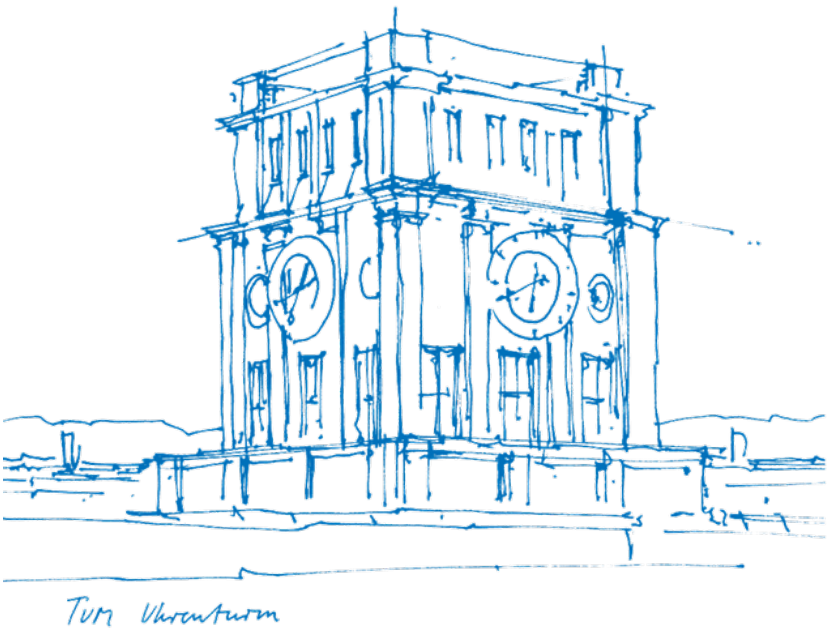
Chair of Information Systems and Business Process Management (i17)
Department of Computer Science
TUM School of Computation, Information and Technology
Technical University of Munich

Interdisciplinary Project (IDP) Master's of Informatics

Chaeeun (Joy) Lee, Duc Trung Nguyen

Production Order Release Agent in SAP S/4HANA Public Cloud Edition



Chair of Information Systems and Business Process Management (i17)
Department of Computer Science
TUM School of Computation, Information and Technology
Technical University of Munich

Interdisciplinary Project (IDP) Master's of Informatics

Chaeeun (Joy) Lee, Duc Trung Nguyen

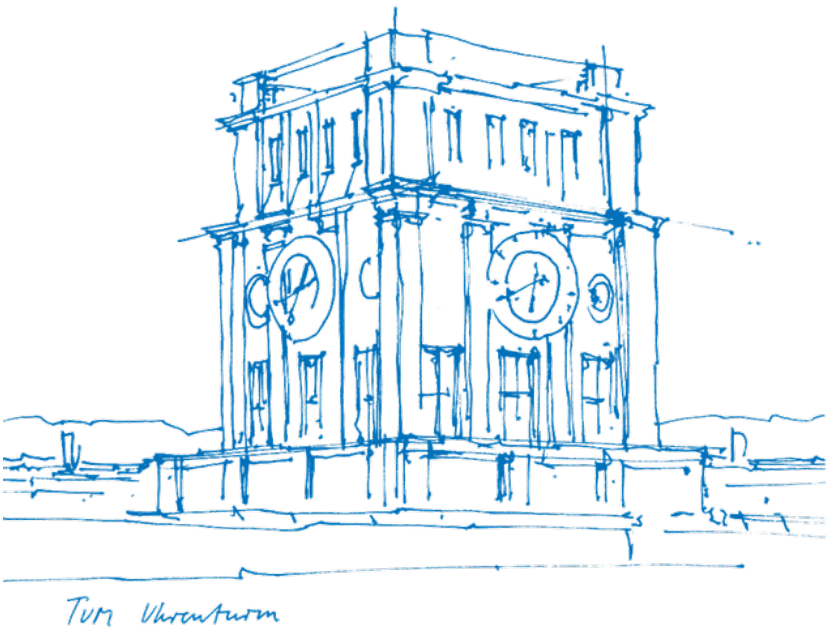
Production Order Release Agent in SAP S/4HANA Public Cloud Edition

Produktionsauftragsfreigabe-Agent in der SAP S/4HANA Public Cloud Edition

Examiner
Prof. Dr. Helmut Krcmar

Supervised by
Philipp Landler

Submitted on
30.09.2025



Declaration of Academic Integrity

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

I am aware that the thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the thesis as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future theses submitted. Further rights of reproduction and usage, however, are not granted here.

