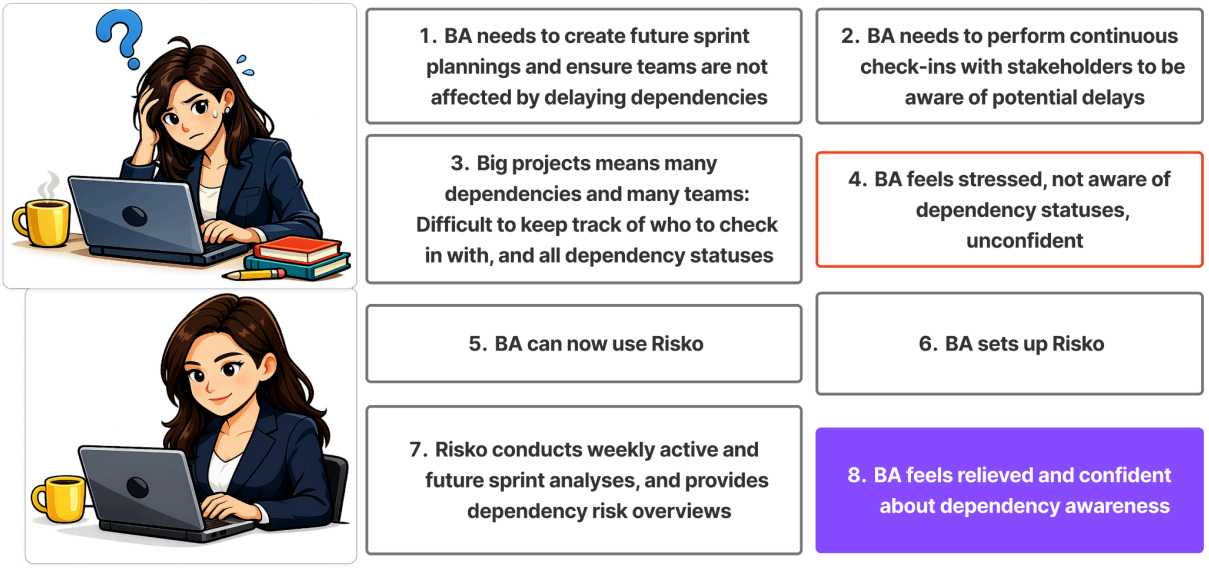


Figure 4.11 Risko scenario storyboard

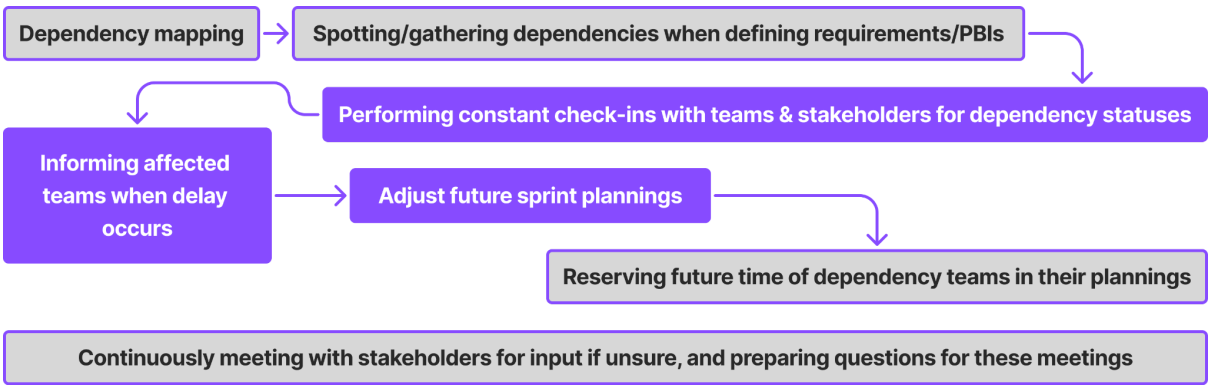


Figure 4.12: Risko addresses the purple parts of the managing dependencies workflow

4.5.2 Risko BA interaction

Risko is built according to the Equal-Partnership human–AI collaboration paradigm (Xi et al., 2023), in which the BA and the tool, together with its AI agents, collaborate as peers. Within this setup, the tool provides the BA with suggested action items through its notifications. In the current prototype, this interaction takes place only through Outlook email, although the intended application is a dedicated Microsoft Teams chat in addition to email.

BAs interact with Risko via its 2 notification messages which are shown in figures 4.13 and 4.14. The **Future Sprint Backlog Risk Analysis notification** (figure 4.13) contains 3 main elements: (1) a short summary that allows the BA to quickly understand the purpose of the notification and thereby improves usability; (2) a list of risky sprints, showing for each sprint the team capacity, planned story points, expected velocity, and number of overcommitted

story points, together with an AI-generated suggested action for the BA; and (3) a list of opportunity sprints, showing the same metrics but with undercommitted instead of overcommitted story points.

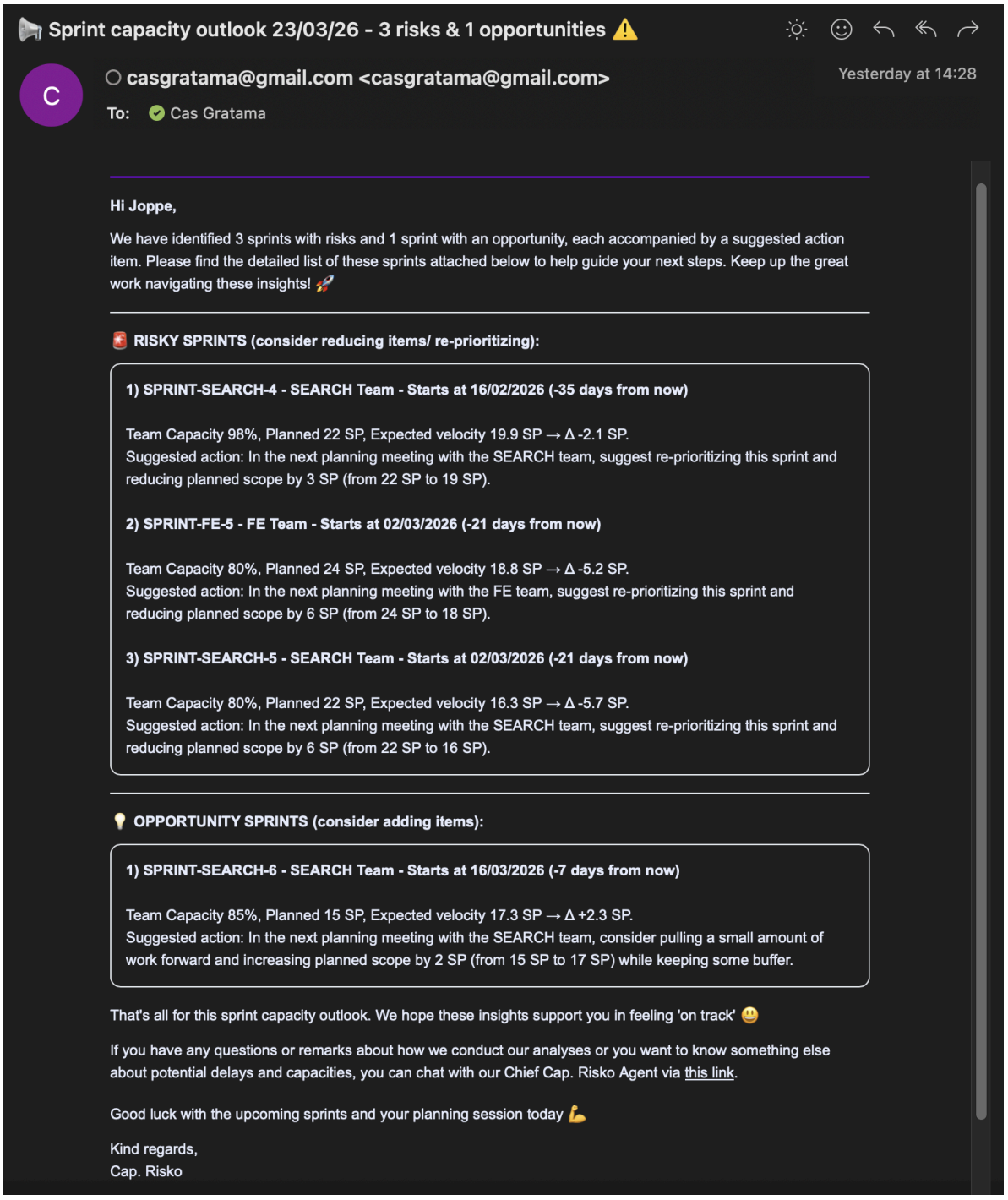


Figure 4.13: Notification email of the Future Sprint Backlog Risk Analysis workflow

The **Active Sprint PBI Delay Likelihood Analysis notification** (figure 4.13) contains 4 main elements: (1) a short summary generated by the Message Summary agent that quickly shows the BA what the notification contains and thereby improves usability; (2) a list of likely delaying dependencies that block PBIs of the BA's teams, showing for each flagged

dependency its name, owning team, the PBIs it is blocking, and in how many days those PBIs are expected to start; (3) a list of PBIs within the BA's own teams that are unlikely to be completed by the end of the active sprint, showing for each PBI its level of impact, the number of PBIs it is blocking, and how many of those blocked PBIs are scheduled to start within the next two sprints; and (4) a list of suggested action items generated by the Action Item agent.

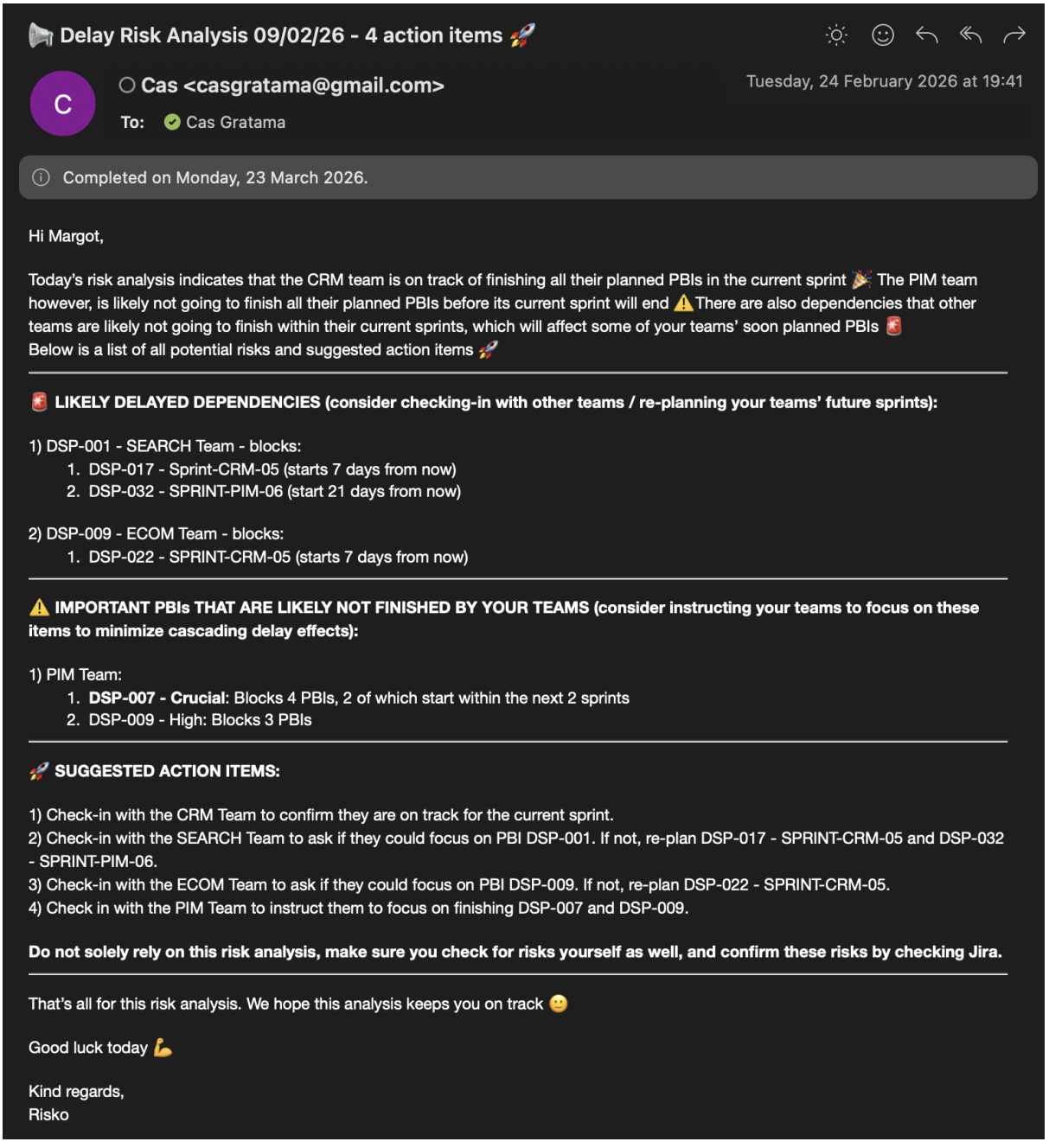


Figure 4.14: Concept email (to-be) of Active Sprint PBI Delay Likelihood Analysis workflow

Across both notification messages and the Risko dashboard, flagged PBI names are intended to function as clickable links to the corresponding Jira pages, while capacity overviews are intended to be linked directly within the notifications. This is designed to

reduce verification effort by enabling BAs to perform quality checks more quickly and directly from the notifications.

