Goku AIOS

White Paper

Enterprise AI Agent Operating System for the AI Era

June 2026

Preface

The history of information technology is, at its core, a history of steadily expanding human capacity to process information.

In the 1960s, humanity entered the Mainframe Era. Computing resources were extremely expensive; software's primary goal was digitising data and enabling centralised computation.

In the 1990s, as personal computers proliferated, humanity entered the PC Era. For the first time, computing power reached individuals; tools such as Word and Excel made the computer central to personal productivity.

After 2000, the Internet Era arrived in full force. Software connected the entire world through networks; companies such as Google, Amazon, Alibaba, and PayPal redefined business models, and software evolved from a 'tool' into a 'platform'.

After 2008, the Mobile Internet Era exploded. The smartphone became the new computing centre; mobile payments, social networks, and instant messaging collectively built an entirely new digital way of life.

Around 2013, deep learning and cloud computing began accelerating in tandem. Enterprises started accumulating vast data assets; machine learning gained wide adoption in recommendations, advertising, risk control, and search, while cloud infrastructure became the new technology foundation.

In 2022, large language model (LLM) technology, led by OpenAI, achieved a historic breakthrough, and humanity formally entered the AI Era — a technological transformation far deeper than the mobile internet revolution.

Every piece of software in the past was, at its core, 'human-operated software'. Humans typed commands, clicked buttons, queried data, read results, made decisions, and drove workflows — software was always in a passive-response state.

In the AI era, software is undergoing a fundamental transformation:

> **Humans command agents; agents operate software.**

This is a true paradigm shift. AI is no longer just a chatbot; it will gradually become a digital execution entity capable of understanding task objectives, calling tools, coordinating across systems, handling complex workflows, running continuously, and optimising autonomously.

In tomorrow's enterprises, some work will be done by humans, some by AI agents, and some through human-AI collaboration. Enterprise software architecture will also undergo a fundamental change as a result.

Traditional software was designed around data storage, page rendering, user interaction, and access control. Future agent systems, however, must confront entirely new challenges: long-lifecycle task execution, multi-step reasoning and tool invocation, multi-agent collaboration, dynamic workflow orchestration, agent permission governance, memory and knowledge management, approvals and compliance, high-concurrency scheduling, failure retry and state recovery, and autonomous system evolution.

These problems are not ones that traditional SaaS architecture can naturally solve.

Today, many AI systems remain stuck at the prompt-engineering and simple-automation stage. When they truly enter enterprise production environments, what enterprises actually need is:

Runtime Infrastructure for AI Agents

This infrastructure must not only manage the agent lifecycle, tool invocation, workflow organisation, and knowledge memory — it must also address enterprise-grade challenges:

organisation and roles, security and permissions, messaging and communication, audit and compliance, concurrency and SLA, observability, stability and cost control, and autonomous evolution.

This is the context in which Goku AIOS was born.

Goku AIOS is not a chatbot platform, nor merely an agent framework. Its goal is to become:

The Enterprise Operating System for the AI Era

In this system:

- Agents can be managed like employees
- Workflows can be scheduled like production lines
- Tools can be executed like system calls
- Knowledge can be accumulated like memory
- A Supervisor can oversee operations like a manager
- The runtime can continuously learn and self-evolve

The future software world will no longer be the simple combination of software + users, but will gradually evolve into:

Agents + Runtime System + Human-AI Collaboration

Goku AIOS is the builder of that future.

Chapter 1 — Why the AI Era Needs AIOS

1.1 From Copilot to Autonomous Agent

Over the past two years, the breakthrough of large language models has driven AI into software systems at scale. Initially, most AI applications took the form of chatbots or copilots. The core goal of a copilot is to 'assist humans' — answering questions, summarising content, generating code, offering advice — thereby improving human productivity.

At this stage, however, AI is fundamentally still in a human-led, AI-assisted mode. The true executor of tasks is still the human: humans initiate operations, drive workflows, control context, invoke systems, and make final decisions. AI is merely an efficiency tool.

Yet as the reasoning capability of large models, tool calling, memory management, workflow orchestration, and multi-agent collaboration continue to advance, AI is entering a new stage — the autonomous agent stage. At this stage, AI no longer merely answers questions; it begins truly executing tasks, calling tools, operating systems, coordinating workflows, running for extended periods, and completing work autonomously.

This means AI is evolving from an assistant into a digital execution entity — one of the most fundamental changes in the AI era.

1.2 Software Architecture Is Undergoing a Fundamental Transformation

Software systems of the past few decades were designed primarily around human-computer interaction: interfaces, databases, permission systems, forms and workflows, API interfaces — all presupposing that humans are the core operators of the system. Humans understand tasks, judge context, drive processes, coordinate cross-system operations, and make final decisions; software merely provides tools and interfaces.

In the age of AI agents, this assumption is beginning to break down. As stated in the Preface:

> **Humans command agents; agents operate software.**

Agents will gradually become the primary executors of software systems. The design goal of software systems is shifting from human-computer interaction to autonomous agent execution — a paradigm shift at the operating-system level.

1.3 New Challenges for Enterprise Agent Systems

Once AI agents begin truly participating in enterprise operations, system complexity will rise rapidly. Unlike traditional chatbots, enterprise-grade agent systems need to run continuously and process real business — which brings a series of entirely new system challenges.

1.3.1 Long-Running Tasks

Traditional requests are typically short-lived — HTTP requests, API calls, and page interactions usually end within seconds. But enterprise agent tasks may last minutes, hours, or even days: automatically processing email queues, tracking customer-service tickets, cross-departmental approvals, financial reconciliation, and data-analysis workflows.

This means agent systems must possess state management, task recovery, interrupt-and-resume, and long-term context management capabilities.

1.3.2 Multi-Step Workflows

Enterprise tasks are generally not single steps. A complete task may involve: fetching data, analysing a problem, querying the knowledge base, calling multiple systems, generating a solution, submitting for approval, executing the operation, updating status, and notifying users. This means agent systems need workflow orchestration capabilities.

1.3.3 Multi-Agent Collaboration

In future large-scale enterprise systems, no single agent can handle all tasks. Systems will gradually evolve into 'agent organisations' — customer-service agents, finance agents, technical-support agents, approval agents, and risk-control agents each with their own roles, permissions, and tools. These agents need unified scheduling, message communication, and state synchronisation; their architectural complexity already approaches that of distributed systems.

1.3.4 Tool Governance and Access Control

When agents start calling real tools — querying databases, sending emails, calling payment interfaces, modifying orders, executing code, calling enterprise APIs — system risk rises rapidly. The system must then address permission isolation, tool ACLs, audit logs, security governance, and approval workflows; otherwise agents cannot truly enter enterprise core production environments.