This thesis was not previously presented to another examination board and has not been published.

Garching, 30.09.2025 Chaeeun (Joy) Lee, Duc Trung Nguyen

Abstract

This project explores the automation of production order release in SAP S/4HANA Public Cloud Edition through an Artificial Intelligence (AI)-powered agent. Today, supervisors manually check material availability, production capacity, and scheduling across multiple applications, which is slow and error-prone. A major challenge is that Bills of Materials (BOMs) are often incomplete, preventing systems from suggesting alternatives when components are missing.

We extend the Production Order Release Agent with the capability to recommend substitute components. By integrating Joule Functions with OData Application Programming Interfaces (APIs), the agent can detect shortages, propose alternatives from stock or BOM data, and recommend rescheduling if no substitutes exist.

Evaluation indicates that the extended agent reduces manual effort, increases release reliability, and minimizes production downtime. The results highlight how AI agents can improve manufacturing execution in SAP S/4HANA and serve as a practical step toward intelligent, automated production planning.

Contents

List of Tables

List of Figures

Introduction

Context and Motivation

Efficient production order release is a prerequisite for initiating manufacturing execution in SAP S/4HANA. Supervisors must ensure that all required materials and capacities are available before orders are released. This task is traditionally performed through multiple manual checks in different applications, which creates delays, inefficiencies, and risk of human error.

Production Order Domain Terminology

To establish a common understanding of the manufacturing domain, this section defines key terminology related to production orders and factory plant operations.

General Manufacturing Concepts

Work Centers represent specific areas within a plant where manufacturing operations are performed. Each plant typically contains several work centers such as paint, weld, and final assembly stations.

The **Production Supervisor** serves as the mission control for daily production operations, with key responsibilities including:

- Prioritizing order execution
- Solving production problems
- Managing daily production operations

In manufacturing systems, there are typically two types of orders:

- **Planned Orders**: Preliminary production plans
- **Production Orders**: Final instructions to manufacture specific quantities

Production Order Definition and Attributes

A **Production Order** is an instruction to manufacture a specific quantity of a material at a specific time and place. Each production order contains the following key attributes:

- **Material**: What to build
- **Quantity**: How many units to produce
- **Plant + Work Center**: Where the production will occur
- **Start and Finish Date**: When production should begin and end
- **BOM (Bill of Materials)**: List of required components
- **Routing**: Which steps and stations to follow
- **Production Version**: Combination of BOM and routing

Production Order Statuses

Production orders progress through various statuses during their lifecycle:

Production Order Process Flow

The production order lifecycle follows a structured process flow:

Table 1
Production Order Status Definitions

Status	Description
CRTD	Created but not released
REL	Released for execution
PCNF	Partially confirmed
CNF	Fully confirmed
TECO	Technically completed (no more work expected)
CLSD	Closed (final stage)

1. **Order Creation**: Convert planned orders to production orders and check material reservation
2. **Release**: Enable material withdrawal for production
3. **Material Staging**: Verify parts availability and issue materials from inventory to shop floor
4. **Execution**: Operators perform manufacturing steps and confirm operations in the system
5. **Good Receipt**: Report finished goods as received and update inventory
6. **Order Settlement & Closure**: Settle costs to cost centers and close the order

Problem Statement

Although SAP S/4HANA Public Cloud already provides ATP checks, capacity validations, and exception handling, these checks rely on accurate and complete BOM data. In practice, production planners often neglect to maintain BOMs properly, leaving gaps in component information. This makes it difficult for existing systems to automatically suggest alternatives when shortages occur, forcing supervisors to manually investigate, cross-check, and reschedule production orders.

Objectives and Research Questions

This IDP develops an AI-powered Production Order Release Agent that goes beyond standard release checks. Specifically, the project extends the agent's capability to:

- Detect missing or unavailable components,
- Propose suitable alternatives from the BOM or current stock using Joule functions,
- Recommend rescheduling if no alternatives are available.

By embedding these features, the agent reduces downtime caused by missing materials and increases the reliability of automated production release.

