

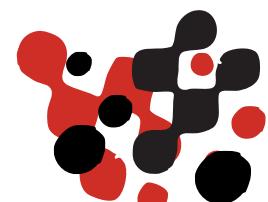


# NETEYE CONFERENCE 2025

## Unlocking Service Excellence: a deep dive into JSM Operations

From ITIL 4 Monitoring Foundations to Practical Alert Handling in Jira Service Management

Giuseppe Di Garbo, System Architect Würth Phoenix



# Agenda

- Context: Why intelligent operations matter
- ITIL 4 Monitoring & Event Management (MEM) essentials
- Jira Service Management Operations and on-call schedules
- Automation and alert flow integration
- AI-powered alert management (Rovo & AIOps)
- Real-world example: NetEye + JSM
- Key takeaways and next steps





**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**

• **Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checko**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**



**Warning: Checkout failed**

# Context and Motivation

- Modern IT = hybrid, distributed, complex
- Multiple monitoring tools, endless alerts
- Risks: alert fatigue, missed signals, silos
- Goal: move from reaction to **intelligent operations**



# ITIL4 Monitoring & Event Management Essentials

## ITIL4 MEM essentials:

- **Monitoring:** observe *continuously*
- **Event:** significant state change
- **Alert:** event requiring action
- **Benefits:** early detection, reduced downtime, reliable service

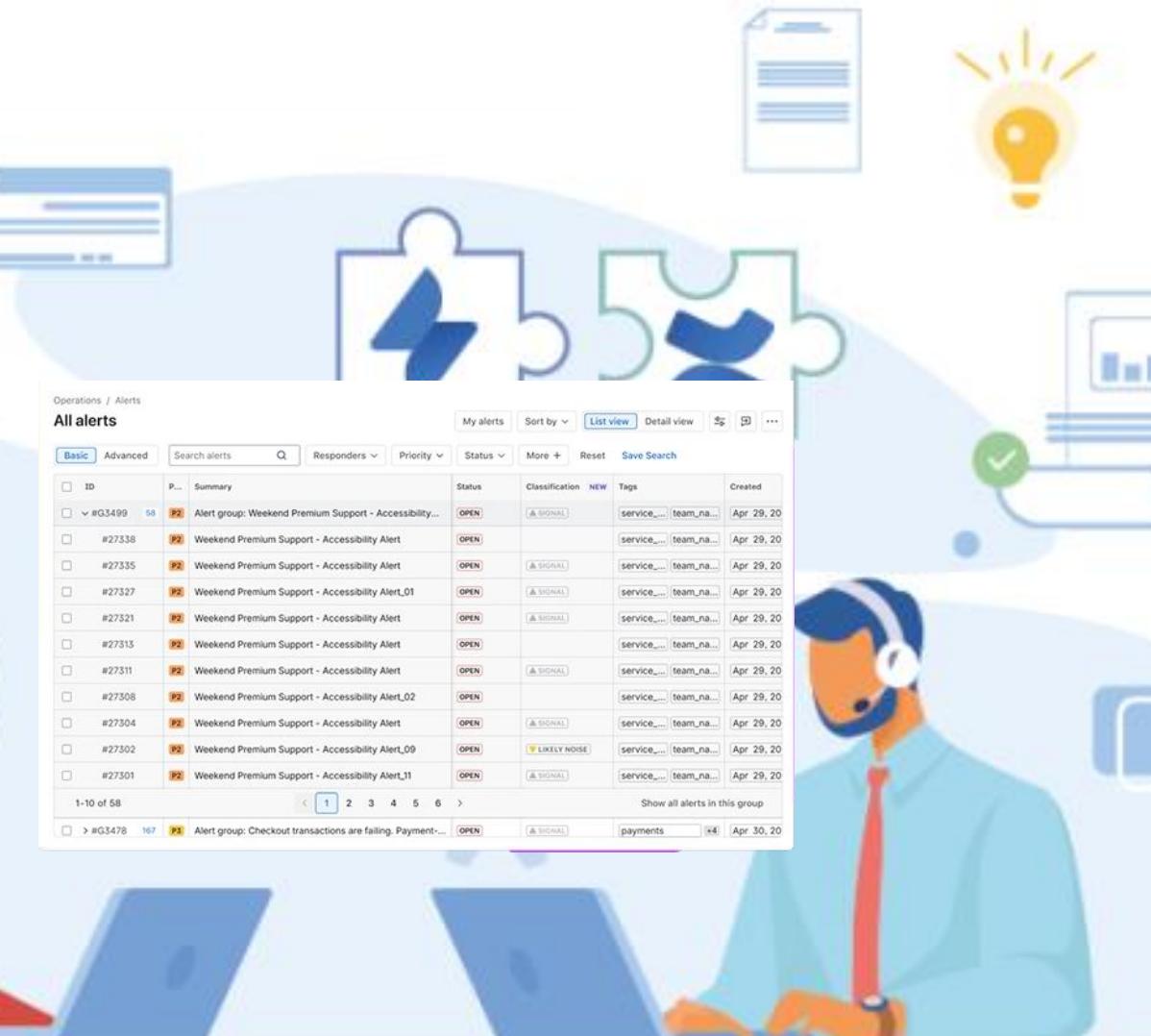
## ITIL4 MEM recommendations:

- **Filter** and **classify** events, distinguishing informative ones from those that require action;
- **Automate event correlation** to link technical events to affected services and identify root causes faster;
- **Integrate MEM with Incident Management**, ensuring that priority alerts are automatically transformed into actionable incidents.
- Balance **reactive** and **proactive** monitoring approaches

“Systematically observe services and service components, and record and report selected changes of state identified as events”

# JSM as an Operations Hub

- Monitoring: observe continuously
- **Central** workspace for operational activities
- Organize by service, technology, or region
- Integrate alerts from multiple sources
- Increase visibility and accountability



# On-Call Schedules

- Who is on call?
- Define **rotations**, **routing**, and **escalation** rules
- Ensure always-on coverage with flexible schedules
- Receive **notification** via email, SMS, phone, or Jira mobile app
- Integrate with external calendars for visibility

The Jira On-call interface displays three active schedules:

- Central Operations\_schedule**: Central Operations - (-08:00) PST America/Los\_Angeles. Enabled. Dante Rodriguez is on call. Shifts: Rot1 (Dante R), Rot2 (Dante Rodriguez, Darrel Rossi).
- Mobile Operations\_schedule**: Mobile Operations - (-08:00) PST America/Los\_Angeles. Enabled. Dante Rodriguez is on call. Shifts: Rot1 (Dante R), Rot2 (Dante Rodriguez, Darrel Rossi).
- Service Desk - Schedule**: Service Desk - (-08:00) PST America/Los\_Angeles. Enabled. Sammy Vito, Serena Mehta are on call. Shifts: Rot1 (Sammy V, SV), Rot2 (Serena M, Dante Rodriguez, Darla Cote, Oily Persz).

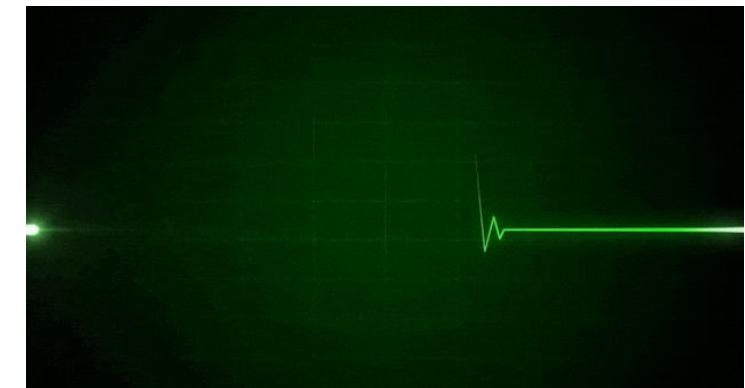
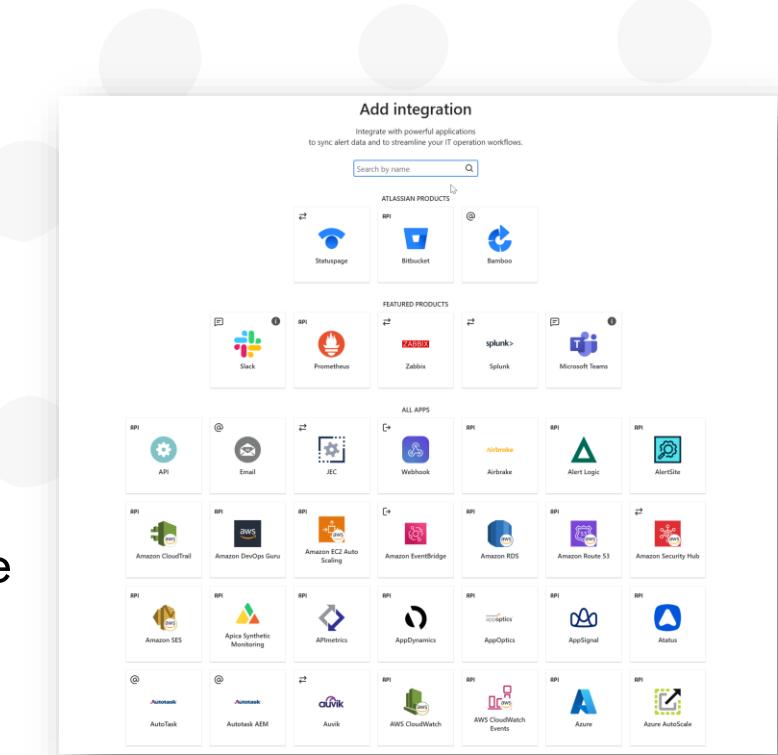
Below the schedules, the Escalation policies section lists the following rules:

- 102911-W-IT 24x7 support escalation**: 0m. On call users in 24x7\_Schedule, if not acknowledged.
- NOC 24x7 - WP Infra - support escalation**: 0m. NOC\_24x7\_WP\_Infra\_support\_schedule, if not acknowledged. 120m. Notify Fortuni, Guglielmo, if not acknowledged.
- NOC 24x7 support escalation**: 0m. On call users in 24x7\_Schedule, if not acknowledged. 120m. Notify Fortuni, Guglielmo, if not acknowledged.
- NOC 8x5 - WP Infra - support escalation**: 0m. On call users in NOC\_8x5\_support\_schedule, if not acknowledged. 60m. Next user in NOC\_8x5\_support\_schedule, if not acknowledged. 120m. Notify Villalba, if not acknowledged. 240m. Notify Cesar, if not acknowledged.
- NOC 8x5 support HIGH escalation**: 0m. On call users in NOC\_8x5\_support\_schedule, if not acknowledged. 60m. Next user in NOC\_8x5\_support\_schedule, if not acknowledged.
- NOC 8x5 support LOW escalation**: 30m. On call users in NOC\_8x5\_support\_schedule, if not acknowledged. 60m. Next user in NOC\_8x5\_support\_schedule, if not acknowledged.



# Advanced Alert Management

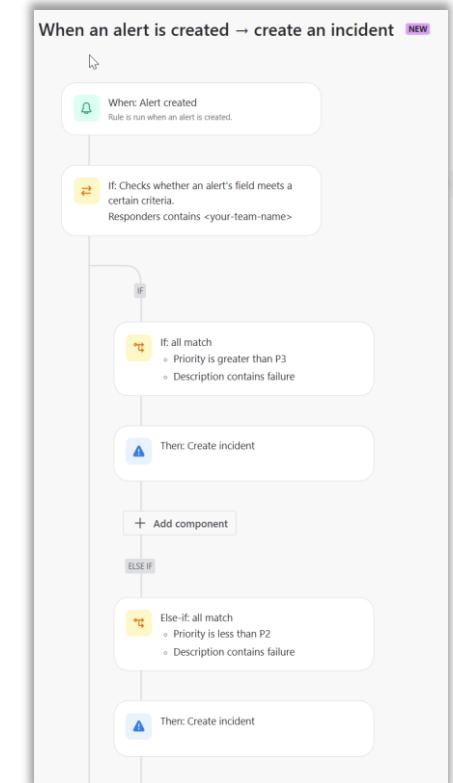
- Collect alerts from **multiple** monitoring sources (API, webhooks, 200+ integrations).
- Apply **deduplication**, correlation and filtering rules to reduce noise
- Define alert **policies** for routing, escalation, and maintenance handling
- Monitor **heartbeat checks** to detect silent failures
- Manage **maintenance** windows to suppress non-critical alerts