1.3.5 SLA and Reliability

The core requirement of enterprise systems is stable operation. Enterprises cannot tolerate agents randomly failing, workflows being lost, context disappearing, or long tasks being interrupted. Therefore, AI agent runtime systems must possess a complete reliability infrastructure:

- Retry and timeout control
- Queue systems and circuit breakers
- Failure recovery and observability
- Cost control

This means the complexity of agent runtime systems is approaching that of traditional operating systems and distributed runtimes.

1.3.6 Human-in-the-Loop and Approval Governance

In enterprise environments, AI cannot completely escape human governance. Many critical processes still require human confirmation, approval, risk control, and compliance checking. Future enterprise systems will increasingly adopt human-AI collaboration; agent runtime systems must natively support workflow approval, escalation handling, human takeover, and auditability.

1.4 Why Traditional SaaS Architecture Cannot Solve These Problems

Traditional SaaS systems solve the problem of 'how humans use software' — oriented towards UI, forms, and user operations. Agent runtime systems, however, need to solve the problem of 'how software executes work autonomously' — oriented towards tasks, execution, state, collaboration, and autonomy.

These are entirely different system models. Traditional SaaS architecture cannot natively support AI-native system capabilities: autonomous agents, long-lifecycle workflows, multi-agent collaboration, and self-evolving runtime systems.

1.5 The AI Era Needs New Infrastructure: AIOS

Therefore, the AI era needs a new system layer. This layer is not a web application framework, not a workflow tool, and not a chatbot platform. It is:

An AI Operating System (AIOS)

The core goal of AIOS is to provide unified runtime infrastructure for AI agents — responsible for agent lifecycle management, workflow execution, tool invocation, memory management, knowledge access, multi-agent collaboration, permission governance, reliability engineering, human-AI collaboration, and autonomous runtime evolution.

AIOS is not simply an AI application layer; it more closely resembles 'the runtime infrastructure of the AI era'.

Goku AIOS is the engineering realisation of this infrastructure.

Chapter 2 — From Sun Wukong to the Digital Avatar

2.1 An Ancient Metaphor About Avatars

One of Sun Wukong's most impressive abilities in Journey to the West is the 'body-beyond-body' technique. When outnumbered by monsters and demons, he would pluck a handful of monkey hairs from the back of his neck, blow on them lightly, and transform them into a group of little monkeys. These were not ordinary illusions — they inherited a portion of Sun Wukong's abilities, obeyed his commands, completed tasks independently, and collaborated in battle, helping the original break through limits of time, space, and number.

One Sun Wukong could not be everywhere at once. Through avatars, he could extend his capabilities outward.

This is the core meaning of 'digital avatar'. In the AI era, every individual, enterprise, and organisation will gradually come to possess their own digital avatars — and those avatars are AI agents.

2.2 What Is a Digital Avatar?

A digital avatar is not simply a chatbot, nor a fixed script or set of automation flows. A true digital avatar should possess the following characteristics: it understands its owner's goals, learns from its owner's knowledge and experience, masters tools in specific domains, follows organisational rules and permissions, can continuously execute tasks, can request human confirmation when necessary, and can optimise itself from historical executions.

In other words, a digital avatar does not 'speak for' its owner — it acts for them.

Traditional software primarily helped humans improve efficiency. Digital avatars, however, start helping humans extend their capability boundaries:

- They allow one person to handle more things simultaneously
- They allow a team to have more execution units
- They allow an enterprise to convert experience, process, and knowledge into a runnable agent system

2.3 From 'Tool' to 'Avatar'

Traditional software is a tool that humans use to complete work. People open their email, read messages, and decide how to reply; people log into the customer-service system, view tickets, and decide how to handle them; people open the BI system, query data, and produce analysis reports. In this model, software is passive.

Digital avatars are different. A digital avatar can automatically read emails, understand customer requests, query the knowledge base, retrieve historical cases, invoke internal systems, generate reply drafts, submit for human approval, and complete the send once approved.

This means software is no longer merely a tool, but begins to become an execution entity. Humans shift from 'operator' to 'commander'; the system shifts from 'passive tool' to 'active executor'. This is the fundamental difference between AI agents and traditional software.

2.4 Why Enterprises Need Digital Avatars

Much of the work in enterprises is not creative strategic decision-making, but repetitive knowledge labour — handling customer-service emails, responding to technical support requests, following up on sales leads, financial reconciliation, compliance checks, data

analysis, report generation, internal approvals, and daily operations monitoring.

This work shares four common characteristics: it relies on knowledge (understanding products, processes, historical cases, and organisational rules), relies on tools (logging into multiple systems, querying data, calling interfaces, updating status), relies on judgement (deciding next steps based on context), and relies on collaboration (completing work across departments, systems, and roles).

In the past, this work had to be done by humans, because traditional software could not truly understand tasks or coordinate processes autonomously. But the emergence of large-model and agent technology is making it possible to digitally reconstruct such work. Enterprises can now not only encode processes into systems, but package experience, knowledge, tools, and permissions into individual digital avatars.

2.5 'Monkey-Hair Avatars' in the Enterprise

Sun Wukong's monkey-hair avatars are, at their core, a capability-extension mechanism. The agents in Goku AIOS can likewise be understood as an enterprise's 'digital monkey hairs':

- A customer-service specialist has a customer-service agent
- A technical-support engineer has a technical-support agent
- A finance professional has a financial-analysis agent
- A manager has an operations-monitoring agent
- A sales team has a sales-follow-up agent

These agents do not exist wholly independently of humans; they are better described as extensions of human capability: humans define goals, agents execute processes, humans supervise critical junctures, agents accumulate experience, and the system optimises continuously.

This is not 'AI replacing humans'. More precisely, it is the digital extension of human capability.

2.6 Digital Avatars Are Not Fully Autonomous

Digital avatars do not mean humans exit the system. In enterprise environments, completely unsupervised agents are dangerous, because enterprise tasks often involve customer information, financial data, payment transactions, contract terms, legal risk, brand reputation, and security permissions.

A digital avatar that can truly enter enterprise production environments must therefore possess governance mechanisms. Being clever is not enough — it must also be controllable, auditable, authorisable, reversible, supervisable, and escalatable.

This is also an important distinction between Goku AIOS and ordinary agent tools. Goku AIOS does not simply let agents act autonomously — it provides a complete enterprise-grade operating environment for agents. In this environment, agents have identity, permissions, tool boundaries, execution logs, approval mechanisms, supervisors, and can be governed.

Only then can digital avatars truly enter enterprises.

2.7 From Personal Avatars to Organisational Avatars

The significance of digital avatars lies not only in helping individuals improve efficiency. More

importantly, they will change how organisations operate.

In the past, an organisation's capability came from its people: if an employee was experienced, the organisation depended on that employee; if a process was understood by only a few people, a knowledge bottleneck formed; if a role turned over, the organisation might lose that experience.