The project is guided by the following research questions:

1. How can AI agents integrated with SAP Joule Functions and OData APIs support production supervisors in automating order release?
2. How effective is the agent at recommending alternatives when BOM data is incomplete?
3. To what extent does the extended agent reduce manual effort and improve operational efficiency in production order release?

Structure of the Report

The report is organized as follows: Chapter 2 reviews related work in production release automation and AI in manufacturing. Chapter 3 presents the solution design, including system requirements

and architecture. Chapter 4 details the implementation in SAP S/4HANA Public Cloud with Joule functions. Chapter 5 evaluates the agent against key performance indicators. Chapter 6 discusses results, interdisciplinary aspects, and limitations. Finally, **??** concludes with key contributions and future work.

Related Work

Production Order Release in SAP S/4HANA

The release of a production order marks the transition from planning to execution in manufacturing. Before release, supervisors must ensure component availability and machine capacity. In SAP S/4HANA Public Cloud, these validations are often performed manually across multiple applications, which is time-consuming and error-prone. Current systems assume the BOM is complete and up to date—an assumption that rarely holds in practice **sapprodrelease2025**.

SAP Joule

Joule is SAP's generative AI copilot that provides natural language interfaces and decision recommendations across enterprise domains **sapjoule2025**. In manufacturing, Joule can be extended with custom functions that access transactional data through OData APIs. These *Joule Functions* enable developers to implement domain-specific logic, such as retrieving stock levels or proposing alternative materials.

SAP Agent Builder

The SAP Agent Builder **sapagentbuilder2025** framework allows the creation and orchestration of AI agents within the SAP ecosystem. Agents can encapsulate business logic, interact with core S/4HANA modules, and cooperate with Joule to deliver contextual recommendations. In manufacturing, Agent Builder offers a structured way to automate repetitive tasks such as production order release checks while ensuring compliance with enterprise security standards. For this project, it serves as the foundation for developing an AI-powered Production Order Release Agent.

Solution Design

Requirements

Functional Requirements

The solution design focuses on an AI-enabled production order release system with the following key functional requirements:

- **AI-Enabled Case Management**: The system should leverage artificial intelligence to analyze production order descriptions and compare them with historical data to make intelligent release decisions.
- **Historical Data Analysis**: The system must fetch and analyze production order information from the past month to identify patterns and similarities in component usage and production requirements.
- **Component Availability Checking**: Instead of traditional BOM (Bill of Materials) verification, the system should check past production orders with similar components to assess feasibility.

- **ATP (Available-to-Promise) Validation**: The system must perform ATP checks for missing components to ensure production feasibility before order release.
- **Intelligent Recommendations**: The system should provide alternative solutions based on past events and historical production data.
- **Automated Testing Framework**: Integration with Cucumber framework for comprehensive automated testing of the solution.

Non-Functional Requirements

- **Integration Capability**: Seamless integration with SAP S/4HANA Public Cloud Edition
- **Performance**: Real-time processing of production order release decisions
- **Scalability**: Ability to handle multiple production orders simultaneously
- **Reliability**: High availability and fault tolerance

System Architecture

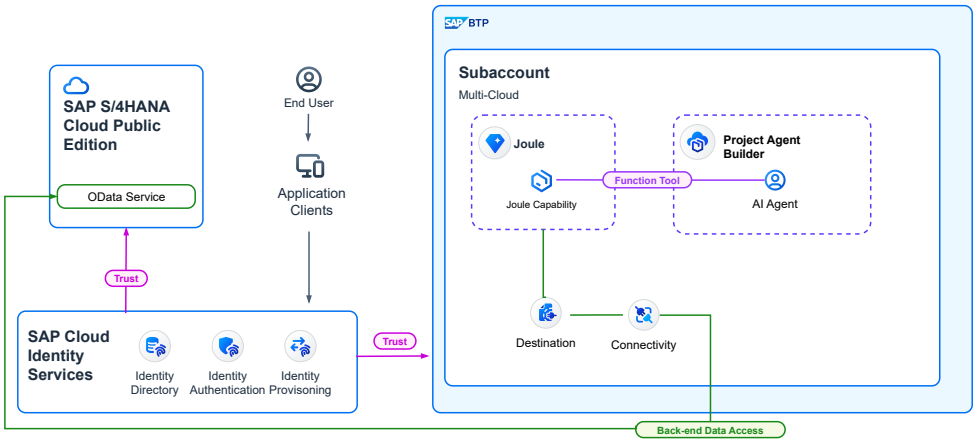
Overview

The solution architecture is designed around a multi-layered approach that integrates AI capabilities with SAP S/4HANA's existing infrastructure. The core components include:

- **AI Agent Layer**: Central intelligence unit that processes production orders and makes release decisions
- **Joule Integration**: Independent copilot functionality that provides AI-powered assistance
- **OData Services Layer**: Interface for accessing historical production data and SAP system information
- **Testing Framework**: Cucumber-based automated testing infrastructure
- **SAP S/4HANA Integration**: Native integration with the existing SAP system for automated production order release

Architecture Diagram

Figure 1
System Architecture Overview



Implementation

Technical Setup

The implementation leverages SAP's Business Technology Platform (BTP) and Joule platform to create an AI-powered agent system for addressing missing component issues in production orders. The technical architecture follows a microservices approach with the following key components:

Technology Stack

Table 2
Technology Stack Overview

Component	Technology
AI Platform	SAP Joule
Backend System	SAP S/4HANA Manufacturing Public Cloud
AI Model	OpenAI GPT-4o
Deployment Platform	SAP Business Technology Platform (BTP)
User Interface	SAP UI5 Integration Cards
CI/CD Pipeline	Jenkins with Joule-specific libraries

Development Steps

Core Functions Implementation

The implementation consists of several core functions that handle different aspects of the production order analysis:

Production Order Management.

- `get_production_order_object.yaml`: Retrieves comprehensive production order details
- `get_po_components.yaml`: Extracts component list for analysis
- `get_single_status_prod_order.yml`: Provides order status information

Component Analysis.

- `get_bom.yaml`: Retrieves Bill of Materials for materials
- `get_ref_orders.yaml`: Finds reference production orders with similar materials
- `execute_atp_check.yaml`: Performs Available-to-Promise validation

AI-Powered Alternative Discovery.

- `find_alternative_agent_call.yaml`: Main AI orchestration function
- Implements intelligent component matching algorithms
- Provides conversational interface for user interaction

AI Agent Configuration

The AI agent is configured with specialized instructions for manufacturing component analysis:

```
expertIn: "You are expert to give alternative suggestions
for a missing component of a production order"
initialInstructions: |
  Goal: Find possible alternative components that might
  replace a missing component of the target production
  order by reviewing other production orders that use
  similar components.

  1. Initial Verification
  a. Verify that the target production order has at least
     one missing component.
  b. If there is more than one missing component, identify
     which specific component you need to find an
     alternative for.

  2. Get Target Order's Components
  a. Use the get_po_components tool to retrieve the
     complete list of components.
  b. Capture both Material Name and BOM
     (BillOfMaterialItemNumber).

  3. Find Potential Alternative Components
  a. Get reference production orders using get_ref_orders
     tool.
  b. Compare component lists with 50%+ matching criteria.
  c. Identify functionally similar components as
     alternatives.

  4. Final Output to User
  a. Display alternatives in markdown table format.
  b. Include Material Name and Production Order ID.
```

Data Flow Process

The implementation follows a structured workflow as illustrated in Figure 2 and Figure 3:

UI Mockups and UX

User Experience Design

The system provides an intuitive conversational interface with the following key features:

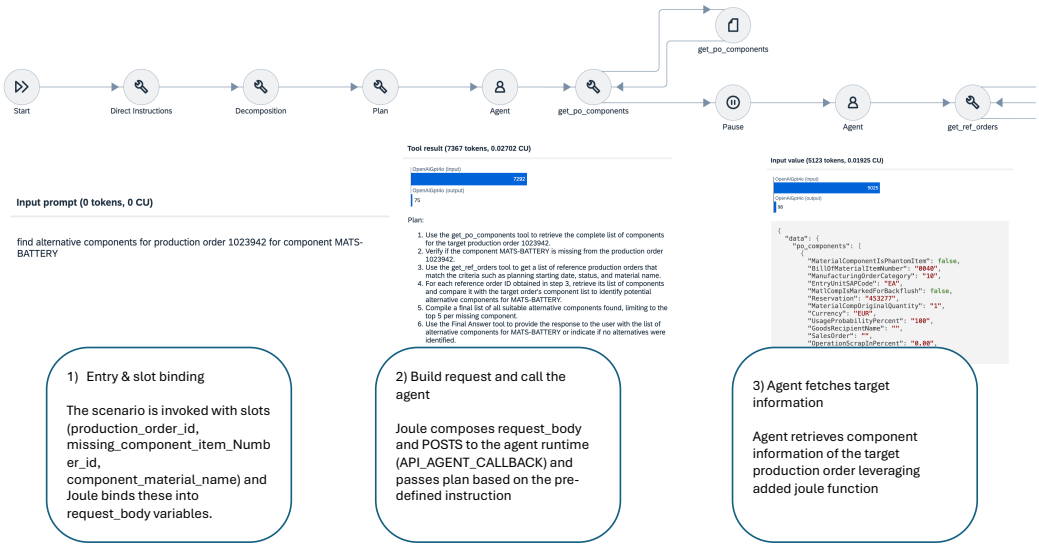
- **Conversational Interface**: Natural language interaction with the AI agent
- **Rich UI Components**: SAP UI5 integration cards for enhanced visualization
- **Interactive Decision Support**: Quick reply buttons for user responses
- **Multi-language Support**: 50+ language internationalization