# Automation and Alert Flow

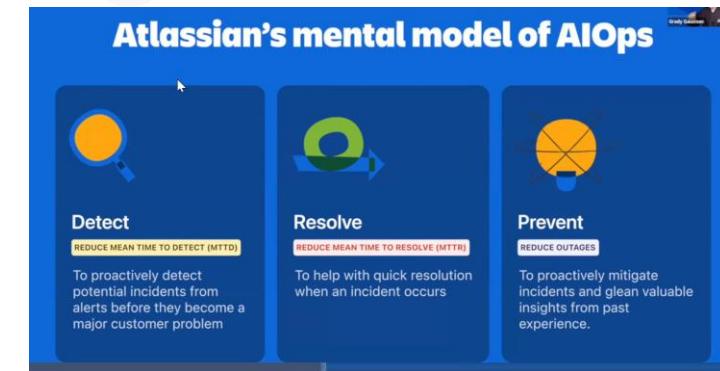
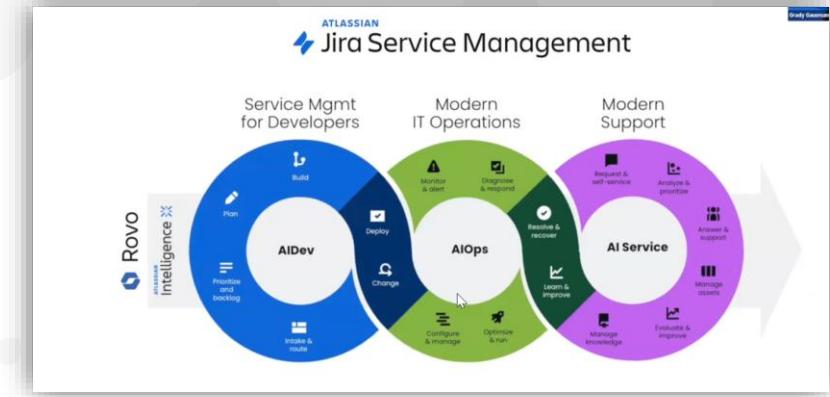
Detection → Filtering → Correlation → Incident creation → Assignment → Escalation → Closure → Review

- Collect alerts from **multiple** monitoring sources (API, webhooks, integrations)
- **Automate** event-to-incident transitions based on defined rules
- Use on-call schedules for automatic assignment and escalation
- Trigger workflows, SLAs, and notifications dynamically
- Capture resolution data for continuous improvement
- Reduce MTTR (Mean Time to Repair) and improve operational resilience



# AI for IT Operations (AIOps) in JSM - Introduction

- From automation to intelligence
- Atlassian AIOps: data-driven IT operations
- **Rovo**: the AI foundation behind JSM
  - Chat • Search • Agents • Dev
- Connecting people, knowledge, and actions



# AI for IT Operations (AIOps) in JSM - Capabilities

- Recognize patterns and **group related alerts**
- Suggest priority, tags, and ideal assignees
- **Auto-summarize** tickets and comments
- Recommend KB articles or runbooks
- Draft PIRs and suggest probable causes

The screenshot shows a JSM (Jira Service Management) interface for managing alerts. At the top, a red box highlights the 'Alert group: \*\* PROBLEM Service Alert /Diskspace (%) - volume D: is CRITICAL \*\*' header. Below this, the 'Alerts (14)' tab is selected in a red box. The main table lists 14 alerts, each with a checkbox, priority (P3), status (CLOSED), assignee (None), and responder count (28). The table is grouped by 'Semantic similarity, Tags' and the grouping stage is 'Grouping ended'. To the right, a sidebar titled 'What more can you do?' contains three suggestions with red boxes:

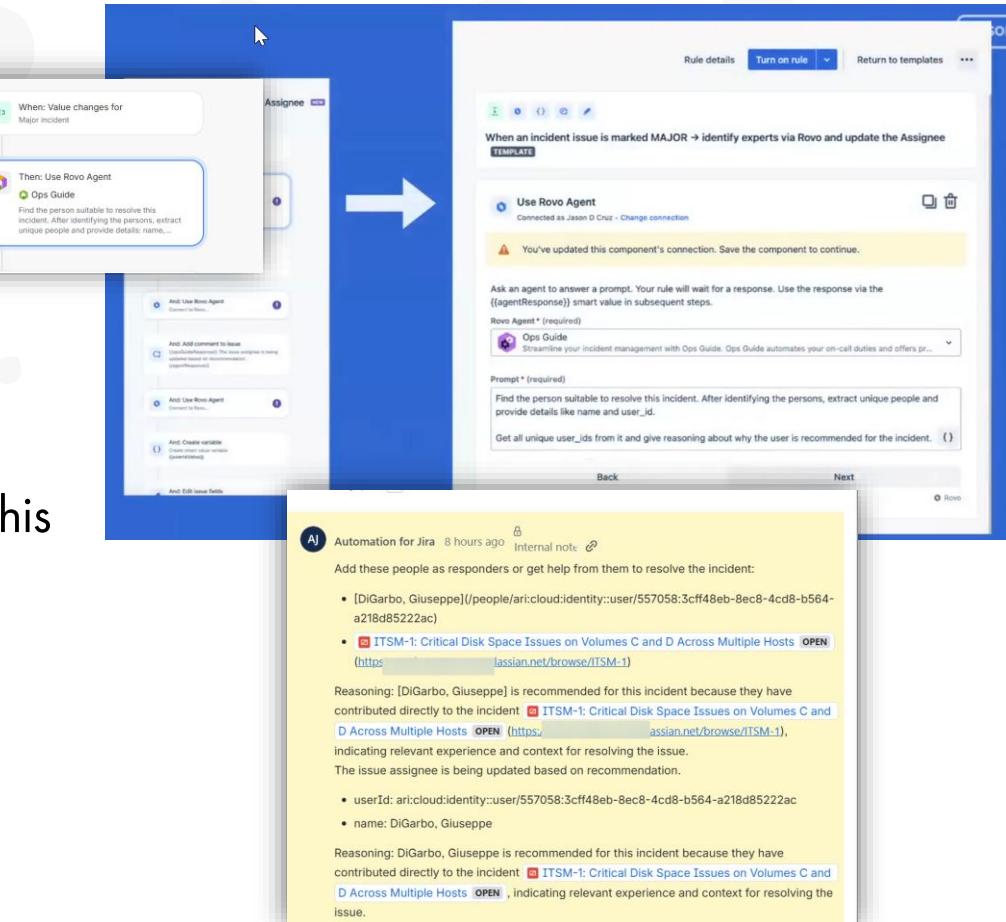
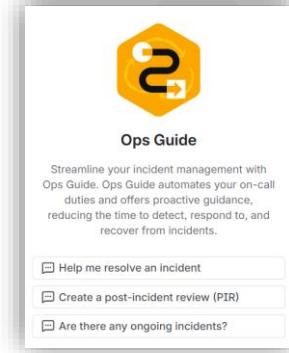
- Create incident**: Create an incident or a work item to resolve issues faster. You can add details of the incident or work item in the next step.
- ISSD**: Reach out to people who were responders to similar alerts and incidents in the past.
- Check the history of similar alert groups**: Shows a list of alert groups with their details, such as 'Alert group: \*\* PROBLEM Service Alert: /Diskspace (%) - volume D: is CRITICAL \*\*' and 'Alert group: \*\* PROBLEM Service Alert: /Diskspace (%) - volume C: is CRITICAL \*\*'.

## How alert grouping uses AI?

<https://www.atlassian.com/trust/atlassian-intelligence/transparency?tab=alert-grouping#>

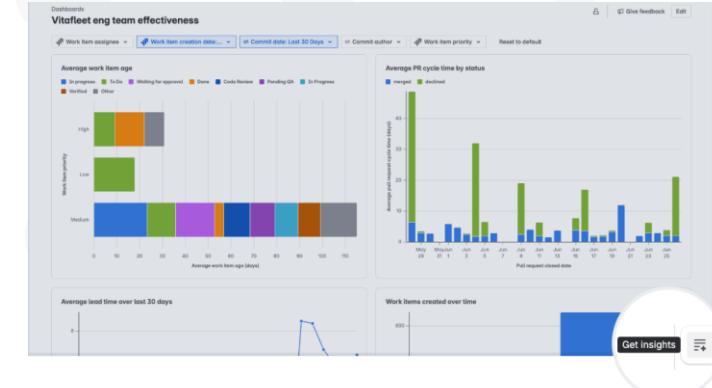
# AI in Practice: Rovo Ops Agent

- Run **natural language queries** to find alerts and incidents
- Access historical context and related knowledge articles
- Triage incidents quickly and suggest next steps
- Summarize incidents and create Post-Incident Reviews (PIRs)
- Update incident fields: priority, severity, major incident tag
- Example of JSM integration: “Find the person suitable to resolve this incident” automation