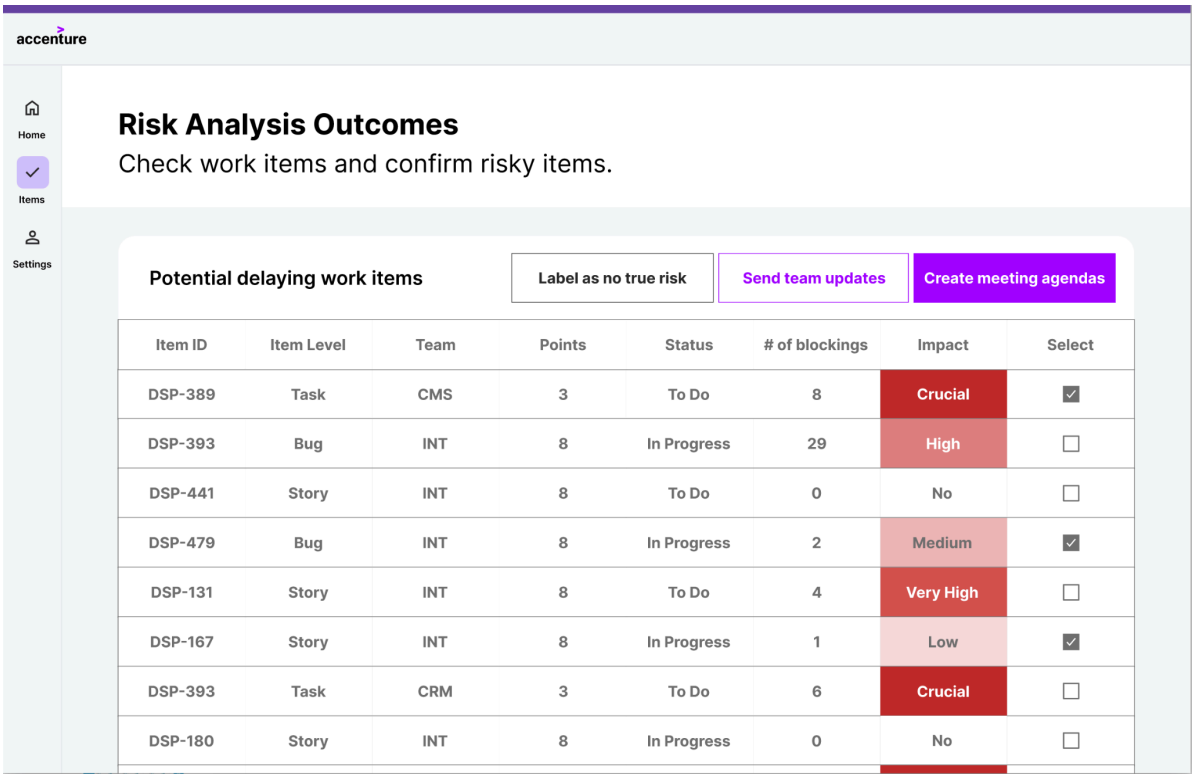


Figure 4.15: Figma design of the Dashboard of the Risko tool. Figma link.

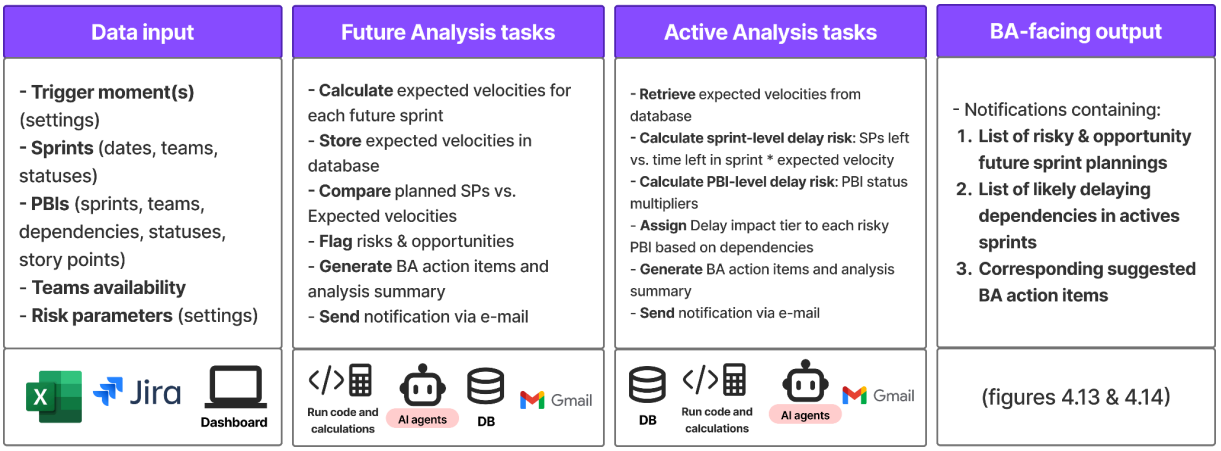
Additionally, BAs interact with Risko through its dashboard (figure 4.15), which is designed to help them respond quickly to flagged PBI risks by generating messages and documents that can be used to inform their teams and other project stakeholders about risks and help mitigate delay effects.

In its current form, Risko runs at predefined times and uses risk parameters configured in n8n and its knowledge database. However, the intended design is for the dashboard to be integrated with this knowledge database, allowing the BA to adjust workflow run times, frequencies, and risk parameters intuitively through the Risko settings page (which still has to be designed). These settings should be configurable, as the sensitivity of the risk-flagging logic has not yet been calibrated with BAs or client stakeholders during prototyping, and personalization is likely to be needed across teams, projects, and clients.

In addition, the dashboard should be extended to allow BAs to intuitively upload or integrate Azure DevOps, Jira, and team-availability Excel files, as well as create separate Risko instances for use across different projects. Although this functionality is not yet included in the Figma prototype, it is considered a key priority for further development. Notification messages are also intended to include a direct link to the dashboard, and BAs should eventually be able to instruct Risko through a dedicated Teams chat whenever needed. This

chat interface could also allow BAs to ask questions about how risks are flagged, thereby strengthening their understanding of and trust in the tool.

4.5.3 How Risko works



4.16: High level overview of what Risko does

Both Risko workflows are triggered by time, meaning that the BA does not need to manually instruct the workflows or their AI agents. The **Future Sprint Backlog Risk Analysis workflow** begins by analysing Jira for new sprints and sprint-status updates, which are then stored in Risko's knowledge base (1). Next, the tool retrieves the Risko parameter settings together with the historical sprint-velocity ratios of the project teams (2). It then reads the team-availability Excel files and calculates the capacity ratio for each future sprint (3). Based on this, the (historical average capacity-adjusted) expected sprint velocity is calculated for each future sprint and stored in the knowledge base (4). If a sprint has ended since the previous run, the historical sprint-velocity ratio is updated accordingly (5).

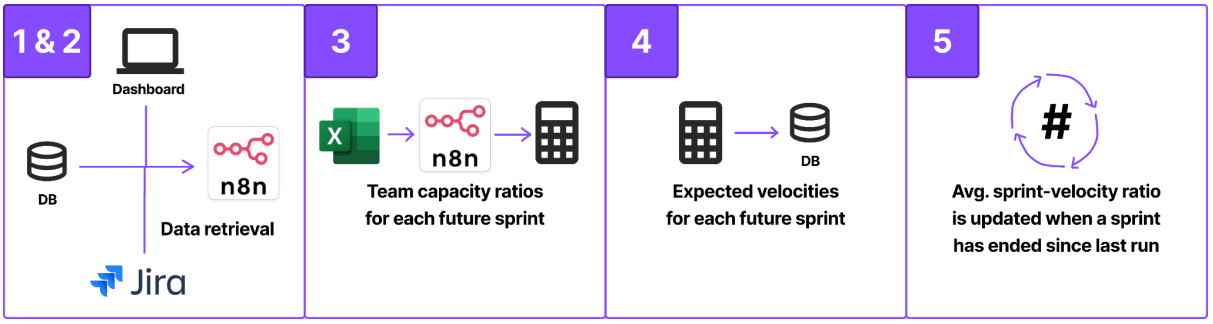


Figure 4.17: Steps 1 to 5 from Risko Future Sprint Analysis workflow

The workflow then identifies risky and opportunity sprints based on the BA-defined risk and opportunity thresholds. An opportunity sprint is a sprint in which the difference between the planned story points and the expected velocity, exceeds the opportunity threshold, indicating that additional work could likely be added. A risky sprint is a sprint in which this difference exceeds the risk threshold in the opposite direction, indicating that the sprint is likely overcommitted and that work may need to be removed (6). For each flagged sprint,

the **Action Item Agent** generates a suggested action, currently advising the BA to address the issue during the next sprint-planning session with the relevant team (7). The risky and opportunity sprint lists are then passed to the Message Summary Agent, which creates a short summary to improve the readability and usability of the notification (8). After this, the Quality & Assurance Agent checks whether sprints have been flagged correctly (9). Finally, the message is formatted and sent to the BA, containing the risky and opportunity sprint lists together with their suggested add or remove actions, which can serve as input for planning discussions and backlog refinements with teams (10).

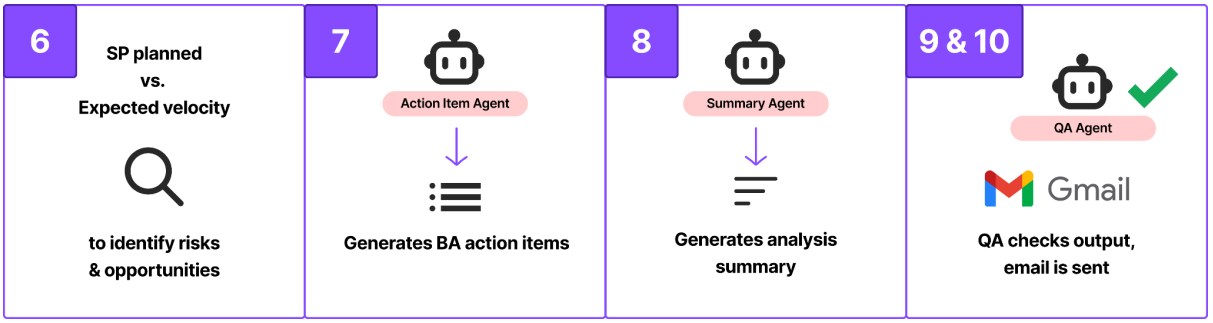


Figure 4.18: Steps 6 to 10 from Risko Future Sprint Analysis workflow

The **Active Sprint PBI Delay Likelihood Analysis workflow** begins by analysing Jira, after which the active sprints and their PBIs are retrieved (1). It then retrieves the expected sprint-velocity values calculated by the Future Sprint Backlog Risk Analysis workflow from Risko's database (2). Next, the remaining sprint fraction is calculated (3). At sprint and team level, the workflow calculates the expected number of story points that can still be completed before the sprint ends, the Risk Ratio, and a Team Risk Score for each active sprint and team, based on the adjustable threshold values *Rsafe* and *Rcritical* (4).