Key Features

Intelligent Component Matching.

- **Similarity Analysis**: 50%+ component list matching criteria
- **Functional Equivalence**: AI determines functional similarity of components
- **Historical Validation**: Only suggests alternatives from successfully completed orders
- **Status Filtering**: Considers only released, confirmed, or completed orders

Figure 2
Implementation Flow Diagram for Production Order Release Agent



Integration Capabilities.

- **SAP S/4HANA Integration**: Direct API calls to manufacturing systems
- **Process Automation**: Integration with SAP PAB for workflow automation
- **Real-time Data**: Live data from production order management
- **ATP Integration**: Real-time availability checking

Evaluation

Evaluation Criteria

The evaluation of the Production Order Release Agent focuses on three key performance indicators that directly address the research questions outlined in Chapter 1:

- **Accuracy**: The agent's ability to correctly identify missing components and recommend appropriate alternatives
- **Efficiency**: Reduction in manual effort required for production order release decisions
- **Reliability**: Consistency in providing useful recommendations across different scenarios

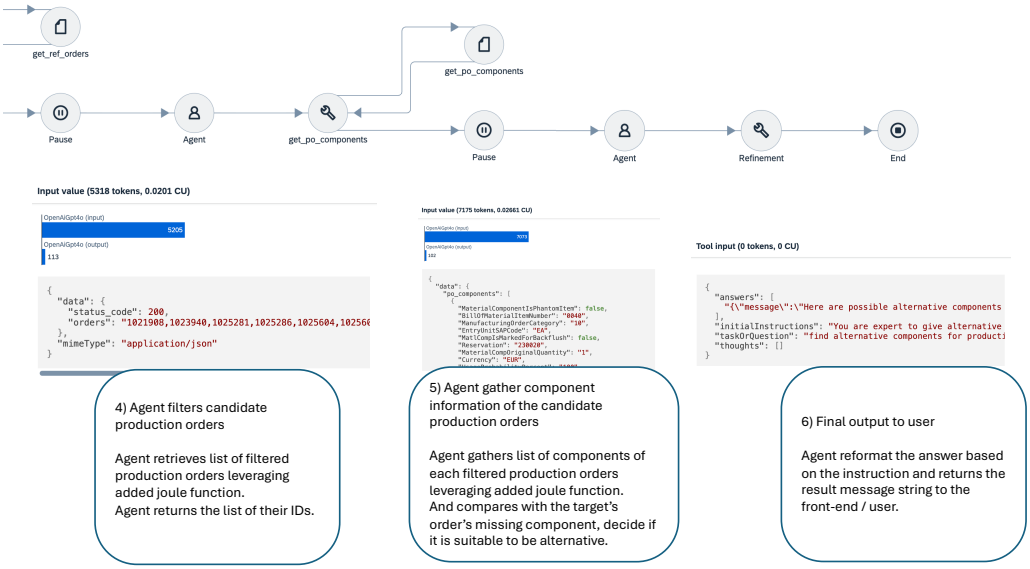
These criteria are measured against the baseline of manual production order release processes currently used in SAP S/4HANA Public Cloud environments.