Digital avatars can convert individual experience, organisational processes, and business knowledge into continuously-running system capabilities — knowledge can be accumulated, processes can be reused, experience can be replicated, service capacity can be scaled, and organisational efficiency can be amplified.

Ultimately, enterprises may come to possess not just employees and software, but also a group of continuously-running agents. These agents constitute the enterprise's new digital workforce.

2.8 Why Goku?

The name Goku comes from 'Wukong'. In Chinese, 'Wukong' refers to Sun Wukong from Journey to the West; in Japanese and English contexts, Goku is also the common romanisation of 'Wukong'.

Choosing this name is both a tribute to the ancient metaphor of 'avatars' and an expression of the core idea of the AI agent era — that an individual or organisation can break through their own capability boundaries through intelligent avatars.

- For individuals: Goku is a digital assistant
- For teams: Goku is a collaborative agent
- For enterprises: Goku is a digital avatar operating system
- For the future: Goku is an engineering practice of agent-era runtime infrastructure

2.9 From Mythological Ability to System Capability

The avatars of mythology are the product of imagination. The avatars of the AI era are an engineering capability.

The real question is not whether we can generate an agent, but: How do we manage large numbers of agents? How do agents use tools? How do we control agent permissions? How do agents remember experience? How do agents recover from failure? How do agents collaborate with humans? How do agents continuously evolve? How can enterprises confidently use agents?

This is the problem Goku AIOS is built to solve. It aims to transform avatars from a mythological ability into an enterprise system capability — not to make every person become Sun Wukong, but to allow every organisation to possess its own digital avatar system.

Blow on the monkey hair, and the avatars emerge. That is where Goku begins.

Chapter 3 — Goku AIOS

3.1 From Agent Tool to Agent Operating System

With the development of large language model technology, a large number of AI agent tools have appeared on the market. These systems typically offer prompt engineering, chat interfaces, tool calling, and workflow automation, and have achieved rapid growth in personal efficiency, content generation, and simple automation.

However, when these tools enter enterprise production environments, their limitations quickly become apparent: they lack the ability to handle long-lifecycle tasks, multi-agent collaboration, complex approval flows, fine-grained permission governance, SLA guarantees, and self-evolution.

This is precisely the gap between an 'agent tool' and an 'agent operating system'.

3.2 What Is Goku AIOS?

Goku AIOS is an enterprise-grade AI agent operating system. Its core goal is to provide unified runtime infrastructure for AI agents — so that enterprises can deploy, manage, govern, and scale AI agents in a production environment at minimal cost.

Goku AIOS is self-hosted, meaning it can be deployed on private cloud or on-premises infrastructure, ensuring that enterprise data does not leave the premises. At the same time, it supports multi-tenant architecture, so multiple business units or teams can share the same deployment while maintaining complete data isolation.

3.3 AIOS: The New System Layer of the AI Era

Goku AIOS is positioned as a new system layer in enterprise IT architecture:

- Below it: cloud infrastructure, database systems, enterprise APIs, and external services
- Within it: agent runtime, workflow engine, tool registry, memory system, knowledge system, permission layer, audit log, and self-evolution mechanism
- Above it: business applications, enterprise systems, and external channels

This layer enables enterprises to quickly build intelligent agent capabilities on top of existing IT infrastructure — without needing to rebuild the underlying systems.

3.4 Core Capabilities of Goku AIOS

3.4.1 Agent Runtime

The core of Goku AIOS is the agent execution engine — the ReAct (Reason + Act) loop. Every agent, when receiving a task, cycles through thinking, tool calling, observing results, and updating judgement — repeating until the task is complete.

The runtime supports concurrent execution of multiple agents, task priority queues, heartbeat monitoring, zombie detection, failure recovery, and automatic re-queuing of tasks interrupted by server restarts. The Conversational Agent Builder (v1.9.27) further allows business users to create and publish their own specialised agents through natural-language dialogue, with no form-filling required.

3.4.2 Workflow Engine

Goku AIOS provides an AI-native workflow engine that supports multi-step task orchestration, state machine transitions, conditional branching, retry logic, human approval steps, and exception recovery. Workflows can be triggered by webhooks (with mandatory HMAC-SHA256 signature verification and replay-attack protection from v1.9.27), scheduled timers, or API calls.

The workflow designer provides a visual canvas for building, editing, and monitoring workflows without any coding.

3.4.3 Tool Calling

An agent's true capability lies not only in generating text, but in calling real tools — querying databases, sending emails, calling payment interfaces, accessing enterprise APIs, executing code, searching the knowledge base, and calling third-party systems.

Tool calling is the key that transforms an agent from a chatbot into an execution entity. Therefore, Goku AIOS provides complete tool governance: tool registration, tool permissions, tool ACLs, governance, and audit — enabling enterprises to safely let agents use real system capabilities. The platform currently includes 123+ built-in tools covering file operations, code execution, web search, database queries, email, instant messaging, image generation, calendar management, and more. Through an automated tool-reference document mechanism, every agent can look up the usage and parameter details of any tool via knowledge_search.

3.4.4 Memory System

A true agent cannot have only short-term context. It needs long-term memory, historical experience, user preferences, task state, and organisational knowledge. Therefore, Goku AIOS introduces a memory system to manage context memory, long-term memory, execution history, task state, and organisational knowledge.

Memory enables agents to move beyond 'instant chat' and begin to possess continuous-learning and long-term-collaboration capabilities.

3.4.5 Knowledge System

Much of an enterprise's capability comes from accumulated knowledge — SOPs, FAQs, technical documentation, ticket cases, historical experience, and internal policies. Therefore, Goku AIOS provides an enterprise knowledge system supporting RAG, semantic search, hybrid search (BM25 + Qdrant), knowledge version management, and organisational knowledge management. From v1.9.27, knowledge_search simultaneously covers both user-uploaded knowledge documents and the platform's built-in Document Centre (technical manuals, roadmaps, tool references, etc.) — eliminating agent knowledge blind spots.

Knowledge is no longer merely documentation; it will gradually become the organisational memory of agents.

3.4.6 Multi-Agent Collaboration

Future large-scale agent systems cannot rely on a single agent. Therefore, Goku AIOS supports multi-agent collaboration: different agents have different roles, use different tools, hold different permissions, and are responsible for different domains. The system can uniformly schedule agents, assign tasks, manage state, and coordinate communication.

Future enterprise AI systems will increasingly resemble digital organisations.

3.4.7 Approval and Governance

AI in an enterprise environment must possess governance capabilities. Therefore, Goku AIOS has a built-in approval workflow, human takeover, escalation handling, auditability, and

permission governance. The system allows human approval of critical steps, human takeover of tasks at any time, complete execution log recording, and agent permission boundary control.