At PBI level, a base risk score is calculated. PBIs with the status In Progress receive a lower risk score than PBIs still marked To Do, reflecting the assumption that work already in progress is less likely to remain unfinished by sprint end. To further refine this assessment, the calculation incorporates In-Progress status maturity (*M* and *α*), meaning that a PBI that has been in progress for several days is assessed as less risky than one that has only recently entered that status (5). Based on the resulting PBI Risk Score, Risko flags PBIs as risky when their score exceeds 40, this threshold can be adjusted by the BA for customization.

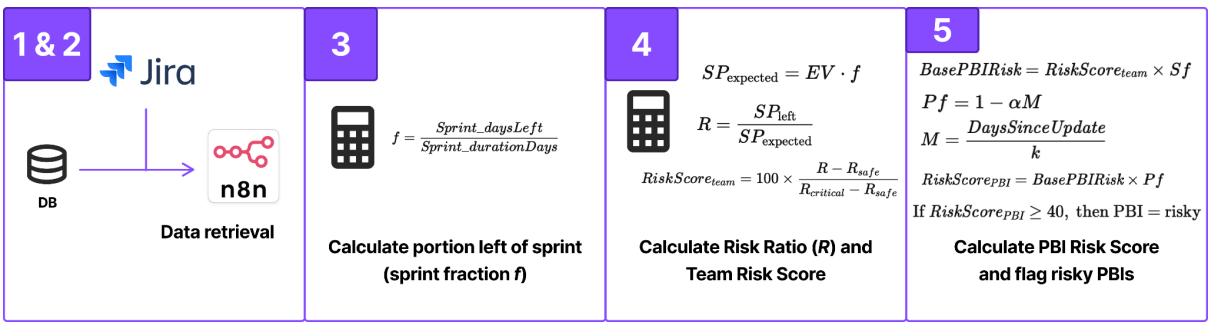


Figure 4.19: Steps 1 to 5 of the Risko Active Sprint Analysis workflow

For each risky PBI, the tool then analyses its Blocks field, which specifies which PBIs it is blocking and thus represents dependency relations. Based on this information, Risko constructs a dependency graph and traces cascading blocker chains (e.g., A blocks B, B blocks C, C blocks D). It then calculates, for each risky PBI, both the total number of PBIs it blocks (*blockedTotal*) and the number of those blocked PBIs that fall within the soon window (*blockedSoon*), meaning they are planned within the first two upcoming sprints after the active sprint (6). Based on these values, each risky PBI is assigned an impact level (table 4.6). This means that impact is determined not only by how many items a PBI blocks, but also by how soon those blocked items are planned, which is intended to improve Risko's usability (7).

Impact tier	**Assigned when**	**Rank**
Crucial	blockedTotal >= 4 or blockedSoon >=3	5
Very High	blockedTotal = 3 or blockedSoon = 2	4
High	blockedTotal = 2 and blockedSoon = 1	3
Medium	blockedTotal = 1 and blockedSoon = 1	2
Low	blockedTotal = 1 and blockedSoon = 0	1
No	blockedTotal = 0	0

Table 4.6: PBI delay impact levels

Next, the risky PBI lists are generated (8). Based on these results, the Action Item Agent creates a list of suggested BA actions, while the Message Summary Agent produces a short summary of the main outcomes and inserts it into the email (*9). Finally, the email is formatted and sent to the BA (*10).

*Note: In its current form, as further explained in chapter 5.2, the Active Sprint PBI Delay Likelihood Analysis workflow's Action Item Agent and Message Summary Agent are not correctly set up so that they generate the action items and a summary as shown in figure 4.14. It now generates a single email containing all flagged PBIs and sends it to one email address and this email does not contain an email summary. Due to time limitations, the functionality to create separate lists for individual BAs has not yet been implemented. However, this is not considered difficult to add, as this functionality is already implemented in the Capacity & Velocity Analyzer workflow.

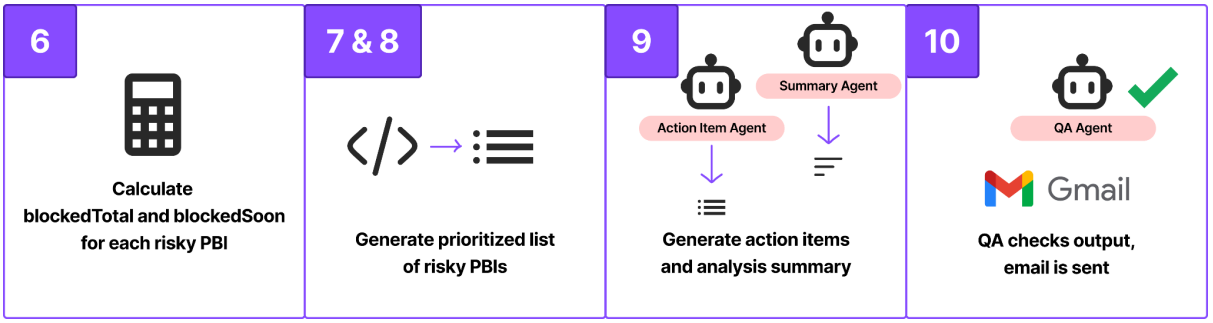


Figure 4.20: Steps 6 to 10 of Risko Active Sprint Analysis workflow

Parameter name	What it is and used for
Opportunity_threshold (future analysis)	Is the value for the SPplanned vs. SPexpected above which the sprint plan is considered a true opportunity.
Risk_threshold (future analysis)	Is the value for SPplanned vs. SPexpected below which the sprint plan is considered too risky. Because the SPplanned vs. SPexpected becomes negative when over-planned, this threshold is typically ≤ 0 (e.g., -2).
R_safe (active analysis)	is the lower ratio threshold used to flag PBIs as safe / on track. If a PBI's ratio R falls at or below Rsafe, it indicates the remaining work is comfortably within what's expected/possible for the remaining sprint time.
R_critical (active analysis)	is the upper ratio threshold used to flag PBIs as critical / highly risky. If a PBI's ratio R rises at or above Rcritical, it indicates the remaining work is too high relative to what's expected/possible, so the PBI is likely to slip.
PBIrisk_threshold (active analysis)	determines when a PBI is officially classified as risky (and should be surfaced in alerts/dashboards). - Lower threshold → flags more PBIs (more sensitive / conservative) - Higher threshold → flags fewer PBIs (less sensitive / more selective)
Sf (active analysis)	scales the baseline risk of a PBI based on its current status. If a PBI is 'In Progress', Sf is set lower to reflect that in progress PBIs are generally less risky than To Do PBIs. BAs can increase or reduce how strongly status affects the risk score. - Higher Sf (closer to 1) → the status has less risk-reducing effect, so the PBI contributes more to the risk score. - Lower Sf (closer to 0) → the status has a stronger risk-reducing effect, so the PBI contributes less to the risk score (and if it's 0, it contributes nothing).
k (active analysis)	controls how quickly the progress maturity factor Pfchanges as DaysSinceUpdate increases. - Lower k means DaysSinceUpdate is divided by a smaller number → M becomes larger sooner → stronger / faster risk reduction (more optimistic). - Higher k means DaysSinceUpdate is divided by a larger number → M grows more slowly → weaker / slower risk reduction (more conservative).
α (active analysis)	controls how strongly the progress maturity factor Pf changes as DaysSinceUpdate increases. - Lower a means weaker risk reduction (= more conservative) - Higher a means stronger risk reduction (= more optimistic)

Table 4.7: Adjustable Risko parameters

4.5.4 Risko's expected impact (value proposition)

By proactively surfacing potential delays and future planning risks, Risko is designed to reduce missed or forgotten check-ins by notifying BAs of emerging risks and providing a targeted list of follow-up actions involving both their own team(s) and other project stakeholders, thereby addressing objective C. In addition, it is intended to reduce the time needed to adjust future sprint plans by suggesting specific PBIs to add or remove when a sprint backlog appears over- or undercommitted relative to expected velocity, additionally addressing objective E.

Through these functionalities, Risko is expected to improve dependency-related planning coordination efficiency and help limit cascading delay effects, thereby supporting project delivery. At the same time, it is intended to strengthen the BA's sense of control, preparedness during check-ins, and overall feeling of being on track.

4.5.5 Risko and the design requirements

R1 cannot yet be considered fully met, although the prototype comes close. The current prototype is integrated with a mock Jira environment and can interpret and store that data in its own knowledge base. However, because Jira naming conventions and formats may differ slightly across projects and clients, and some clients/projects use Azure DevOps instead of Jira, the tool still requires a mechanism to recognize and normalize these variations in order to enable quick and intuitive project-context capture. This is considered a suitable task for an AI agent, as an LLM can analyse such differences and identify similarities effectively.

R2, R4, and R6 are met. Risko uses AI only where it adds clear value: calculations and data retrieval are handled through non-AI n8n nodes, while AI is used only for generating textual content. This keeps costs as low as possible. In addition, Risko uses specialized, heterogeneous agents, each with a distinct task, and the "AI chain" is limited to one: No AI output is passed to a second agent, as each agent operates on a separate part of the workflow and does not depend on the output of another agent.

R3 is met. Risko relies on accessible and easily integrable tools, namely Excel sheets and Jira. n8n has built in features to integrate with these tools and files.

R5 is met. Risko is designed to augment rather than replace the BA. Its outputs are BA-facing and intended to help the BA perform their work more efficiently.

R7 is met. Risko is designed around a HITL principle: the BA always has the final say and is expected to validate AI output before acting on it. The tool does not autonomously change sprint plans or update stakeholders; instead, it provides actionable recommendations that the BA may choose to act on, such as initiating a check-in with a specific team about a flagged PBI, addressing high-risk PBIs during sprint discussions, or considering a specific

PBI for an opportunity sprint. Any recommendation must ultimately be reviewed and implemented by the BA in Jira, which serves as an embedded quality check. In addition, BAs are expected to validate flagged risks through stakeholder conversations before making actual planning adjustments. Additionally, before the BA creates any meeting discussion documents, the dashboard shows a pop-up reminding the BA to perform quality checks before these documents are made by the tool (figure 4.22).

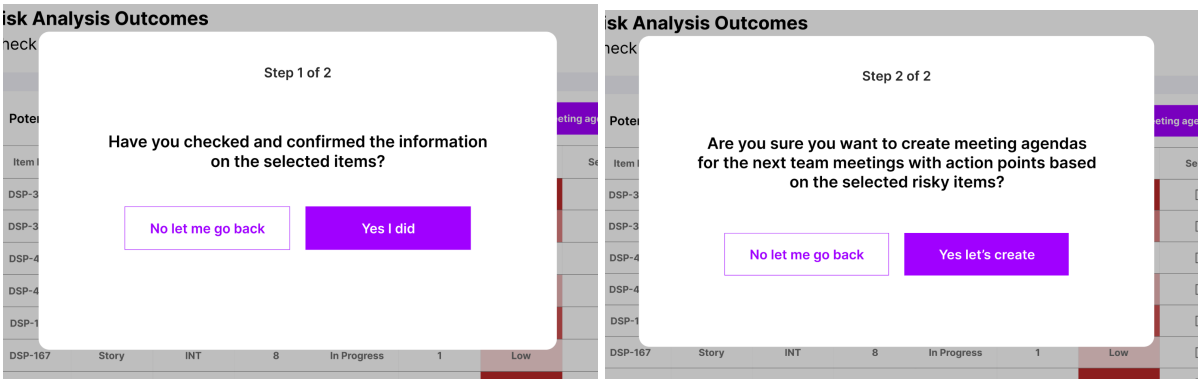


Figure 4.22: Risko dashboard containing quality check pop-ups

R8 is met. As shown in figures 4.13, 4.14, and 4.15, both the notification messages and the dashboard incorporate Accenture's purple brand colour. In addition, the dashboard is designed to align visually with other internal Accenture dashboards.

4.6 Agentic AI development learnings from prototyping

4.6.1 Design trade-offs between autonomous and deterministic MAS setups

When developing Jumpstart the key lesson learned is that there is a trade-off between autonomy and ease of implementation in tool design. If opting for a more Agentic approach, a tool could obtain more autonomy, flexibility, scalability and therefore more usability (it could potentially perform more different tasks), while a more deterministic approach allows for more controllability, easier testing, and easier initial setup.

Jumpstart is initially structured around an Orchestrator agent that receives the BA's message and coordinates the other agents needed to generate an answer (figure 4.23), adopting an ordered MAS coordination model. However, this setup introduces several challenges. Because the Orchestrator must determine which agent to call and in what order, its system prompt contains the full workflow logic. Testing shows that this leads to increased token usage, since the full system prompt is reprocessed each time the Orchestrator calls another agent and receives its output. In addition, the Orchestrator does not always invoke agents in the correct sequence, which results in poor answers to BA questions.