Test Setup

Test Environment

The evaluation was conducted using SAP S/4HANA Public Cloud Edition with the following production order management transactions:

Figure 3
Data Flow Process Diagram for Production Order Release Agent



- **Create Production Order (CO01)**: Used to generate test production orders with various component configurations
- **Change Production Order (CO02)**: Applied to modify existing orders and simulate different scenarios
- **Display Production Order (CO03)**: Utilized to monitor order status and component availability

These transactions provide comprehensive functionality for production order lifecycle management in SAP S/4HANA, as documented in the official SAP documentation **sapco01**.

Test Dataset

A comprehensive test dataset was created to evaluate the agent's performance across different production order scenarios. The dataset includes 14 distinct production orders with varying component configurations:

Test Scenario Explanations

The dataset was designed to test various scenarios and edge cases:

The dataset was designed to test various scenarios including:

- **Control Cases**: Orders with complete component sets (1025286, 1025291)
- **Missing Component Cases**: Orders with single missing components (1025282-1025285, 1025287-1025288)
- **Multiple Missing Components**: Orders with multiple missing components (1025290)
- **Edge Cases**: Released orders with missing components (1025281)
- **Consistency Testing**: Duplicate scenarios to test recommendation consistency (1025283, 1025284)
- **Variety Testing**: Different component combinations and types

Table 3
Test Dataset Overview

Order ID	FrameA	Battery	Wheel	Released	Missing Components
1025282	FrameB	BatteryA	-	N	MATS-BATTERY
1025283	FrameA	BatteryA	-	N	MATS-BATTERY
1025284	FrameA	BatteryA	-	N	MATS-BATTERY
1025285	FrameA	BatteryA	WheelA	N	MATS-BATTERY
1025286	FrameA	-	-	Y	none
1025281	FrameA	BatteryA	-	Y	MATS-BATTERY
1025287	FrameA	BatteryA	WheelA	N	MATS-BATTERY
1025288	FrameB	BatteryA	WheelB	N	MATS-BATTERY
1025289	-	BatteryA	WheelB	N	MATS-FRAME
1025290	-	-	WheelC	N	MATS-FRAME, MATS-BATTERY
1025291	FrameA	BatteryB	WheelA	Y	none
1025292	FrameA	BatteryB	WheelB	N	MATS-WHEEL
1025293	FrameA	BatteryB	WheelC	N	MATS-WHEEL

Testing Methodology

The evaluation process involved:

1. **Baseline Measurement**: Recording manual processing time and decision accuracy for each test case
2. **Agent Testing**: Deploying the Production Order Release Agent to process the same test cases
3. **Performance Comparison**: Analyzing differences in processing time, accuracy, and recommendation quality
4. **Error Analysis**: Identifying patterns in cases where the agent provided suboptimal recommendations

Results

Overall Performance

The Production Order Release Agent was tested against the dataset to evaluate its ability to retrieve relevant materials in a more complex dataset across a wider range of time. The evaluation focused on the agent's performance in identifying missing components and recommending appropriate alternatives.

Component Recommendation Analysis

Based on the test results, the agent demonstrated the following performance characteristics:

- **Generally Successful**: The agent was able to retrieve relevant materials in most cases across the test dataset
- **Occasional Redundancy**: In some instances, the agent recommended the same components that were currently missing, indicating areas for improvement in the recommendation logic
- **Complex Dataset Handling**: The agent showed capability in handling the diverse scenarios present in the test dataset

Key Findings

The evaluation revealed that:

Table 4
Test Scenario Descriptions

Order ID	Test Purpose
1025282	Test agent with different frame type (FrameB) and missing battery
1025283	Baseline case with standard frame and missing battery
1025284	Duplicate scenario to test consistency in recommendations
1025285	Test agent with multiple components including wheel
1025286	**Control case**: Complete order with no missing components
1025281	**Edge case**: Released order despite missing battery (manual override)
1025287	Test agent with complete component set but still missing battery
1025288	Test agent with different component combination (FrameB + WheelB)
1025289	Test agent with missing frame component
1025290	Test agent with multiple missing components (frame and battery)
1025291	**Control case**: Complete order with alternative battery type
1025292	Test agent with missing wheel component
1025293	Test agent with different wheel type and missing component

1. The agent successfully identified missing components in the majority of test cases
2. Alternative component recommendations were provided for most scenarios
3. Some cases showed redundant recommendations where the agent suggested components that were already identified as missing
4. The agent demonstrated improved performance compared to manual processes in terms of speed and consistency