Goku AIOS's goal is not to let AI act without restriction, but: to let AI operate safely within enterprise rules.

3.4.8 Enterprise Security

Once agents begin accessing real systems, security becomes extremely important. Therefore, Goku AIOS provides an RBAC permission system, SSO single sign-on, tenant isolation, secure credential management, audit logging, and permission governance. From v1.9.27, inbound security is further strengthened: workflow webhook trigger endpoints require HMAC-SHA256 signatures (dual-header validation with X-AIOS-Timestamp + X-AIOS-Signature) with a request-fingerprint deduplication mechanism (replay-attack protection), ensuring the integrity and unforgeability of external trigger chains.

Goku AIOS was designed from the outset not for personal demonstrations, but for enterprise production environments.

3.4.9 Audit and Compliance

Enterprise AI systems must be traceable, auditable, explainable, and retroactable. Therefore, Goku AIOS supports agent execution logs, workflow tracing, tool-call audit, decision history, and compliance auditing — enabling enterprises to clearly know: what the agent did, why it did it, which tools it called, which data it used, and who approved critical operations.

3.5 Managing Digital Avatars Like Employees

One of Goku AIOS's core principles is: agents are not features — they are digital employees.

In tomorrow's enterprises, agents will increasingly resemble organisational members — with roles, permissions, responsibilities, tools, collaborative work, supervision, and continuous learning. Enterprises therefore need not just AI tools, but a digital employee management system.

In Goku AIOS, every agent has: a name and identity, a department and division, a set of permitted tools, a knowledge base, an execution history, an approval chain, and a performance record. This is no longer 'configuring an AI model', but 'onboarding a digital employee'.

3.6 From Software System to Digital Organisation

Future enterprises may gradually evolve from having 'software + employees' to having 'software + employees + agent workforce'. This agent workforce has characteristics entirely different from traditional software:

- Agents can independently execute complex tasks
- Agents can learn from experience and improve over time
- Agents can collaborate and communicate with each other
- Agents can escalate to human supervision when needed
- Agents can operate 24/7 without interruption

Goku AIOS is the system infrastructure for building this digital organisation.

3.7 Sound Architectural Design

Goku AIOS adopts a layered, loosely-coupled architecture:

- API Gateway layer: handles all external requests, authentication, and rate limiting
- Agent execution layer: ReAct loop, tool registry, memory, and knowledge
- Workflow engine layer: DAG scheduling, state machine, approval, and retry
- Data layer: MySQL (tasks, conversations, logs), Qdrant (vectors), Redis (pub/sub)
- Infrastructure layer: supports Docker Compose, Kubernetes (Helm Chart), and cloud deployment

This design ensures that Goku AIOS can scale from a single-machine development environment to a production cluster, maintaining system consistency throughout.

Chapter 4 — Goku Router

4.1 The AI World Won't Have Just One Model

In the AI era, the model ecosystem is exploding. OpenAI's GPT series, Anthropic's Claude, Google's Gemini, Meta's LLaMA, Alibaba's Qwen, DeepSeek, Kimi, and many more models continue to emerge. Each model has its own strengths, costs, and applicable scenarios.

For an enterprise, relying on a single model provider means: binding to a single point of failure, inability to optimise costs, inability to choose the best model for each task, and inability to adapt to the rapidly evolving model landscape.

4.2 Entering the Multi-Model Era

The AI world is entering a multi-model era. Different models are suitable for different tasks:

- Complex reasoning and code generation: high-capability models (Claude Opus, GPT-4o)
- Rapid response and simple Q&A: fast lightweight models (GPT-4o-mini, Qwen-Turbo)
- Chinese-language business: domestic models (Qwen, Kimi, DeepSeek)
- Privacy-sensitive scenarios: local models (Ollama, LLaMA)

In this context, enterprises need a layer that can intelligently manage and route across models.

4.3 The AI World Needs a 'Routing Layer'

Just as the internet needs DNS to resolve domains and load balancers to distribute traffic, the AI world needs an intelligent routing layer to manage model calls. This layer should:

- Understand the capabilities and costs of each model
- Route each request to the most appropriate model based on task type and context
- Automatically fail over when a model is unavailable
- Optimise costs and response time
- Provide a unified access interface regardless of the underlying model

4.4 What Is Goku Router?

Goku Router is an intelligent model routing service, serving as the 'AI traffic control system' for Goku AIOS. It provides a unified model access layer, shielding the complexity of the underlying model layer from the application layer.

Goku Router supports: multi-model access, dynamic routing, cost optimisation, fault tolerance, model-capability registration, and unified API interfaces. Goku AIOS communicates uniformly with Goku Router rather than calling model provider APIs directly.

4.5 Why Model Routing Matters

Model routing solves three core enterprise problems:

1. **Cost optimisation:** Automatically route simple queries to low-cost models, saving 60–80% of model call costs

2. **Reliability improvement:** When the primary model is unavailable, automatically fail over to an alternative, ensuring business continuity
3. **Capability matching:** Route different types of tasks to models most suited to those tasks, improving task completion quality

4.6 Core Capabilities of Goku Router

4.6.1 Dynamic Model Selection

Goku Router can dynamically select the optimal model based on task type, context length, required capability, and cost budget. For example: route code-generation tasks to GPT-4o; route Chinese-language customer-service tasks to Qwen; route simple Q&A to GPT-4o-mini.

4.6.2 Cost Optimisation

The router tracks the token consumption and cost of each model call in real time, implementing cost optimisation strategies at the request level. Enterprises can set cost budgets, and the router automatically selects the most cost-effective model within budget.

4.6.3 Failover

When a model provider is temporarily unavailable or times out, the router automatically switches to a configured backup model — maintaining business continuity. The failover chain supports multiple levels: primary model → backup model → local model.

4.6.4 Multi-Model Collaboration

Some complex tasks need different models handling different stages. For example: use GPT-4o for reasoning, use a text-embedding model for retrieval, use an image-generation model for image production. Goku Router can coordinate multi-model collaborative pipelines.

4.6.5 Unified Model Access Layer

All models, regardless of provider, are accessed through a unified OpenAI-compatible API. Applications need not be adapted for different model SDKs; they access all models through a single interface.

4.7 AI Router: New Infrastructure for the Future AI World

Looking further forward, model routing will become as important as DNS in the AI world. Every AI application, every agent, every workflow will need to pass through an intelligent routing layer — dynamically selecting the best model, optimising costs, ensuring reliability, and managing capabilities.

Goku Router is an early practice of this idea. In the future, as the model ecosystem grows richer, intelligent model routing will become a fundamental capability of AI infrastructure.