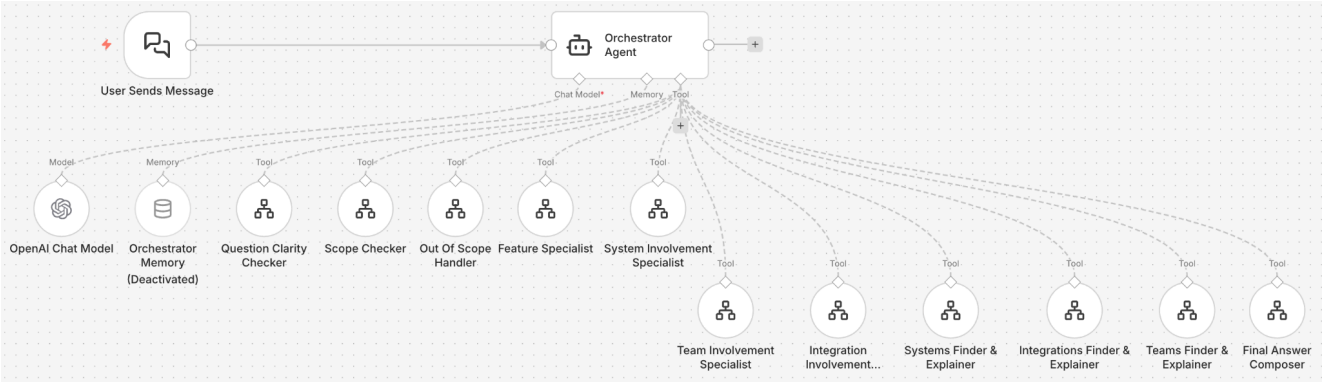


Figure 4.23 Jumpstart orchestrator setup

In principle, such an orchestration-based setup remains desirable in many Agentic AI-driven applications, as it would allow Jumpstart for example to plan its own approach to answering a question and could support more natural back-and-forth interaction through the Orchestrator's memory component. However, implementing this effectively would require more advanced prompt chaining (IBM, n.d.), better dynamic prompt injection across agents, more effective use of agent tool descriptions within n8n so that the Orchestrator understands when to call each tool (figure 4.23), and testing the MAS' behavior. Due to time limitations, this hasn't been feasible to implement.

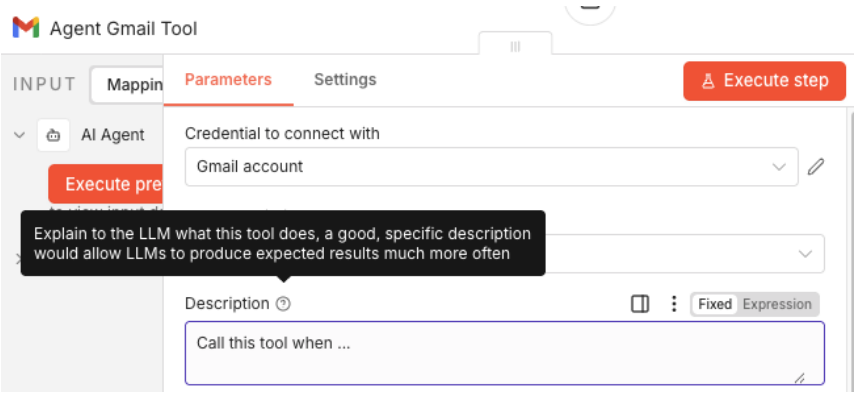


Figure 4.24: An AI agent's tool description in n8n

To make Jumpstart functional, it is restructured into a more deterministic setup (figure 4.24), which improves controllability and feasibility but limits flexibility and scalability. Agents don't call tools themselves. Instead, relevant tools and RAG sources are accessed before or after an agent is invoked, and the required information is passed directly into the agent's prompt (figure 4.25). This results in a more structured and predictable sequence of tasks, making the system easier to build and reducing the number of API calls and tokens required.

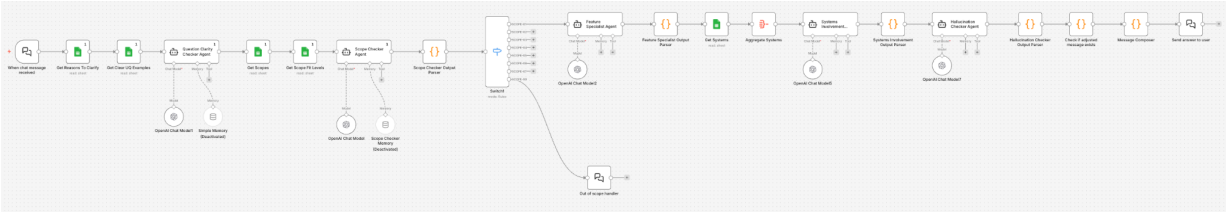


Figure 4.25: Jumpstart deterministic setup overview

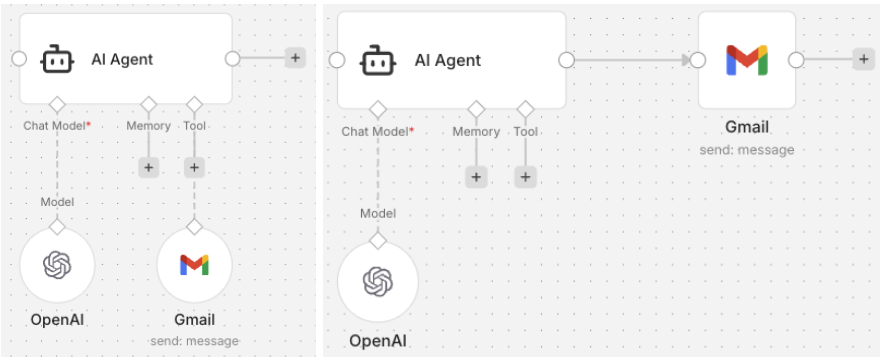


Figure 4.26: A more ‘agentic’ approach (left) vs. a more deterministic approach (right)

The trade-off of this deterministic setup is that each supported question type in Jumpstart requires its own predefined scope route containing n8n nodes and corresponding agent prompts. As a result, more of the workflow must be hard-coded manually, rather than allowing the AI to determine its own reasoning path. While this improves feasibility within the project, it reduces the system’s autonomy, usability and scalability. It is important to understand that you can adopt different approaches in different parts of the entire workflow in n8n, and therefore it is important to understand when and where adopting each approach fits best. A high-level rule of thumb could be: Whenever the outcome should be the same every time that particular part of the workflow is run, it should be set up using the deterministic approach, and if the agent should always use a particular tool before performing its task, it should also be set up using the more deterministic approach. If the agent should decide when to call a tool, it should be set up using the more ‘agentic’ approach.

4.6.2 Use parser nodes for AI output validation

Whenever an AI agent is expected to format its output in a specific way, or to generate structured items containing required elements, such as a likelihood level and its explanation, inserting a parser node directly after that agent is highly useful. This helps ensure that the output matches the intended format and allows errors to be flagged immediately, thereby supporting faster evaluation, error handling, and tool improvement.

4.6.3 Reduce hallucination risk using output parsers, QA agents and HITL

Placing output parser nodes after agent steps can help detect errors such as hallucinations and formatting mistakes. In addition, QA agents can be included to evaluate whether the output of other agents is correct in relation to the available data and the assigned task, for

example whether an agent correctly answers a BA question or selects an appropriate PBI to add to or remove from a sprint backlog. Lastly, HITL checks can be embedded at critical points in the workflow by explicitly requiring a human quality check before further action is taken. Implementing such control mechanisms effectively is highly important, as they increase the reliability and usability of the tool.

4.6.4 Ensure HITL by designing for augmentation instead of automation

Before any output is shared with other stakeholders, a HITL check should be in place. 1 way to implement this is by adding a dedicated HITL node to the n8n workflow. However, another approach is to design Agentic AI primarily as a means of augmenting, rather than automating, BA work. This is important because experts emphasize that BAs should continue developing their own skills and understanding, which could be undermined if their work were fully or heavily automated. In addition, a BA can only perform a meaningful quality check if they remain sufficiently involved in the task itself.

Both Risko and Jumpstart are therefore designed as BA-augmentation tools that produce BA-facing output. Rather than taking over BA tasks, they provide information and suggestions that help BAs perform their work more efficiently. Because the BA remains the one who interprets and acts on the output, a human check is inherently embedded in the workflow.

4.6.5 Assign AI and deterministic logic to the right tasks

As the collected primary data shows, it is important to understand which tasks should be performed by AI and which are not well suited to AI. This distinction is equally important when designing an AI-driven tool. Tasks should be deliberately divided between LLMs and deterministic components, such as code nodes. LLMs are better suited to tasks involving the interpretation of unstructured data, such as text analysis, summarization, explanation generation, and the creation of natural-language content in different formats. Deterministic nodes, by contrast, are more appropriate for tasks such as calculations, filtering, routing, formatting, and message delivery.

4.6.6 Allow for customization whenever possible

Notification preferences and risk settings are likely to differ across BAs, projects, and clients. Participants mentioned different preferred moments for receiving Risko notifications, such as at sprint start, mid-sprint, toward sprint end, or aligned with weekly planning rituals. This suggests that notification timing and frequency should be adjustable. In addition, key assumptions built into Risko, such as the risk multiplier assigned to PBIs with an In Progress status, should also be configurable through the tool settings. These parameter values have not yet been fully validated and may need to vary across projects and client contexts.

4.6.7 Design for seamless integration into existing workflows

Gathered feedback suggests that the designed Risko dashboard appears capable of helping BAs handle flagged risks more efficiently. However, 1 BA also notes that it already works with many dashboards for different purposes, and that adding “yet another dashboard” is undesirable. The BA therefore indicates that Risko should ideally be integrated directly with Azure DevOps and Jira to enable more seamless use and increase the likelihood of successful workflow adoption.

At the same time, participants’ integration preference does not fully remove the need for a dashboard concept: even if analysis is surfaced inside Jira, a lightweight dashboard (or settings page) would still be required to manage tool configuration, such as selecting the BA’s team scope, risk thresholds, and notification preferences.

4.7 Justification for only developing Risko further

Based on the prototyping review sessions and interviews, Risko is selected as the primary prototype because of multiple reasons.

4.7.1 Risko has lower hallucination risk

First, when compared to Jumpstart, Risko has a lower risk of hallucinations as Risko’s core functionality relies on deterministic calculations of capacities, velocities, risks and impacts, while Jumpstart’s core functionality is built upon the reasoning of its AI agents when generating an answer to the BA question.

4.7.2 Risko has higher evaluability in mock environment

Risko can be evaluated better in its mock environment because it is easy to assess whether a PBI has the correct number of story points, a certain number of cascading dependencies, a risk score reflects the underlying calculation logic, or whether the flagged PBIs match the configured thresholds, while, Jumpstart’s functionality is harder to evaluate because it relies more on AI interpretation quality, and it is harder to verify whether a generated explanation or system involvement suggestion is “good enough” in practice without extensive real project validation.

Because Risko’s outputs can be checked against input data and deterministic rules, it is easier to determine whether the prototype is working correctly and whether its results are meaningful.

4.7.3 Risko has higher feasibility

Risko is assessed as easier to build during prototyping within the available project time and technical constraints, which increased its feasibility for development during this project. Jumpstart’s capabilities are feasible in principle, but they introduce more complexity and

uncertainty during prototyping. Jumpstart is also harder to maintain: It should constantly update its knowledge base based on changing Confluence data, which is more variable than Risko's Jira data on which it relies. Risko is more 'stable'. Therefore, even if thesis project time limitations are taken out of the equation, Risko is considered more feasible than Jumpstart.

4.7.4 Risko addresses a more continuous BA need according to BAs

Another important reason for prioritizing Risko is that BAs indicate that Risko addresses a continuous problem in BA work as risks need to constantly be flagged during projects, while Jumpstart is believed to offer significant value when onboarding, but value shrinks when the BA learns enough.

Later, Jumpstart could implement a functionality in which it helps flag dependencies by analyzing the API/system documentations and provide suggestions for questions/stakeholders to talk to which could offer more continuous value, but in its current form it does not do this, and that functionality would also be hard to evaluate without real-life project data.