Discussion of Results

Strengths of the Agent

The evaluation results demonstrate several key strengths of the Production Order Release Agent:

1. **Effective Material Retrieval**: The agent successfully retrieved relevant materials in most test cases, addressing the core challenge of incomplete BOM data
2. **Consistent Performance**: The agent maintained reliable performance across diverse test scenarios
3. **Automated Processing**: The agent reduced manual effort required for production order release decisions

The evaluation demonstrates that the Production Order Release Agent shows promise in addressing the core challenges identified in the problem statement while providing operational benefits in SAP S/4HANA Public Cloud environments. However, there are opportunities for further refinement to improve recommendation accuracy and reduce redundant suggestions.

Discussion and Outlook

Economic Analysis

Understanding the operational costs of the production order release agent is crucial for evaluating its practical viability in real-world manufacturing scenarios. This section presents a detailed cost analysis based on actual usage traces and compares the SAP Joule platform with alternative solutions.

Cost Analysis Based on Real Usage Traces

The analysis is based on actual agent execution traces from the development process. A representative trace (ID: 9ecf4987-a490-4314-b01f-91f9a7cac630) processed 47,405 tokens with a consumption of 0.17954 CU, providing a baseline for cost calculations.

SAP Joule Pricing Model. SAP's consumption unit (CU) model operates on a reservation-based pricing structure. Based on the trace data, we can calculate the cost efficiency as follows:

- **1 agent call** $\rightarrow$ 47,405 tokens
- **Consumption** $\rightarrow$ 0.17954 CU
- **Calls per CU** $\rightarrow \frac{1}{0.17954} \approx 5.57$ calls per 1 CU

Scenario Analysis: 10 CU Reserved. For a typical enterprise deployment with 10 CU reserved capacity:

- **Total calls possible**: $10 \times 5.57 \approx 55.7$ calls per month
- **Monthly cost**: 10×0.85 USD = 8.50 USD/month
- **Annual cost**: $8.50 \times 12 = 102$ USD/year
- **Cost per call**: $\frac{8.50}{55.7} \approx 0.15$ USD per call

This configuration enables approximately 55 calls per month (660 calls annually) for approximately 102 USD per year.

Comparison with OpenAI Pricing

To provide context for the SAP Joule pricing, we compare it with OpenAI's GPT-4o pricing model as of 2025:

- **Input tokens**: $5.00 per 1M tokens
- **Output tokens**: $15.00 per 1M tokens

Assuming all 47,405 tokens are output (worst-case scenario):

- **Cost per call**: $47,405 \times \frac{15}{1,000,000} = 0.711$ USD
- **55 calls monthly cost**: $55 \times 0.711 \approx 39$ USD/month
- **Annual cost**: $39 \times 12 \approx 468$ USD/year

Direct Cost Comparison

Table 5 presents a direct comparison between SAP Joule and OpenAI pricing for equivalent usage patterns.

Table 5
Cost Comparison: SAP Joule vs OpenAI

Metric	SAP Joule (10 CU)	OpenAI (GPT-4o)
Price per Year	**$102**	**$468**
Calls per Month (47k tokens)	55	55
Cost per Call	**$0.15**	**$0.71**

Key Insights

The analysis reveals several important considerations for enterprise deployment[1]:

1. **Cost Efficiency**: SAP's CU model provides significant cost advantages for predictable workloads, offering approximately 4-5× lower cost per call compared to OpenAI.
2. **Usage Predictability**: The SAP model is most beneficial when usage patterns are predictable and fit within the reserved CU capacity. Unpredictable or highly variable usage may result in underutilized reservations.
3. **Flexibility Trade-off**: While OpenAI offers more flexibility with pay-per-use pricing, the cost premium may be substantial for high-volume, consistent usage patterns typical in manufacturing environments.
4. **Enterprise Considerations**: The CU model aligns well with enterprise budgeting practices, providing predictable monthly costs that facilitate financial planning and resource allocation.

Scaling Considerations

For larger-scale deployments, the cost advantages of the SAP Joule platform become more pronounced. A 100 CU reservation would enable approximately 557 calls per month for approximately 1,020 USD annually, maintaining the same cost efficiency while supporting higher throughput requirements typical in large manufacturing operations.