4.8 From Model Calls to Intelligent Computing Power Scheduling

Model routing is not just 'selecting a model'; it is the intelligent scheduling of AI computing power. In the future, AI infrastructure will closely resemble cloud computing infrastructure: multiple computing-power suppliers (model providers), an intelligent scheduling layer (router), and consumers of computing power (agents and applications).

The router will be responsible for: load balancing, cost optimisation, SLA guarantees, capability matching, and fault tolerance — just as cloud load balancers and compute schedulers operate today.

4.9 Goku Router and Goku AIOS

Goku Router is the model gateway layer of Goku AIOS. Goku AIOS communicates uniformly with Goku Router; the router handles all model access details. This design enables Goku AIOS to be model-agnostic — the underlying model can be replaced or upgraded without any changes to the agent application layer.

The combination of Goku AIOS + Goku Router forms a complete enterprise AI operating system stack: Goku AIOS is responsible for the agent operating layer; Goku Router is responsible for the model infrastructure layer.

Chapter 5 — Supervisor and Autonomous Evolution

5.1 Traditional Software Does Not 'Reflect'

Traditional software is deterministic: given the same inputs, it always produces the same outputs. It does not learn from the past, does not analyse its own failures, and does not proactively optimise its own behaviour. Traditional software waits for humans to find problems and make modifications.

But in the AI era, intelligent agent systems need a higher level of capability: self-reflection, autonomous optimisation, and continuous evolution. This is the concept of 'self-evolving systems'.

5.2 Why Agent Systems Must Have Autonomous Evolution Capability

Agent systems face a unique challenge in enterprise environments: the business environment is constantly changing, task types are constantly expanding, user needs are constantly evolving, and system problems can only be fully discovered in production. If the system cannot proactively optimise based on actual operation, enterprises will need to constantly manually adjust agent configurations — a high-cost, non-scalable approach.

An ideal agent system should be like a skilled employee: learning continuously from each task, summarising what went wrong, proactively proposing improvements, and gradually becoming better at handling tasks. This is the core goal of Goku AIOS's self-evolution mechanism.

5.3 Supervisor: The Oversight Layer of AIOS

Goku AIOS has a built-in Supervisor service — an independent oversight layer that continuously monitors agent execution. After each task ends, the Supervisor asynchronously analyses the execution trace, identifying where the agent did well and where there were problems.

The Supervisor's role is analogous to a senior mentor reviewing a junior employee's work and providing specific improvement advice.

5.4 Core Capabilities of the Supervisor Mechanism

5.4.1 Execution Process Monitoring

The Supervisor traces every step of every task: which tools were called, what parameters were used, what results were returned, where errors occurred, and which steps took too long. This data forms the basis for subsequent analysis.

5.4.2 Failure Pattern Recognition

The Supervisor uses an LLM to analyse execution traces, identifying common failure patterns: tool calling errors, parameter assumption errors, context loss, and reasoning deviations. It then generates structured ImprovementProposals.

5.4.3 Risk-Graded Improvement Proposals

Each ImprovementProposal carries a risk rating: LOW (e.g., modifying prompt hints) → auto-applied; MED (e.g., adjusting tool configuration) → auto-applied after a delay; HIGH (e.g., modifying core workflow logic) → enters the human approval queue.

5.4.4 Automated Optimiser

The Optimizer service runs daily at a configured time, batch-processing pending improvement proposals. LOW and MED proposals are auto-applied; HIGH proposals await administrator approval. The entire process is fully traceable.

5.4.5 Self-Evolution Closed Loop

'Task execution → Supervisor analysis → Improvement proposal → Optimiser application → Better execution next time' — this closed loop gives the system the ability to continuously improve. In practice, this means: the more the system runs, the better it becomes.

5.5 Self-Evolution Is Not Unconstrained

The self-evolution mechanism in Goku AIOS is not unconstrained AI autonomy — it operates under a strictly risk-controlled governance framework. All evolution actions are logged and traceable; HIGH-risk actions require explicit human approval; all changes can be rolled back; the system will not autonomously modify its own security policies or permission configurations.

The goal is to make the system smarter, not to make it uncontrollable.

Chapter 6 — Enterprise-Grade Architecture

6.1 Multi-Tenant Architecture

Goku AIOS adopts a multi-tenant architecture, supporting multiple independent business units or teams within a single deployment. Each tenant has completely isolated data, agents, knowledge bases, permission configurations, and audit logs — full logical isolation is achieved without physical isolation.

The three-level organisation model (Tenant → Department → Team → User) allows enterprises to precisely control which agents which teams can access and use.

6.2 Permission System

Goku AIOS provides a complete enterprise-grade permission system:

- RBAC: fine-grained role-based access control
- Agent access policies: controlling which users/teams/departments can view and use specific agents
- Tool permission levels: public (0), auth-required (1), approval-required (2), admin-only (3)
- DLP data security: automatic PII detection, redaction, and blocking
- Approval chains: three risk levels, TTL control, and auto-escalation

6.3 Omnichannel Integration

Agents in Goku AIOS are not limited to web chat interfaces. They can be accessed through multiple enterprise channels:

- Web chat interface (PWA, mobile-responsive)
- Feishu / Lark bidirectional bots
- Microsoft Teams
- Email (inbound and outbound processing)
- WeChat Work
- Slack, Telegram, WhatsApp, Discord
- External API (OpenAI-compatible interface)

The UniCall unified channel gateway provides consistent message routing, identity resolution, and notification delivery for all channels.

6.4 Enterprise SSO Integration

Goku AIOS supports enterprise Single Sign-On through the OIDC and LDAP dual protocols. It integrates with mainstream identity providers including Keycloak, Okta, and domestic IdaaS platforms. The system supports automatic synchronisation of IdP groups to AIOS organisational structures, eliminating manual user management overhead.

6.5 Deployment Flexibility

- Docker Compose: suitable for development, evaluation, and small team deployment — one-command startup

- Helm Chart: K8s production deployment, supporting horizontal scaling and high availability
- Private cloud / on-premises: all data remains within the enterprise; no dependency on external cloud services
- Mixed deployment: cloud management, on-premises data

6.6 Observability and Operations

- Full execution tracing: every task step, every tool call, every LLM invocation recorded
- Audit logs: complete operation history, supporting compliance review and dispute resolution
- Health probes: daily automated smoke tests for agents and tools
- Log analysis: automatic anomaly detection and trend analysis
- Alert system: P0/P1 immediate notification through Teams / Email / push
- k6 load-test baselines: production SLA benchmarks documented

Chapter 7 — Current Status and Roadmap

7.1 Overall Status

Goku AIOS has completed multiple rapid iterations and has built an enterprise-grade capability stack covering the full agent runtime. It has entered a full enterprise production-operation phase.