4.7.5 Risko is easier to deploy across projects

Risko is also considered more transferable across projects, because its logic is based on project-management structures that are common in many Agile delivery contexts, such as sprints, PBIs, statuses, team capacities, and dependencies. Although naming conventions may differ between projects, the underlying data is generally similar and available across contexts. Jumpstart, by contrast, depends more heavily on documentation that can vary substantially between projects. Different clients may use different documentation formats or strategies, and some may lack up-to-date system architecture diagrams or organizational charts altogether. This makes it more difficult to design Jumpstart in a way that is consistently usable across projects.

4.7.6 Risko is picked, but Jumpstart is not rejected

Selecting Risko as the primary prototype should not be interpreted as a rejection of Jumpstart. BAs consider both concepts valuable as feedback suggests that Risko may offer greater organizational value, whereas Jumpstart may offer greater personal value to BAs. Prioritizing Risko therefore does not imply that Jumpstart is less relevant overall. Rather, due to the time limitations of this project, Jumpstart is assigned a lower development priority and is not developed further.

5. Deliver

This chapter marks the Deliver phase of the project, in which the chosen concept is further translated into practical and evaluative deliverables. Building on the decision to continue with Risko, the chapter first explains how the tool can be used within the BA workflow, after which it outlines a roadmap for its further development and deployment within ASC. The chapter concludes by validating Risko in relation to its intended contribution to BA efficiency and by reflecting on its broader organisational and professional implications.

AI used in this chapter:

- ChatGPT to support formatting text and improving text
- ElevenLabs to improve the voice over in the demo video
- Gemini to generate images of demo video characters and environment
- Vertical Studio to generate videos for demo video

5.1 Risko usage guideline

5.1.1 Where to use Risko

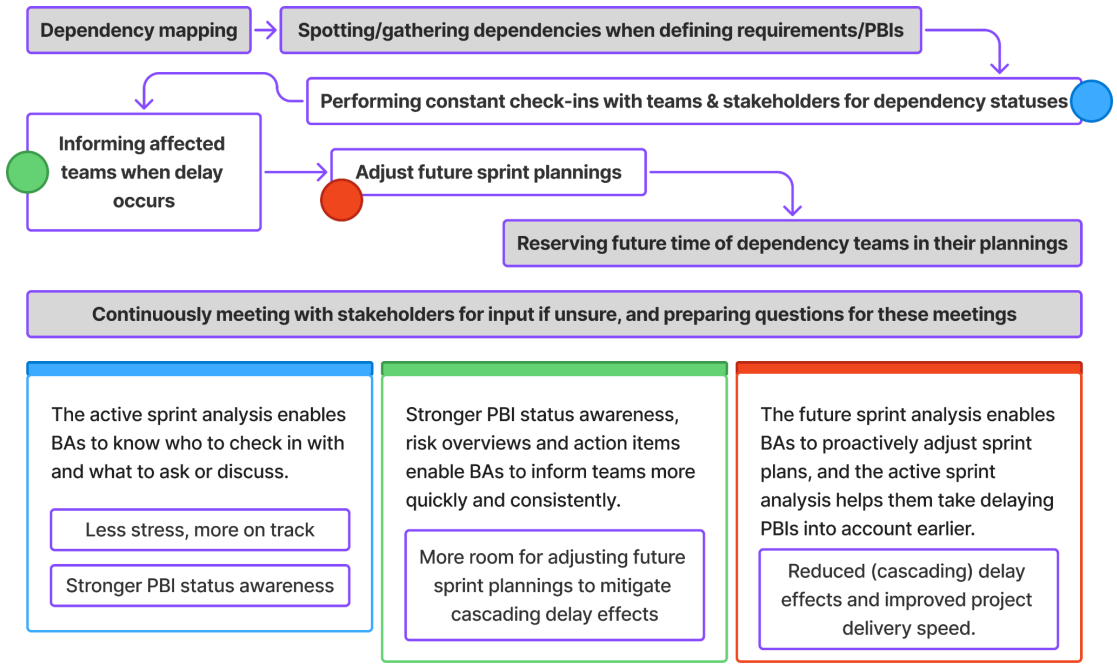


Figure 5.1: Overview of where in the dependency management workflow Risko offers support to BAs

Risko is designed to support BAs in the parts of their workflow where maintaining oversight of PBI progress, dependencies, and sprint feasibility is most difficult. This support is especially relevant during the current sprint, in the period before sprint planning, and during sprint planning itself.

During the current sprint, BAs often rely on manual check-ins and memory to stay aware of PBI progress and dependency status. Risko supports this by providing a current sprint progress overview that highlights potential delay risks and indicates which teams or stakeholders require follow-up. This enables the BA to perform more targeted check-ins, address risks earlier, and better inform the PO and impacted teams. At the end of the sprint, Risko can also show which PBIs were not completed, helping the BA adjust upcoming plans more proactively.

Risko also supports the BA before sprint planning. In this phase, BAs need to collect updates on delayed PBIs, check team capacity, and prepare a realistic sprint draft together with the PO. Risko contributes by providing a capacity analysis overview that combines team capacity, expected velocity, and historical average velocity. This helps the BA create a more realistic sprint draft that takes delayed dependencies and expected delivery pace into account.

During sprint planning itself, Risko serves as decision support rather than replacing team discussion. The BA still aligns with team members and stakeholders on spillovers, availability, PBIs, and sprint goals, but does so with better insight into dependency delays and expected team velocity. In this way, Risko helps make sprint planning more realistic, more proactive, and less reactive.

Overall, Risko fits into the BA workflow as a proactive support layer that improves awareness and planning quality across sprints. Its main value lies in helping BAs identify where action is needed, reduce missed follow-ups, and support smoother and more efficient product delivery.

5.1.2 How to use Risko

Figure 5.2 shows a functional overview of how Risko works, including how BAs will work with it. Risko runs at predefined moments which the BA can set in its settings, and additionally to what is visualized in figure 5.2, BAs interact with Risko via the 2 email notifications it sends, in which one is focused on future sprint backlog risk notification messages: one focused on future sprint backlog risk and one focused on active sprint PBI delay likelihood. After reading the short summary at the top of each notification, the BA reviews the flagged risks, opportunities, and suggested actions.

The BA then uses this information to decide where follow-up is needed. This may include checking in with relevant teams or stakeholders, discussing flagged issues with the PO, verifying linked Jira items, and adjusting sprint planning where necessary. When more support is needed, the BA can use the dashboard to inspect risks further and generate communication or documentation for stakeholders. Risko therefore supports the BA in acting more proactively and efficiently, while keeping the BA in control of interpretation and decision-making.

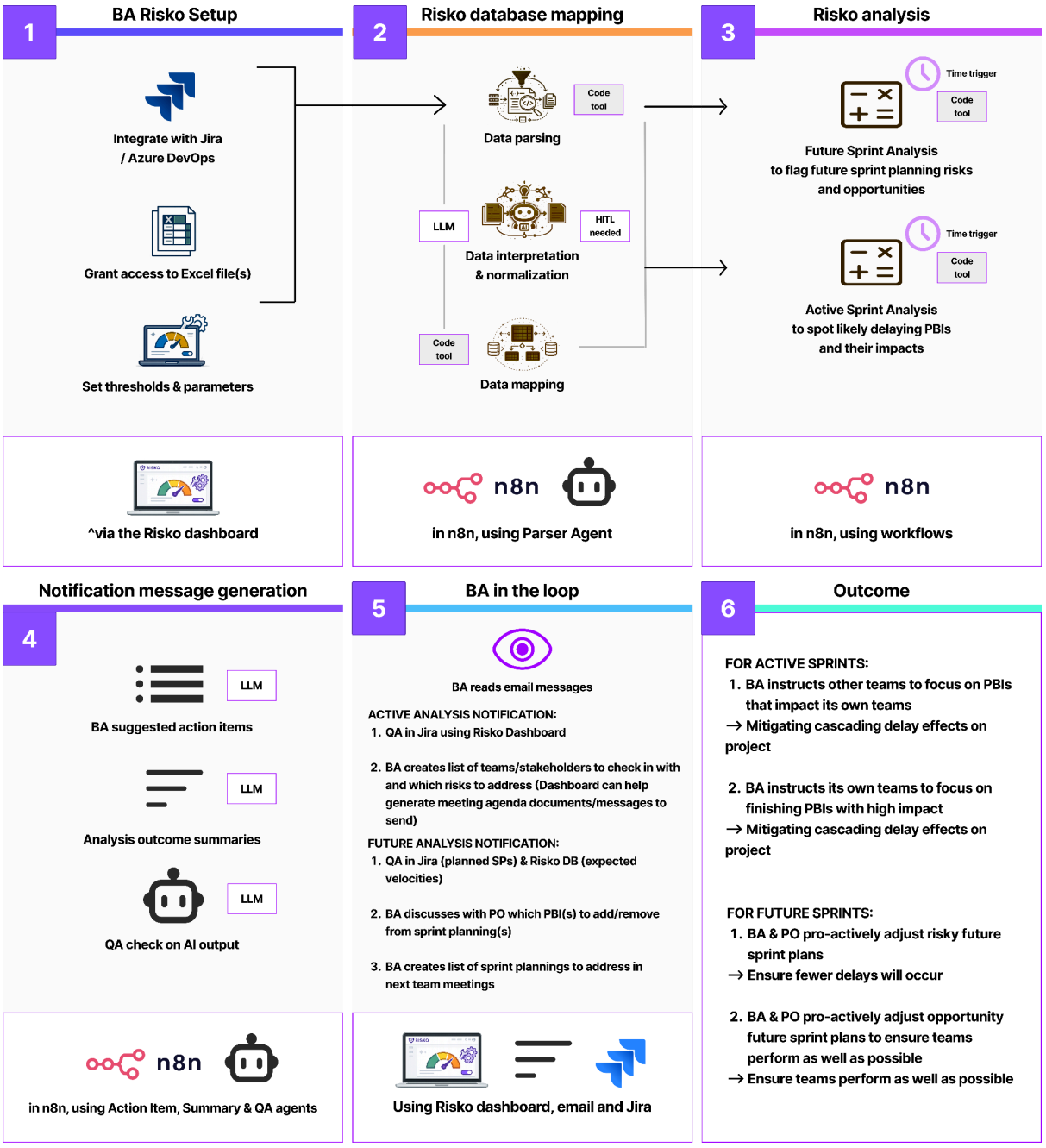


Figure 5.2: Risko functional overview containing Parser Agent

5.2 What is next for Risko

5.2.1 Where to add more AI in Risko to add value

Across both Risko workflows, a number of AI agents could be added to offer more AI value according to the interviews held during prototyping and the researcher's personal understanding.

Increase cross-project applicability with parser AI agents

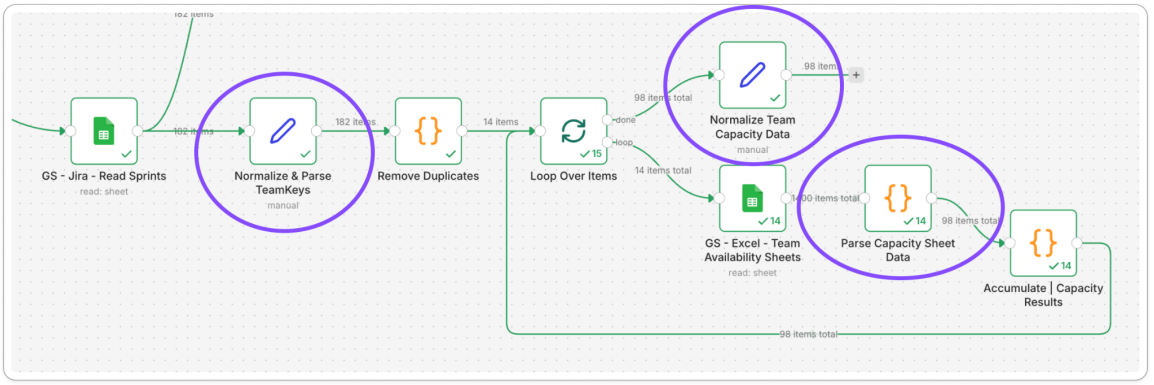


Figure 5.3: Risko's highlighted, hard-coded Jira/Excel normalization and parser nodes

Across both workflows, the parser and normalization nodes after the Google Sheets retrieval nodes are currently hard-coded. This makes them reliable within the prototype, but only for the specific Jira- and Excel-like formats used. If data structures or naming conventions differ, the workflow will not function correctly. In a real-world version, **AI parser agents** could interpret and map such variations dynamically, making the workflow more adaptable across tools and projects. Such an agent's functionality is portrayed in figure 5.2.