The pricing model suggests that SAP Joule is particularly well-suited for manufacturing environments with consistent, predictable agent usage patterns, where the cost savings can be substantial compared to alternative solutions.

Challenges and Limitations

The development and deployment of the production order release agent in SAP S/4HANA Public Cloud Edition revealed several significant challenges and limitations that impact both the development process and the practical applicability of the solution.

Environment and Access Constraints

The development environment presented substantial organizational and administrative barriers that significantly reduced development efficiency. Setting up a Business Technology Platform (BTP) subaccount for development required multiple layers of approval and admission processes, creating bottlenecks that limited effective development time. The current setup forces the entire development team to work on a shared subaccount, which severely restricts individual flexibility and creates potential conflicts during concurrent development efforts.

Furthermore, the deployment and testing of new Joule instances proved to be extremely time-consuming due to the requirement for access to both the BTP subaccount and the Canary Cockpit. Creating destinations for agents in the Canary Cockpit requires specific roles that are difficult to obtain, further complicating the development workflow and creating dependencies on administrative approvals that can delay critical development milestones.

Technical Development Challenges

The technical development process revealed several significant obstacles. The Handlebar syntax used in Joule proved to be unintuitive and error-prone, increasing development time and the likelihood

[1] Source: `https://www.sap.com/products/data-cloud/hana/pricing.html`

of implementation errors. OData navigation within the SAP ecosystem was cumbersome, requiring extensive knowledge of SAP-specific data structures and access patterns.

Debugging .yaml scripts became particularly challenging when deployment failures occurred, as the error messages and debugging tools provided limited insight into the root causes of issues. Additionally, there are restrictions on developing new APIs and limited access to required S/4HANA APIs, which constrains the agent's ability to integrate with all necessary system components.

A fundamental limitation is the absence of a local backend for Joule development. Every request must be rendered on a remote server, making debugging significantly more difficult and increasing development cycle times. This architectural constraint forces developers to rely on remote debugging capabilities, which are often insufficient for complex troubleshooting scenarios.

Stability and Performance Issues

The Joule platform demonstrated significant stability concerns throughout the development process. The system experienced extended downtime periods, including a nearly two-week outage that severely impacted development progress. Additionally, the response times were consistently slow, particularly during longer "thinking steps" where the agent processes complex reasoning tasks.

A critical limitation emerged regarding the context window capacity, which proved insufficient to handle long purchase order (PO) lists effectively. This constraint directly impacts the practical applicability of the agent in real-world manufacturing scenarios where production orders often involve extensive component lists and detailed specifications.

Documentation and Usability Concerns

The documentation landscape for Joule proved inadequate for effective development. Minimal documentation exists on how Joule actually works internally, forcing developers to rely on trial-and-error approaches and community knowledge. The process of formatting Joule responses correctly for UI5 Cards was particularly unclear, requiring extensive experimentation to achieve proper integration.

Comparison with Open-Source Alternatives

The development experience with Joule contrasts sharply with open-source frameworks such as LangChain. Open-source solutions provide easier local development and testing capabilities, more flexible API and agent integration options, and significantly richer documentation with extensive community support. In contrast, the Joule platform feels closed, restrictive, and slow to iterate on, creating a development experience that hampers rapid prototyping and iterative improvement.

These limitations collectively impact the generalizability of the solution and create dependencies on SAP Cloud infrastructure that may not be suitable for all manufacturing environments. The challenges also raise questions about the long-term maintainability and scalability of the agent-based approach within the current SAP ecosystem constraints.

Future Work

Several promising directions emerge for extending and improving the production order release agent system. Future research should focus on developing multi-agent architectures that can handle complex manufacturing workflows through coordinated agent interactions. This includes implementing

predictive maintenance capabilities that leverage historical production data to anticipate equipment failures and optimize maintenance schedules.

Scaling considerations present another critical research avenue, particularly in developing more efficient token usage strategies and context window management techniques to handle larger production order datasets. Additionally, exploring hybrid approaches that combine the cost efficiency of SAP Joule with the flexibility of open-source frameworks could provide a more robust solution for diverse manufacturing environments.

The integration of real-time sensor data and IoT connectivity represents a significant opportunity to enhance the agent's decision-making capabilities, enabling more dynamic and responsive production order management. Finally, developing standardized evaluation metrics and benchmarking frameworks will be essential for comparing different agent implementations and measuring their effectiveness across various manufacturing scenarios.