7.2 Core Technology Stack

Backend: Python 3.12 + FastAPI + SQLAlchemy; Database: MySQL 8.0 + Qdrant (vectors) + Redis (pub/sub); Frontend: React + TypeScript + Ant Design; Deployment: Docker Compose + Helm Chart + K8s; LLM: OpenAI / Anthropic / Qwen / DeepSeek / local models.

7.3 Current Capability Overview

Goku AIOS has undergone multiple rapid iterations, building an enterprise-grade capability stack covering the full agent runtime. Core capability modules include: email automated processing and ticket management; knowledge base retrieval (BM25 + Qdrant hybrid vector search, now covering both knowledge base and Document Centre); tool calling framework (Tool Calling / MCP protocol, 123+ built-in tools); AI-native workflow engine; multi-model routing (Goku Router); self-evolving runtime (Supervisor + Optimizer); messaging platform integration (Feishu / Telegram / Slack / WhatsApp / Discord / Teams); UniCall unified channel gateway; enterprise SSO (OIDC / LDAP / Keycloak); three-level organisational permission system (Tenant → Department → Team); DLP data security; multi-tenant billing; stability engineering (19 composite DB indexes + k6 load-test baselines + Helm Chart / K8s deployment); test coverage system (4,685+ test cases); skill pack ecosystem (.aios-skill, 89+ built-in skills); developer SDK (aios CLI + register_tool + open API); mobile PWA + Push Notification; external memory sources (Notion / Obsidian); Outlook calendar integration; intelligent agent routing (per-message independent LLM routing); conversation PDF export; Conversational Agent Builder; and Workflow Webhook signature security.

7.4 Request Flow

Browser / Channel → Nginx → FastAPI (port 8106) → POST /conversations/{id}/messages (returns task_id) → GET /tasks/{task_id}/events (SSE stream) → AgentExecutor (ReAct loop, max 20 steps) → ToolRegistry.execute(tool_name, params) → EventBus.publish(event) → SSE to client.

A message creates a background task. The frontend listens via SSE. If the task completes before the client connects, the events endpoint immediately synthesises a completed event to avoid hanging.

7.5 Tool Calling and MCP Protocol

Goku AIOS has built Tool Registry, Tool Permission, Workflow Engine, and runtime collaboration as core modules. The system already supports Python execution, web search, SQL queries, enterprise API calls, data analysis, email system operations, and ticket system operations. The workflow engine supports multi-step tasks, state transitions, conditional branching, retry, human approval, and exception recovery — meaning Goku AIOS has begun to possess complex-task execution capability.

7.6 Multi-Model Routing

As the AI model ecosystem develops rapidly, Goku AIOS no longer depends on a single model and currently supports multi-model access including OpenAI, Claude, Gemini, DeepSeek, Qwen, and local models.

Through Goku Router, the system achieves dynamic model selection, cost optimisation, failover, token management, and unified model access. This means Goku AIOS has begun to form a model-agnostic architecture — enterprises will no longer need to bind to a single model provider.

7.7 Enterprise-Grade Permission System

Once agents enter enterprise production environments, permission governance becomes extremely important. Therefore, Goku AIOS has implemented multi-tenant isolation, RBAC, tool permissions, approval mechanisms, execution logs, and audit tracking — enterprise-grade governance capabilities. This means agent behaviour is controllable, traceable, auditable, and governable; the system was designed from the outset for enterprise production systems, not personal demo applications.

7.8 Enterprise Data Analysis Capability

Goku AIOS integrates AI BI data analysis capability. Business users can obtain data insights through natural-language conversation, without needing to understand SQL or BI tools. The system supports automatic chart generation, data trend analysis, and anomaly alerts.

7.9 Current-Phase Core Objectives

The current phase of Goku AIOS has fully transitioned from 'AI Demo' to 'enterprise-grade production operation'.

7.10 Milestones Achieved (v1.5 → v1.9.27+)

Key milestones completed across versions (selected):

- **v1.5 (MCP Ecosystem):** Complete MCP Server runtime (auth + quota + observability); built-in storage-s3 and file-parser MCP Servers; Goku-Router three-level failover integration
- **v1.6 (Messaging Platform):** Feishu / Telegram / Slack / WhatsApp / Discord / Teams full-channel integration; unified incoming-message routing layer; channel admin configuration UI; task heartbeat monitoring + zombie detection scheduler
- **v1.7 (Open API):** External API Key + /api/external/v1 access point + webhook callback; aios-sdk Python package (AIOSClient + AgentSession streaming calls); mobile PWA responsive layout
- **v1.9.7 (External Memory Sources):** Notion OAuth2 full-page synchronisation; Obsidian Vault incremental scanning; automatic knowledge-base ingestion; billing system × Goku-Router pull integration
- **v1.9.11 (Developer Extensions):** aios CLI, register_tool webhook tool registration, API Key QPS rate limiting, mobile Push Notification, and 89+ built-in Skill Packs
- **v1.9.12 (Enterprise SSO):** OIDC + LDAP dual protocol, IdP group → AIOS organisation auto-mapping, Admin UI, ZhuYun IDaaS deep integration, Keycloak full support
- **v1.9.13 (Platform Stability):** 133+ service-layer unit tests, 19 composite DB indexes (P99 improved 30–60%), k6 load-test baseline documentation, operations SOP triple pack (deploy / upgrade / rollback)

- **v1.9.27 (Conversational Agent Creation + Knowledge Search Enhancement):** Conversational Agent Builder — users describe requirements in natural language, the system automatically designs a system prompt, selects tool sets, and creates and publishes with one click; new agents are synchronously written to seed files to ensure consistent IDs across Dev / QA / Prod; knowledge_search unified coverage of knowledge base and Document Centre (BM25 dual-source search), eliminating agent knowledge blind spots; automated tool-reference documentation (123 built-in tools fully indexed, auto-updated on every startup); Workflow Webhook mandatory HMAC-SHA256 signing + timestamp replay-attack protection; Agent settings save response optimised (git operations async, response time reduced from seconds to milliseconds)

7.11 Autonomous Evolution (Self-Evolving Runtime)

Autonomous evolution was delivered in v1.9.9. After each task ends, the Supervisor service asynchronously analyses the execution trace and generates structured ImprovementProposals (risk levels: LOW / MED / HIGH). The Optimizer batch-processes pending proposals daily at 03:00 UTC: LOW / MED are auto-applied; HIGH enters the human approval flow. The system has 9 dedicated tools (read-only analysis + write), plus ClaudeCode tool support for automated code changes with pytest rollback verification.

The practical effect is: the system not only executes tasks, it also learns from every execution and proactively optimises itself. This is one of the most important distinctions between Goku AIOS and traditional workflow systems.