In this prototype, however, the focus is on developing Risko's core analytical functionality rather than dynamic data processing. These steps were therefore hard-coded, which limits broader applicability but was considered necessary within the project timeframe.

Reduce replanning time through suggested, targeted PBI to remove or add

By adding either a deterministic **PBI-retrieval tool** to the **Action Item Composer Agent** or a separate **PBI Retriever Agent** (figure 5.3), the notification messages could include concrete PBI suggestions (figure 5.4). This could reduce the time needed to adjust planning, as BAs would no longer need to search manually for suitable PBIs to add or remove. Although the BA would still need to perform a quality check, this functionality could provide a useful jumpstart for updating sprint plans. Feedback from P4 suggests that this would be a valuable feature to add.

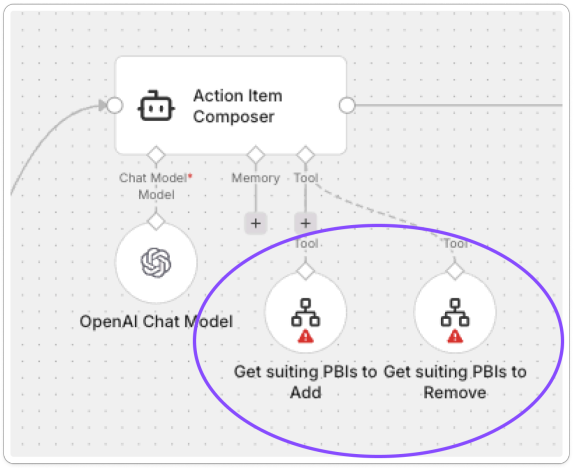


Figure 5.4: Risko's conceptual PBI-retrieval tools

Figure 5.5: PBI retrieval tools enable PBI removal/addition suggestions

When a risky or opportunity sprint is identified, the **Action Item Agent** could call this PBI add/remove tool. For risky sprints, this tool would suggest one or more low-importance PBIs for removal whose combined story points match the overcommitment. For opportunity sprints, it would suggest one or more high-importance PBIs from the product backlog for addition whose combined story points match the available capacity. These suggestions would then serve as concrete action items in the notification message.

The **QA Agent** should then verify whether the suggested PBIs exist in Jira and whether their importance levels are correct. Because this functionality is mainly based on data retrieval and calculation logic, it could likely be implemented using deterministic nodes only which the Action Item Composer can call as a tool.

Increase adoption-rate and value by adding conversational capabilities

By modularizing the workflow into separate components and exposing these as tools for AI agents, Risko could evolve into a more conversational system, in line with the more agentic approach discussed in the key prototyping lessons. This would allow BAs to ask, for example, a Risko Explainer Agent questions about flagged risks, opportunities, and the underlying logic of the tool. In turn, this could strengthen trust in the tool and support adoption.

A more agentic setup could also enable BAs to instruct Risko to perform additional analyses beyond those explored in this project. In the longer term, this may allow Risko and Jumpstart to converge into a single tool that supports BAs both in understanding systems and dependencies and in identifying delay, planning, and opportunity risks. Such a combined tool is considered promising, as it could improve BA efficiency across multiple key areas identified in this report.

5.2.2 Risko adoption roadmap

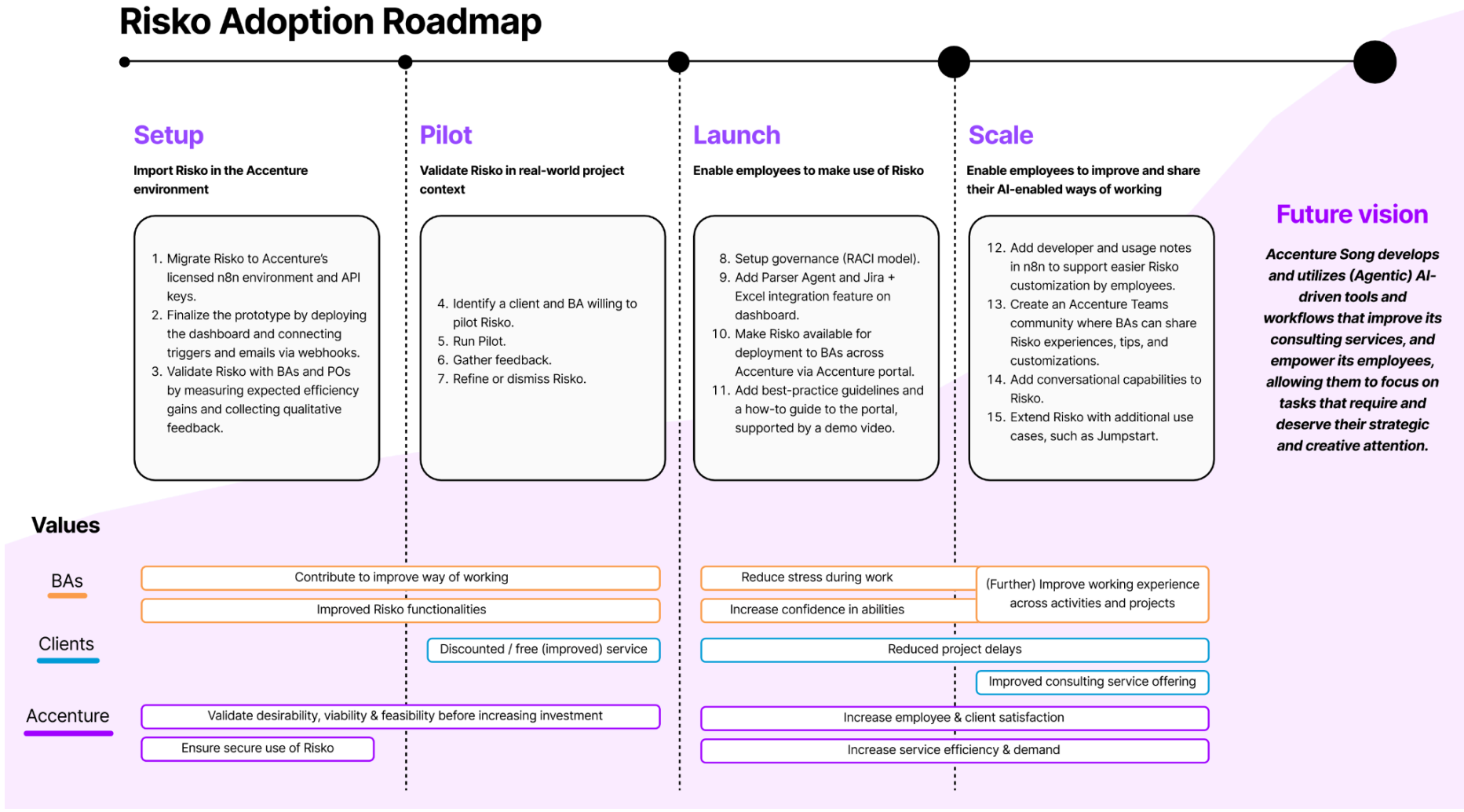


Figure 5.6: Risko adoption map

The roadmap shows how Risko could progress from prototype to practical adoption. It highlights that the next steps involve technical setup, pilot validation in a real project context, and, if successful, wider launch and scaling within Accenture. This shows that the future of Risko depends on both further development and successful organisational embedding.

5.3 Risko validation

To determine whether Risko is worth further development, the concept is assessed using 3 validation dimensions: desirability, feasibility, and viability. Together, these dimensions help evaluate whether the tool meets BA needs, can be realistically developed, and offers sufficient value for Accenture.

Desirability

Throughout this project, it became clear that BAs strongly desire support in dependency management within composable commerce projects. In the survey, eight participants rated their interest in such support at an average of 4.63 out of 5.00, or about 93% of the maximum score. In the interviews, three of the four BAs indicated that dependency management was the area in which they most wanted support. During prototyping feedback sessions, BAs also stated that they would want to use both Jumpstart and Risko if these tools were available.

This strong interest aligns with the challenges BAs described. They reported feeling stressed and overwhelmed, and some noted that it is nearly impossible to maintain full awareness of all dependencies and their statuses in a complex project environment. At the same time, experts emphasized that sufficient dependency and system awareness is essential for effective dependency management. Because BAs must continuously check in with multiple teams and stakeholders across a complex system landscape, they often struggle to know whom to contact, when to do so, and what to discuss. Some even reported occasionally missing or forgetting relevant check-ins.

Risko is designed specifically to address this challenge. It can therefore be considered a tool with clear desirability among BAs.

Feasibility

From a development perspective, Risko is considered feasible for several reasons. First, the data required to perform its analyses is relatively easy to retrieve and integrate. Jira and Azure DevOps provide supported APIs through which Risko can access relevant project data, while Excel files are also accessible when the tool is deployed in an Accenture-licensed n8n environment. Second, the underlying data, such as PBIs, sprints, story points, and team availability, is largely similar across clients and projects. This increases the potential for cross-project deployment, particularly if an additional AI agent is introduced to parse and map Jira and Excel data to the internal data structure on which

Risko performs its analyses, as discussed in chapter 5.2. Third, the analytical logic of Risko is based on relatively simple calculations, which supports technical feasibility, and this feasibility is proven through the working prototype. Fourth, Accenture enterprise licenses for n8n and AI models are expected to become available soon, or are already available, which would allow Risko to be integrated into the Accenture environment more easily. Finally, the tasks assigned to AI agents within Risko are relatively limited in complexity, as they mainly involve generating textual content.

Viability

For ASC, the project client, Risko appears to offer significant viability potential through multiple value streams. First, by increasing BAs' sense of being on track and strengthening their confidence, the tool may help reduce stress and thereby contribute to higher employee satisfaction. Second, by improving dependency-status awareness and supporting sprint-planning activities, Risko may help reduce project delays. This, in turn, could improve client satisfaction and potentially contribute to additional sales opportunities.

At the same time, further validation is needed to better quantify this value for BAs, clients, and Accenture. For example, it would be valuable to assess how much time Risko could save BAs in building dependency-status awareness, to what extent it improves that awareness, and whether it helps reduce missed check-ins with relevant teams during sprints.

Rigorous validation testing still needed and highly recommended

Although the findings of this project suggest that Risko has strong potential and scores positively on desirability, feasibility, and viability, additional validation is still needed before it can be concluded that the tool is worth pursuing for Accenture. At this stage, Risko should be seen as a promising concept rather than a sufficiently validated solution.

Further validation should focus on questions that are critical to its practical value. For example, it would be useful to examine how much time BAs need to spend reading the notification messages and performing the necessary quality checks before acting on the suggested action items, and whether this is still considered desirable, feasible, and viable in practice. In addition, true desirability should be tested by asking BAs to put some "skin in the game" (Savoia, 2019). For instance, would they sign up for a waitlist by submitting their email address, or register for a training session to learn how to use Risko? At present, BAs only indicate that they would like to use the tool.

Such validation can be conducted relatively quickly without requiring substantial further development of Risko, for example through fake-door or mechanical turk experiments (Savoia, 2019). At a later stage, a pilot study should be used to assess real-world feasibility and viability, as discussed in chapter 5.2.

6. Concluding project

6.1 Conclusion

This thesis set out to explore the following MRQ:

How might an Agentic AI-driven project management tool support Business Analysts in increasing their work-efficiency during composable commerce platform development projects?

Through a mixed-method approach combining literature review, interviews, observations, survey data, and documentation analysis, the study shows that BA work in this context is characterised by fragmented information, high coordination complexity, frequent replanning, and a strong dependence on both system understanding and cross-team alignment. From the identified improvement opportunities, managing dependencies emerged as the most relevant focus area, as it is shown to be both highly desired by BAs and highly consequential for delivery progress.

The findings show that dependency management is difficult because BAs must continuously maintain awareness of PBI statuses, monitor upstream and downstream impacts across teams, and make timely planning adjustments in environments where information is incomplete, scattered, and often difficult to retrieve. In composable commerce delivery, this challenge is intensified by the large number of interconnected backend systems and by siloed ways of working that weaken shared ownership of dependency consequences. As a result, BAs carry substantial cognitive load and coordination pressure, making this a suitable area for Agentic AI support.