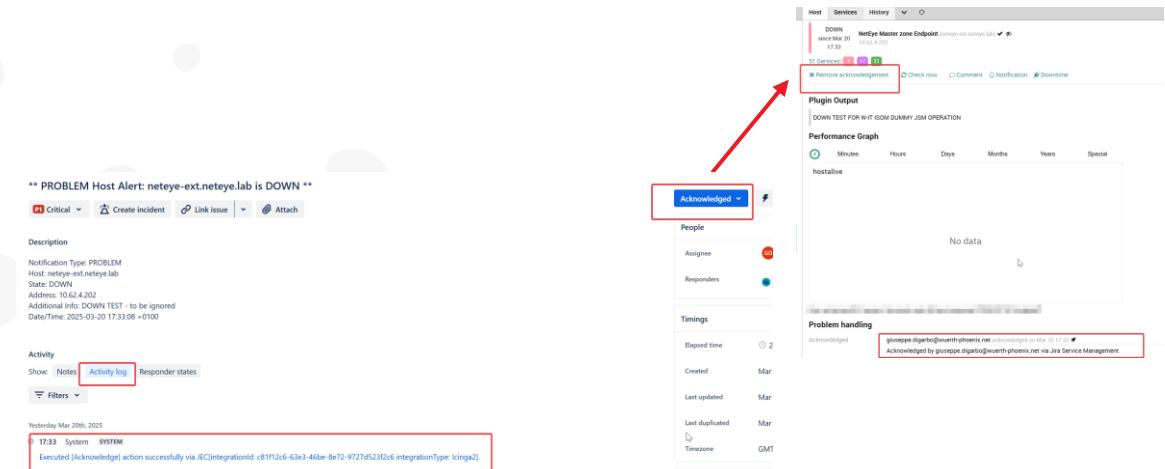
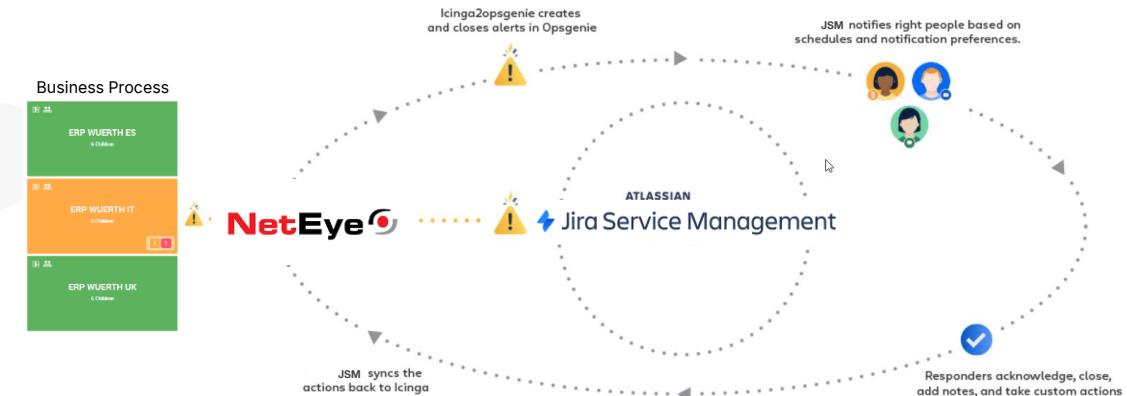
# AI-Driven Proactive Insights

- Detect emerging service patterns and anomalies before impact
- Surface change risks by correlating historical incidents and current changes
- Recommend preventive actions and remediation steps
- Integrate Rovo agents for predictive and proactive insights



# NetEye + JSM Example (with Rovo AI)

- NetEye detects anomaly → sends alert via nep-notification-jsm
- JSM deduplicates & groups alerts via Rovo-powered AI
- Incident created & assigned via on-call schedule + Rovo suggestion
- Bidirectional ack: Acknowledgement in JSM updates NetEye
- Rovo agent recommends runbook and actions
- SLA, escalation, resolution
- Post-resolution: Rovo drafts PIR and suggests tuning rules



# Key Takeaways and Next Steps

- ITIL4 gives the framework
- JSM Operations provides the platform
- Rovo AI accelerates triage, reduces noise, adds intelligence and automation
- NetEye closes the loop with monitoring integration

## Next Steps

- Analyze your alert patterns and noise sources
- Start a pilot with on-call schedules and alert policies
- Measure improvement in MTTR and team responsiveness
- Scale automation and AI adoption progressively

**NetEye** 