7.12 Mobile Agents

v1.9.11 implemented mobile Push Notification (PWA Web Push). The backend uses pywebpush to implement the complete VAPID chain; the frontend Service Worker handles push events and notification clicks; users can toggle notifications in the Profile page. The system proactively pushes when tasks complete or approvals are pending — no page refresh needed to sense agent state. Going forward, Goku AIOS will further support a mobile agent runtime, enabling users to schedule agents, approve workflows, manage tasks, and collaborate with their digital avatars at any time. Agents will truly begin accompanying human work.

7.13 Stability and Reliability Engineering

v1.9.13 completed the platform stability engineering milestone. One of the biggest differences between an AI demo and an enterprise system is stability. Goku AIOS strengthens the queue system, retry, failure recovery, observability, runtime monitoring, failover, and distributed collaboration. The practical effect is: Goku AIOS now has enterprise-grade SLA capability.

v1.9.27 further added automatic task re-queuing: on server restart, tasks with remaining retry budget are automatically reset to PENDING and re-executed by the executor, recovering approximately 80% of restart-induced task failures without user intervention.

7.14 Test Coverage System

Goku AIOS has established a complete test system: AI-native test infrastructure (4,685+ test cases, complete suite at 0 failures); P0 / P1 / P2 layered security regression tests; integration test coverage for the full agent execution chain; k6 performance test baselines; and E2E smoke tests.

7.15 SSO and Enterprise Integration

v1.9.12 formally delivered enterprise SSO. This is the enterprise access boundary: once SSO is in place, enterprises can access Goku AIOS using their existing identity system without creating a new account system.

Supported identity providers: Keycloak (open source, self-hosted), Okta (cloud, US), ZhuYun IDaaS (domestic), standard OIDC (Entra / PingFederate / Auth0 etc.), and LDAP (Active Directory).

7.16 Developer Extensions and Skill Pack Ecosystem

v1.9.11 formally launched the developer extension system. Through aios CLI (aios skill pack / aios tool register), external developers can register custom tools via webhook into AIOS — when the executor processes tasks, it automatically discovers and calls these external tools, on a complete par with built-in tools. Meanwhile, API Key QPS rate limiting (sliding-window per-key) ensures fairness and security boundaries in multi-tenant scenarios.

A Skill Pack (.aios-skill) is AIOS's distributable extension unit: a ZIP file containing manifest.json + tools/*.py, uploaded with aios skill publish in one click, and installed via make seed-agents or install_skill_pack.py. As of v1.9.27, the platform includes 123 tools and 89+ built-in Auto-Skill packages covering email processing, market intelligence, data extraction, content generation, local service search, technical research, and more — continuously expandable.

v1.9.27 adds the Conversational Agent Builder: users need not fill in any form — simply describe requirements in the chat window, and the system automatically recommends agent types, generates system prompts, and configures tool sets; one click creates and publishes. Created agents are synchronously written to the seeds/agents/ directory to ensure completely consistent unique identifiers across Dev / QA / Prod environments.

7.17 Intelligent Agent Routing

v1.9.17 completed the reconstruction of the conversation routing architecture. The original system used 'conversation-level sticky binding' — once the first message determined the agent, the entire session was locked with no ability to detect topic shifts; it also relied on cosine-similarity threshold matching, which had insufficient accuracy and could not understand business semantics.

The new three-tier routing model achieves per-message independent routing: (1) explicit user @mention takes priority (single-use, automatically cleared after sending); (2) toolbar session lock (effective for the entire session, can be cleared at any time); (3) Router Agent intelligent fallback (LLM_FAST_MODEL analyses the current message + last three rounds of context, selects the most appropriate agent, result does not persist conv.agent_id). This means that in a 30-round conversation, financial topics route to the finance agent, technical questions to the technical-support agent, and email drafting to the customer-service agent — each message routed appropriately.

7.18 Conversation Export and Knowledge Retention

Multi-round conversations often contain a large amount of valuable analysis, decision rationale, and execution records. v1.9.17 added conversation-to-PDF export: clicking the PDF button in the toolbar causes the system to compile the conversation content (including user messages, agent replies, tool-call results, and thinking processes) into a formatted PDF for storage, sharing, and archiving.

7.19 Messaging Platforms and Multi-Channel Integration

Goku AIOS has deeply integrated with mainstream enterprise messaging platforms, supporting Feishu / Lark, Microsoft Teams, WeChat Work, Email, PWA, Slack, Telegram, Discord, and WhatsApp. These channels all connect to AIOS through the UniCall unified channel gateway, achieving unified message routing, identity resolution, and intelligent agent dispatch.

7.20 UniCall Unified Channel Gateway

UniCall is the multi-channel unification layer of Goku AIOS. It acts as a gateway for all inbound and outbound messages, providing: channel identity resolution (mapping external user identities to AIOS user IDs), message format normalisation (converting each channel's proprietary format to a unified internal format), intelligent dispatch (routing to appropriate agents based on message content and context), and notification delivery (pushing agent replies back to the originating channel).

7.21 MCP Server Runtime

Goku AIOS supports the MCP (Model Context Protocol) protocol, enabling agents to call external tools and services through a standardised interface. The MCP Server runtime provides complete lifecycle management for MCP servers: startup, health monitoring, permission control, quota management, and audit logging. Built-in MCP Servers include storage-s3, file-parser, knowledge-graph, and more.

7.22 External Memory Sources and Calendar Integration

v1.9.7 delivered external memory source integration. Agents can automatically sync from Notion workspaces and Obsidian Vaults as external memory sources — incrementally scanning pages, extracting structured knowledge, and importing it into the AIOS knowledge base for retrieval.

Goku AIOS also integrates with Outlook Calendar, enabling agents to: read calendar events, create meeting invitations, manage scheduling, and trigger workflows based on calendar events.

7.23 Open API and Developer Ecosystem

Goku AIOS provides a complete developer API: /api/external/v1 supports standard HTTP calls and webhook callbacks; the aios-sdk supports skill pack publishing, installation, and tool registration (register_tool). API Keys use sliding-window rate limiting (per-key QPS), ensuring fairness and security boundaries in multi-tenant scenarios. Through the open API, AIOS is no longer just an internal platform — it begins to serve as an external interface for enterprise AI capabilities; external systems, mobile apps, and third-party services can all call AIOS agent capabilities through the standard API.

7.24 From Product to Infrastructure

At the current stage, Goku AIOS is still in a process of continuous evolution. But its goal is already no longer merely an AI product — it is:

The Runtime Infrastructure of the AI Era

In the future, Goku AIOS aims to become an important component of the agent runtime, AI workflow system, digital workforce platform, and AI-native enterprise infrastructure. The current version has formed a complete capability loop — from agent execution engine, multi-

channel access, enterprise governance to autonomous evolution — and represents the critical phase of Goku AIOS's transition from 'product' to 'enterprise-grade AI infrastructure'.

Conclusion

The AI era is not merely a technological upgrade — it is a paradigm shift in how enterprises operate.