To address this opportunity, Risko is developed: An Agentic AI-driven prototype that supports BAs in maintaining dependency and future sprint planning-status awareness by analysing project data, surfacing potential risks and opportunities, and guiding more targeted check-ins and planning adjustments. In doing so, the project demonstrates that Agentic AI can offer value as a more proactive and embedded form of workflow support to BAs. The prototype translates relevant Agentic AI capabilities, such as data retrieval, tool-use and text generation, and embeds HITL control, into a concrete tool that supports BAs in their work.

In conclusion, this thesis does not claim to deliver a finished Agentic AI-driven tool, but provides a foundation for further validation and development, and shows that Agentic AI can become a meaningful enabler of more efficient, scalable, and resilient BA work in Agile composable commerce delivery projects.

6.2 Discussion

The findings of this project point to several broader insights with regards to how Agentic AI-driven project management tools can and should be developed to support BAs in their work.

First, efficient project-context capture should be treated as a core design requirement rather than a secondary implementation detail. AI support becomes weak when tools lack sufficient organizational and project-specific context, and that BAs often do not have the time, patience, or sometimes even the knowledge needed to manually provide this context through extensive prompting. This means that tools should rely on integrations, RAG, and structured context-provision mechanisms rather than expecting BAs to repeatedly explain their project situation through writing.

Second, Agentic AI tools should be designed around selective AI use, maximum user benefit, and clear practical value. This project shows that more AI, more agents, or more autonomy does not automatically lead to better support. Instead, effective tool design depends on applying AI only where it adds clear value, keeping AI chains short, and combining AI with deterministic logic in a deliberate way. In practice, this means using autonomy selectively where flexibility is needed, while implementing stable and repeatable workflow steps deterministically. It also means assigning LLMs to unstructured, interpretive tasks and relying on deterministic logic for calculations, routing, validation, and formatting. Together, these findings suggest that strong BA support tools should be designed as hybrid systems in which deterministic logic provides the stable backbone and AI adds value where interpretation, reasoning, or communication is required.

Third, augmentation is often more valuable than automation, and reliability mechanisms must be embedded throughout the tool workflow. AI output parsers, QA agents, deterministic validation nodes, and HITL checks are valuable for detecting hallucinations, formatting mistakes, and incorrect reasoning before outputs are acted upon. This is especially important in consulting contexts, where unreliable AI output can create project risk and undermine trust. At the same time, keeping the BA in the loop is important not only for QA, but also for preserving their learning, contextual understanding, and ability to make informed decisions. Both Risko and Jumpstart are intentionally developed as BA-facing tools that support interpretation and action, rather than replacing the BA in client-facing communication or decision-making.

The contribution of this project lies not only in Risko itself, but also in a broader set of principles for shaping Agentic AI into usable workflow support for BAs in complex projects. Rather than being applied merely to automate output generation, the technology should be used to help people perform their work in a more meaningful, effective, and pleasant way.

6.3 Limitations

This project has several limitations that should be considered when interpreting its findings. First, the study is deliberately scoped to a specific context: the work of BAs at ASC during the Design & Build phases of Agile composable commerce delivery projects. This narrow scope is appropriate for making the project feasible within the available timeframe and for generating context-specific insights, but it also limits generalizability. The findings therefore cannot be assumed to apply directly to other consulting contexts, other delivery phases, other industries, or other project roles such as POs, Scrum Masters, or developers.

Second, the empirical basis of the project is relatively limited in size. The study uses purposive sampling and combines interviews, a survey, observations, and documentation analysis to strengthen credibility through triangulation, but the total number of participants remains small. The survey includes eight respondents, the observation is based on one BA, and the semi-structured interviews cover a limited number of BAs and experts. As a result, the findings should be understood as indicative and exploratory rather than statistically representative of all BAs within Accenture or beyond.

Third, the project relies partly on self-reported data, which introduces a risk of recall bias and subjective interpretation. Although observations and documentation analysis are included to complement interview and survey findings, some conclusions still depend on how participants describe their own work, inefficiencies, and AI needs.

A fourth limitation is that the prototypes are developed and evaluated in a mock environment rather than in a live client setting. Because an enterprise n8n license and an Accenture LLM license are not available during prototyping, personal n8n and OpenAI accounts are used instead, which means confidential client data cannot be used. To address this, mock project data and Google Sheets stand-ins for tools such as Jira, Confluence, and Excel are created. Although this enables prototyping and early evaluation, it also means that the prototypes are not tested against the messiness, incompleteness, and variability of real project data and real organizational environments.

Related to this, the evaluative strength of the project is limited. Risko is assessed positively on desirability, feasibility, and viability, but further validation is still needed before concluding that the tool is truly worth pursuing. At this stage, Risko should be seen as a promising concept rather than a fully validated solution. In particular, the project does not yet demonstrate quantified real-world efficiency gains, such as time saved, increased dependency-status awareness, or fewer missed check-ins in live delivery settings.

6.4 Recommendations

Based on the findings, recommendations can be made for both ASC and for future designers and researchers.

6.4.1 Recommendations for Accenture

For Accenture, the main recommendation is to treat Risko as a promising pilot candidate, rather than as a tool ready for deployment. The next step should be to further refine the prototype, integrate it more closely with existing tools such as Jira or Azure DevOps, and test it in a live project setting. A pilot should examine whether Risko truly improves BA efficiency, dependency-status awareness, and planning quality in practice, by measuring outcomes such as time saved, perceived usefulness, improvement in dependency-status awareness, and reduction of missed check-ins.

In addition, Accenture should support adoption by providing clear governance, usage guidance, and room for controlled customization. Since projects, teams, and client contexts differ, tool settings such as notification timing, thresholds, and scope should be adjustable, while ownership of development and maintenance should remain clearly defined.

Lastly, Accenture should enable and encourage employees to share knowledge on Agentic AI usage and tools, which it can do by setting up a dedicated Agentic AI knowledge sharing Teams channel as indicated in the roadmap or a newsletter for example. This could improve employee understanding of the promising technology, while at the same time, allow them to improve their own ways of working, which in turn could positively affect Accenture's services offering.

6.4.2 Recommendations for designers/researchers

For designers, it is recommended to develop Agentic AI-driven BA support tools as hybrid systems in which AI and deterministic logic are deliberately assigned to different tasks. AI is most useful for interpretation, explanation, and summarization, while deterministic components are more reliable for calculations, routing, and validation. In addition, future tools should be designed for augmentation rather than automation, with strong HITL and validation mechanisms in place.

For researchers, the main recommendation is to conduct stronger real-world validation. Although this project shows that dependency management is a promising opportunity area and that Risko has potential, it does not yet demonstrate quantified effects in practice. Future research should therefore test such tools in live delivery settings and examine their impact on BA efficiency, stress, and delivery outcomes.

Finally, future designers and researchers should pay more attention to the question of how much autonomy Agentic AI should actually be given in business contexts. This project suggests that greater autonomy does not automatically create greater value. Instead, the

level of autonomy should be balanced against risks such as hallucinations, reduced human learning, higher computational cost, and greater energy usage. Future work should therefore examine which tasks should remain human-led, which can be supported by AI, and under what conditions more autonomous Agentic AI truly creates meaningful business value.

6.5 Reflection

This project has been complex, challenging, and highly meaningful to me. At the start, I had no experience with building AI-driven tools or working with n8n, and I knew relatively little about the broader project context. Looking back, I can clearly see how much I have learned, not only about Agentic AI and technical prototyping, but also about consulting, software delivery, Agile ways of working, composable architecture, and commerce platforms. My supervisors at Accenture gave me the opportunity to explore these topics in depth, which strongly contributed to both my development and motivation throughout the project.

During the project, I encountered several challenges. I often felt that I was lagging behind and rushing to keep up with tasks and deadlines, which made it difficult to step back, set priorities, and define clear boundaries for what should and should not be included in the project. I also found scoping and decision-making difficult, because I found many parts of the topic interesting and struggled to balance academic relevance with practical value for Accenture. Looking back, earlier choices would likely have helped me create more focus and more room for iteration. In future projects, I will improve this by working with clearer planning methods, such as setting SMART goals and using tools like Trello to structure priorities and keep track of progress.

What I liked most during this project was the development phase. I greatly enjoyed building and testing tools, which confirmed my interest in hands-on design and prototyping work. Through this process, I learned how strongly prompt design affects agent output quality, and how important it is to design AI-driven systems in a structured and purposeful way. One of the key lessons for me was to see Agentic AI not only as a means of automation, but especially as a tool for augmentation.

Although this reflection may sound critical, I do not look back on the project negatively. Rather, I see these points as concrete areas for improvement. This project pushed me into a new and complex domain, through which I developed new knowledge, practical skills, and a clearer understanding of how I work under pressure. I am also very grateful for the guidance, tips, and feedback I received from Jiwon and Giulia. They helped me steer the project in the right direction, make important decisions, stay motivated, improve my work, and gain meaningful insights into how I can continue to develop myself.

7. Reference list

Adeleke, A. G., Sanyaolu, T. O., Efunniyi, C. P., Akwawa, L. A., & Azubuko, C. F. (2024). Leveraging UX design and prototyping in agile development: A business analyst's perspective. Engineering Science & Technology Journal, 5(8), 2670–2693. https://doi.org/10.51594/estj.v5i8.1518

Almalki, S. S. (2025). AI-driven decision support systems in Agile software project management: Enhancing risk mitigation and resource allocation. Systems, 13(3), 208. https://doi.org/10.3390/systems13030208

Alrabaiah, H. A., & Medina-Medina, N. (2021). Agile Beeswax: Mobile app development process and Empirical study in real environment. Sustainability, 13, 1909. https://doi.org/10.3390/su13041909

Al-Saqqa, S., Sawalha, S., & AbdelNabi, H. (2020). Agile software development: Methodologies and trends. International Journal of Interactive Mobile Technologies, 14(11), 246–270. https://doi.org/10.3991/ijim.v14i11.13269

Amajuoyi, N. P., Benjamin, N. L. B., & Adeusi, N. K. B. (2024). Optimizing agile project management methodologies in high-tech software development. GSC Advanced Research and Reviews, 19(2), 268–274. https://doi.org/10.30574/gscarr.2024.19.2.0182

Appoh, M., Frempong, D., Akinboboye, O., Okoli, I., Afrihyia, E., Umar, M. O., Umana, A. U., & Omolayo, O. (2022). Agile-based project management strategies for enhancing collaboration in cross-functional software development teams. Journal of Frontiers in Multidisciplinary Research, 3(2), 49–64. https://doi.org/10.54660/IJFMR.2022.3.2.49-64

Bahi, A., Gharib, J., & Gahi, Y. (2024). Integrating generative AI for advancing Agile software development and mitigating project management challenges. International Journal of Advanced Computer Science and Applications, 15(3), 54–61. https://doi.org/10.14569/IJACSA.2024.0150306

Bajpai, S., Eppinger, S. D., & Joglekar, N. R. (2020, October 13–15). Enhancing visibility in agile program increment DSMs. In Proceedings of the 22nd International Dependency and Structure Modeling Conference (DSM 2020) (pp. 155–164). Massachusetts Institute of Technology.

Barcaui, A., & Monat, A. (2023). Who is better in project planning? Generative artificial intelligence or project managers? Project Leadership and Society, 4, 100101. https://doi.org/10.1016/j.plas.2023.100101

Bathina, S. (2025). Composable Commerce Architectures: Building agile retail systems [Research Article]. Journal of Information Systems Engineering and Management, 507. https://jisem-journal.com/

Bekhet, A. K., & Zauszniewski, J. A. (2012). Methodological triangulation: An approach to understanding data. Nurse Researcher, 20(2), 40–43.

Bharti, M. (2025). AI agents: A systematic review of architectures, components, and evolutionary trajectories in autonomous digital systems. International Journal of Computer Engineering and Technology, 16(1), 809–820. https://doi.org/10.34218/IJCET_16_01_065

Camara, R., Federal University of Pernambuco, Informatics Center (CIn), Marinho, M., Federal Rural University of Pernambuco, Computing Department (DC), Moura, H., & Federal University of Pernambuco, Informatics Center (CIn). (2024). Agile tailoring in distributed large-scale environments using agile frameworks: A Systematic Literature Review. In CLEI Electronic Journal (No. 1; Vol. 27, p. Paper 8).

Cinkusz, K., & Chudziak, J. A. (2025). Agile software management with cognitive multi-agent systems. In Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025), Volume 1 (pp. 385–392). SciTePress. https://doi.org/10.5220/0013153000003890

Cinkusz, K., Chudziak, J. A., & Niewiadomska-Szynkiewicz, E. (2025). Cognitive agents powered by large language models for Agile software project management. Electronics, 14(1), 87. https://doi.org/10.3390/electronics14010087

Dhanji, K. (2025, October 6). Deloitte to pay money back to Albanese government after using AI in $440,000 report. The Guardian. http://www.theguardian.com/australia-news/2025/oct/06/deloitte-to-pay-money-back-to-albanese-government-after-using-ai-in-440000-report

Dragos, P. (2021). The impact of stakeholders in agile software development. The Annals of the University of Oradea, Economic Sciences, 30(2), 353–362.

Eyeregba, M. E., Mokogwu, C., Ewim, S. E., & Olorunyomi, T. D. (2024). Developing a framework for business analysts to bridge stakeholder and technical team gaps: Enhancing collaboration and project success. International Journal of Applied Research in Social Sciences, 6(12), 2847–2865. https://doi.org/10.51594/ijarss.v6i12.1747

Fávero, L. F., Almeida, N. R. D., & Affonso, F. J. (2025). A Systematic mapping study on the modernization of legacy systems to microservice architecture. Appl. Syst. Innov., 8, 86. https://doi.org/10.3390/asi8040086

Fawzy, A., Tahir, A., Galster, M., & Liang, P. (2025). Exploring data management challenges and solutions in agile software development: A literature review and practitioner survey. Empirical Software Engineering, 30, Article 77. https://doi.org/10.1007/s10664-025-10630-4

Fissalma, H., Ferdinansyah, A., & Purwandari, B. (2024). Investigating challenges in Agile software development: A cross-country comparative analysis. International Journal of Electrical and Computer Engineering, 15(1), 855–869. https://doi.org/10.11591/ijece.v15i1.pp855-869

Gadesha, V., & Kavlakoglu, E. (n.d.). What is prompt chaining? IBM. https://www.ibm.com/think/topics/prompt-chaining

Garcia, M. H., Couturier, C., Diaz, D. M., Mallick, A., Kyrillidis, A., Sim, R., Rühle, V., & Rajmohan, S. (2025). Exploring how LLMs capture and represent domain-specific knowledge Preprint. arXiv. https://arxiv.org/abs/2504.16871

Geetha, L. S., El-Ebiary, Y. A. B., Rao, B. S., Rautrao, R. R., Rao, T. S. M., Ramesh, J. V. N., & Al-Omari, O. (2025). Challenges and solutions in agile software development: A managerial perspective on implementation practices. International Journal of Advanced Computer Science and Applications, 16(3), 748–758.

Ghimire, D., & Charters, S. (2022). The impact of agile development practices on project outcomes. Software, 1(3), 265–275. https://doi.org/10.3390/software1030012

González Moyano, C., Pufahl, L., Weber, I., & Mendling, J. (2022). Uses of business process modeling in agile software development projects. Information and Software Technology, 152, 107028. https://doi.org/10.1016/j.infsof.2022.107028

He, J., Treude, C., & Lo, D. (2025). LLM-based multi-agent systems for software engineering: Literature review, vision and the road ahead. ACM. https://doi.org/10.48550/arXiv.2404.04834

Hoda, R., & Murugesan, L. K. (2016). Multi-level agile project management challenges: A self-organizing team perspective. Journal of Systems and Software, 117, 245–257. https://doi.org/10.1016/j.jss.2016.02.049

Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Zhang, C., Wang, J., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., Ran, C., Xiao, L., Wu, C., & Schmidhuber, J. (2024). MetaGPT: Meta programming for a multi-agent collaborative framework Conference paper. ICLR 2024. arXiv. https://arxiv.org/abs/2308.00352

Itzik, D., Roy, G., & The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature. (2023). Does agile methodology fit all characteristics of

software projects? Review and analysis. In Empirical Software Engineering (Vol. 28, pp. 105–105). https://doi.org/10.1007/s10664-023-10334-7

Jevons, W. S. (1865). The coal question: An inquiry concerning the progress of the nation, and the probable exhaustion of our coal-mines. Macmillan and Co.

John, J. J., & Sharma, S. S. (2024). A comparative study of agile and waterfall software development methodologies. International Journal of Advanced Research in Science, Communication and Technology, 4(2), 54–57. https://doi.org/10.48175/IJARSCT-15207

Joshi, S. (2025). Comprehensive review of Artificial General Intelligence (AGI), Agentic AI and GenAI: Current trends and future directions. International Journal of Multidisciplinary Research and Growth Evaluation, 6(3), 681–688. https://doi.org/10.54660/IJMRGE.2025.6.3.681-688

Kotaiah, B., & Khalil, M. A. (2017). Approaches for development of software projects: Agile methodology. International Journal of Advanced Research in Computer Science, 8(1), 237–242.

Lercher, A. (2024). Managing API evolution in microservice architecture. In 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion '24) (pp. 195–197). ACM. https://doi.org/10.1145/3639478.3639800

Lercher, A., Glock, J., Macho, C., Pinzger, M., & Department of Informatics Systems, University of Klagenfurt. (2023). Microservice API Evolution in Practice: A Study on Strategies and challenges [Journal-article]. arXiv. https://arxiv.org/abs/2311.08175v1

Lipmanowicz, H., & McCandless, K. (n.d.). 1. 1-2-4-All. Liberating Structures. https://www.liberatingstructures.com/1-1-2-4-all/

Lipmanowicz, H., & McCandless, K. (n.d.). 7. 15% Solutions. Liberating Structures. https://www.liberatingstructures.com/7-15-solutions/

Marinho, M., Camara, R., Sampaio, S., Department of Computer Science (DC), Federal Rural University of Pernambuco (UFRPE), Recife, Pernambuco 52171-900, Brazil, & Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brazil. (2021). Toward unveiling how SAFE Framework supports agile in global software development [Journal-article]. https://doi.org/10.1109/ACCESS.2021.3101963

Mishra, A., Alzoubi, Y. I., & The Author(s). (2023). Structured software development versus agile software development: a comparative analysis. Int J Syst Assur Eng Manag, 14–14(4), 1504–1522. https://doi.org/10.1007/s13198-023-01958-5

Munteanu, V. P., & Dragos, P. (2021). The case for agile methodologies against traditional ones in financial software projects. European Journal of Business and Management Research, 6(1), 134–141. https://doi.org/10.24018/ejbmr.2021.6.1.741

Narajala, V. S., & Narayan, O. (2025). Securing agentic AI: A comprehensive threat model and mitigation framework for generative AI agents
Preprint. arXiv. https://arxiv.org/abs/2504.19956

Ndlela, M., & Tanner, M. (2023). Business analysts' contributions to the dynamic capabilities of agile software development teams. Information Technology & People, 36(8), 1–20. https://doi.org/10.1108/ITP-08-2021-0656

Nguyen, M. H., Phan, T. C., Nguyen, P. X., & Bui, N. D. Q. (2024). AGILECODER: Dynamic collaborative agents for software development based on Agile methodology Preprint. arXiv. https://arxiv.org/abs/2406.11912v2

Rasheed, A., Zafar, B., Shehryar, T., Aslam, N. A., Sajid, M., Ali, N., Dar, S. H., & Khalid, S. (2021). Requirement engineering challenges in agile software development. Mathematical Problems in Engineering, 2021, Article 6696695. https://doi.org/10.1155/2021/6696695

Radwan, A. M., Abdel-Fattah, M. A., & Mohamed, W. (2024). AI-driven prioritization techniques of requirements in Agile methodologies: A systematic literature review. International Journal of Advanced Computer Science and Applications, 15(9), 812–823. https://doi.org/10.14569/IJACSA.2024.0150989

Radwan, A. M., Abdel-Fattah, M. A., & Mohamed, W. (2025). Smart agile prioritization and clustering: An AI-driven approach for requirements prioritization. IEEE Access, 13, 127335–127349. https://doi.org/10.1109/ACCESS.2025.3589959

Saklamaeva, V., & Pavlič, L. (2024). The potential of AI-driven assistants in scaled agile software development. Applied Sciences, 14(1), 319. https://doi.org/10.3390/app14010319

Sami, M. A., Rasheed, Z., Waseem, M., Zhang, Z., Herda, T., & Abrahamsson, P. (2024). Prioritizing software requirements using large language models Preprint. arXiv. https://arxiv.org/abs/2405.01564

Savoia, A. (2019). *The right it: Why so many ideas fail and how to make sure yours succeed*. HarperOne.

Schneider, J. (2025). Generative to agentic AI: Survey, conceptualization, and challenges [Preprint]. arXiv. https://arxiv.org/abs/2504.18875

Shah, D. (2023). Agile adoption challenges in large-scale software projects. ShodhKosh: Journal of Visual and Performing Arts, 4(2), 2026–2034. https://doi.org/10.29121/shodhkosh.v4.i2.2023.3225

Shameem, M., Kumar, R. R., Nadeem, M., & Khan, A. A. (2020). Taxonomical classification of barriers for scaling agile methods in global software development environment using fuzzy analytic hierarchy process. Applied Soft Computing, 90, 106122. https://doi.org/10.1016/j.asoc.2020.106122

Show, R. (2025). *Build a multi-agent system with LangGraph: A complete tutorial on agent orchestration*. FutureSmart AI. https://blog.futuresmart.ai/multi-agent-system-with-langgraph

Singh, K. (2021). Agile Methodology for Product Development: A Conceptual study. In International Journal of Recent Technology and Engineering (IJRTE) (Vol. 10, Issue 1, pp. 209–210). Blue Eyes Intelligence Engineering and Sciences Publication. https://doi.org/10.35940/ijrte.A5899.0510121

Sporsem, T., Dingsøyr, T., & Stol, K.-J. (2025). User stories as boundary objects in agile requirements engineering: A theoretical literature review. Journal of Systems and Software. Advance online publication. https://doi.org/10.1016/j.jss.2025.112693

Truss, M. & RheinMain University of Applied Sciences. (2020). Scaling Scrum with the Scaled Agile Framework: Potentials and Importance in Large Development Organizations. In RheinMain University of Applied Sciences [Journal-article]. https://ssrn.com/abstract=4683212

van Zyl, L. (2025, February 23). n8n AI agent tutorial [Video]. YouTube. youtube.com/watch?v=o2Pubq36Pao&feature=youtu.be

Verwijs, C., Russo, D., The Liberators BV, & Department of Computer Science, Aalborg University, Copenhagen, Denmark. (2024). Do Agile scaling approaches make a difference? An empirical comparison of team effectiveness across popular scaling approaches. Empirical Software Engineering, 29, 75–75. https://doi.org/10.1007/s10664-024-10481-5

volodith. (n.d.). After 1000 hours of prompt engineering, I found the 6 patterns that actually matter [Online forum post]. Reddit. https://www.reddit.com/r/PromptEngineering/comments/1nt7x7v/after_1000_hours_of_prompt_engineering_i_found/

Wagner, N. (2010). An agile BA: A case study of the business analyst in Agile (Master's thesis, Regis University). Regis University ePublications. https://epublications.regis.edu/theses/668

Waja, G., Shah, J., Nanavati, P., & IT Department, KJSCE. (2021). AGILE SOFTWARE DEVELOPMENT. In International Journal of Engineering Applied Sciences and Technology: Vol. Vol. 5 (Issue Issue 12, pp. 73–78). https://www.ijeast.com/papers/73-78.Tesma512.IJEAST.pdf

Wang, W.-J. (2025). Achieving business scalability with composable enterprise architecture. IEEE Access, 13, 76464–76472. https://doi.org/10.1109/ACCESS.2025.3564710

Wang, Y., Kadiyala, H., & Rubin, J. (2021). Promises and challenges of microservices: an exploratory study. Empirical Software Engineering, 63. https://doi.org/10.1007/s10664-020-09910-y

Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., Zhang, M., Wang, J., Jin, S., Zhou, E., Zheng, R., Fan, X., Wang, X., Xiong, L., Liu, Q., Zhou, Y., Wang, W., Jiang, C., Zou, Y., . . . Gui, T. (2023). The rise and Potential of large Language Model Based Agents: A survey. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2309.07864

Yahya, N., & Maidin, S. S. (2023). Hybrid agile development phases: The practice in software projects as performed by software engineering team. Indonesian Journal of Electrical Engineering and Computer Science, 29(3), 1738–1749. https://doi.org/10.11591/ijeecs.v29.i3.pp1738-1749