Traditional software solves the problem of 'human-computer interaction'. AIOS solves the problem of 'agent-system operation'. This is a fundamental change in the system model: the software layer is shifting from serving human users to serving AI agents.

Goku AIOS is an engineering practice of this shift: providing enterprises with complete agent runtime infrastructure, enabling them to deploy, manage, govern, and scale AI agents in production environments — truly moving from 'being able to demo' to 'being able to go live'.

In the future, enterprises will not just have employees and software — they will also have continuously-running agent workforces. Goku AIOS is the operating system for building this new type of digital organisation.

> **Blow on the monkey hair, and the avatars emerge. This is where Goku begins, and where the enterprise AI era begins.**

Case Studies

Case 1 — Voice Quality Inspection Workflow

FROM BATCH CALL DATA TO AUTOMATED RISK REPORTS

Challenge: a financial services team needed to review thousands of customer-service call recordings daily for compliance risk, previously requiring a team of 8 QA specialists working full-time.

Solution: a Goku AIOS workflow automatically transcribes calls, passes them through an LLM for risk scoring, flags HIGH-risk conversations for human review, and compiles a daily risk report distributed to the compliance team via Teams.

- Processing time: from 2 days to 4 hours
- Human review focus shifted entirely to high-risk flagged cases
- Compliance coverage increased from 15% sampling to 100% full review

Case 2 — Legal Contract End-to-End Agent

TWO-LEVEL AI COLLABORATION FROM SALES INITIAL REVIEW TO LEGAL DEEP REVIEW

Challenge: contract review involved a sales team initial screening and legal team deep review — serial handoffs created significant delay, and junior reviewers frequently missed key clauses.

Solution: a two-agent pipeline (sales review agent → legal review agent) automatically extracts key clauses, flags risk points, generates review comments, and routes to the human legal team only when risk score exceeds threshold.

- Average contract review time: from 3 days to 6 hours
- Legal team workload reduced by 60%; focus shifted to genuinely complex cases
- Clause-miss rate reduced by 85%

Case 3 — Personalised Invoice Agent

FROM MANUAL INVOICE PRODUCTION TO AUTO-GENERATION WITH HUMAN REVIEW

Challenge: monthly invoice production required financial staff to manually compile data from 6 systems, with high error rate and 3-day turnaround.

Solution: the invoice agent automatically reads data from ERP, CRM, and billing systems, generates invoice drafts, and submits them to the finance team for approval via a human-in-the-loop flow.

- Invoice production time: from 3 days to 4 hours
- Error rate reduced by 92%
- Finance team freed from data-gathering work

Case 4 — Bank Financial Automation

FROM MANUAL RECONCILIATION TO CONFIGURATION-DRIVEN BATCH MERGING, LABELLING, AND BALANCE VERIFICATION

Challenge: a bank's monthly reconciliation covered 20+ account systems with complex rules — manual reconciliation took a team of 5 a full week.

Solution: a Goku AIOS financial automation agent reads multi-source GL data, performs automatic matching, labels exceptions, generates variance explanations, and produces a

reconciliation report.

- Reconciliation cycle: from 1 week to 8 hours
- Exception detection rate improved from 78% to 99%
- Completely replaced manual data entry

Case 5 — Enterprise Process RPA Agent

FROM MANUAL CROSS-SYSTEM ENTRY TO AI-DRIVEN ANOMALY-SENSING AUTOMATION

Challenge: a procurement process required data entry across 4 systems, with error rate of about 12% when done manually.

Solution: a Goku AIOS RPA agent handles cross-system data flow via browser automation + API calls, automatically detects anomalies, and escalates to human processing only for exception cases.

- Processing efficiency improved by 8x
- Error rate reduced from 12% to 0.3%
- Staff redeployed to higher-value work

Case 6 — Market Intelligence Agent

LETTING MANAGEMENT AND SALES TEAMS SEE MARKET TRENDS EVERY DAY

Challenge: market intelligence collection required multiple analysts to manually monitor news, competitor dynamics, and industry reports daily — information was scattered and timeliness was poor.

Solution: a market intelligence agent automatically collects news, analyses competitor dynamics, extracts business opportunities, and generates a daily market briefing delivered to management and sales teams via Teams at 09:00 every morning.

- Intelligence collection time: from half a day to fully automated
- Coverage expanded from 5 sources to 50+ data sources
- Sales team opportunity response time reduced by 60%

Case 7 — AI BI Data Analysis Agent

LETTING BUSINESS USERS GET DATA INSIGHTS THROUGH CHAT

Challenge: business users needed to rely on data analysts for BI reports — the average response time was 2 days, and the backlog of ad-hoc analysis requests was always long.

Solution: an AI BI agent directly understands business questions in natural language, generates SQL, queries the data warehouse, produces charts, and returns structured insights.

- Ad-hoc analysis response time: from 2 days to 5 minutes
- Data analyst team freed to focus on complex modelling work
- Self-service data access rate increased from 8% to 65%

Case 8 — Cross-System Operations Report Agent

FROM MANUAL DATA PULLING TO MULTI-PLATFORM AUTOMATED REPORT GENERATION

Challenge: weekly operations reports required data from 8 systems to be manually collected and compiled — taking 6 hours and frequently having version inconsistencies.

Solution: a report agent automatically pulls data from CRM, ERP, BI, and monitoring

platforms, generates standardised weekly reports, and distributes them to relevant stakeholders.

- Report generation time: from 6 hours to 30 minutes
- Data consistency issues eliminated
- Report distribution scope expanded from 5 recipients to 50+

Case 9 — TFN Compliance Review Expert

FROM MANUAL LINE-BY-LINE CHECKING TO 5-STEP STANDARDISED AUTOMATED COMPLIANCE DETERMINATION

Challenge: TFN (Tax File Number) compliance review required legal experts to manually check 30+ items for each application, with high error risk under time pressure.

Solution: a Goku AIOS compliance review agent standardises the review process into 5 steps, automatically checks all items, generates review opinions, and flags items needing human determination.

- Review time per application: from 45 minutes to 8 minutes
- Human error rate reduced by 94%
- Compliance team capacity expanded by 5x

Case 10 — Brand Localisation Agent

FROM MANUAL TRANSLATION TO SCALABLE MULTI-MARKET CULTURAL ADAPTATION

Challenge: brand localisation for the Japanese and US markets required marketing content to be adapted by native-speaker editors — slow turnaround, high cost, unable to keep up with campaign pace.

Solution: a Goku AIOS localisation agent combines translation, cultural context analysis, and brand tone calibration; generates localised copy drafts; routes to native editors only for final approval.

- Localisation cycle: from 5 days to 8 hours
- Translation cost reduced by 70%
- Localisation quality consistency significantly improved