TECHNICAL DATASHEET

AITHANEX AI

Platform to create, orchestrate and run AI agents

Type:	Technical datasheet
Product:	AITHANEX AI
System:	AITHANEX
Audience:	Engineering, installation, integration and support
Document version:	1.0
Review date:	2026-10-04
Fecha de creación:	04/10/2026

Product type

Platform to create, orchestrate and run AI agents. It includes routing between LLM models (large language models), RAG (retrieval of authorized knowledge to answer with context) and MCP (protocol to connect external tools and actions).

It provides assistance, analysis and supervised automation on the data, documents and processes of the AITHANEX system.

Role within the system

AITHANEX AI makes it possible to define, deploy and run specialized AI agents.

These agents operate on the data, documents, permissions and events of AITHANEX.

Agents act as intelligent intermediaries:

- They access knowledge bases through RAG: documentation, history, conversations and telemetry.
- They route tasks between LLM models according to cost, privacy, latency or complexity rules.
- They integrate with external tools through MCP, APIs, webhooks and corporate systems.
- They are used in AITHANEX Things to query operational data, assist the operator and run controlled actions.
- They are used in AITHANEX Customer service to bring intelligence to support bots, classification and assisted replies.

Main components

- **Agent designer.** Creates agents with role, instructions, permissions, sources and associated model.
- **Flow designer.** Defines visual flows that can use agents, rules, events, APIs or external actions.
- **Execution engine.** Runs agents in real time. Manages concurrency, retries, errors and timeouts.
- **AI orchestrator.** Coordinates RAG, MCP, model selection, permissions, execution and server-side auditing.
- **RAG server.** Manages knowledge retrieval, indexes, authorized sources and context for agents.
- **MCP server.** Exposes tools, connectors and external actions under permissions and traceability.
- **Model router.** Selects the model according to cost, privacy, latency, availability and complexity rules.
- **RAG / MCP / LLM console.** Administration interface to configure sources, tools, models, quotas and tests.
- **Usage reports.** Shows token consumption, conversations, executions, estimated costs and usage per model.

RAG, MCP and LLM architecture

The architecture is organized around an administration console and an AI orchestrator.

From the console, knowledge sources, tools, models, quotas and policies are configured. The orchestrator applies that configuration on each execution and coordinates context retrieval, tool use, model selection, auditing and usage reports.

Data

- **Agents.** Creation, management and versioning of custom agents per project or use case.
- **LLM models.** Support for Claude, GPT, Llama and other models (extensible by configuration).
- **Routing.** Rules defined per customer: latency, cost, accuracy and privacy policy.
- **RAG context.** Documentation, AITHANEX history, real-time data and external databases.
- **MCP integrations.** APIs, webhooks, external systems and analysis tools.
- **Workflows.** Visual flows that coordinate agents, events, rules, APIs and external actions.
- **Permissions.** Granular control: who creates agents, who runs actions and on which resources.
- **Observability.** Tokens, conversations, executions, errors, estimated cost and usage per model.
- **Configurable limits.** Timeouts, retries, cost limits and execution quota per agent or project.

Main use cases

1. Intelligence for AITHANEX Customer service

Verifiable objective: Bring intelligence to support bots, classification and assisted replies.

Technical:

- Input: conversation, channel, contact, history and support context.
- RAG: queries documentation, knowledge base and authorized replies.
- Model router: selects the model according to complexity, cost and project policy.
- Handoff: transfers the conversation to a human operator when the flow requires it.

Operational control:

- Escalation: hands off to a human operator if confidence is low or if requested.
- Audit: records query, sources, model, tokens and result.

Technical example:

Step	Example
Input	Customer query from a connected channel.
RAG context	Product documentation, support policies and conversation history.
Routing	Selection of the appropriate model for reply or classification.
Bot output	Suggested reply, label, priority or request for data.
Handoff	Transfer to a human agent with history and context.

2. Assisted control in AITHANEX Things

Verifiable objective: Query operational data and run controlled actions on devices when the deployment enables it.

Technical:

- Input: telemetry, state, alarm, asset, user and permissions.
- Query: the agent interprets data available in AITHANEX Things.
- Decision: the flow validates rules, permissions and action limits.
- Execution: the action is launched as a command, configuration, incident or external call.

- Logging: sources, model, tokens, user, action and result are stored.

Technical example:

Step	Example
Query	"Which chargers have active alarms?"
Context	States, alarms, location and latest events from Things.
Suggestion	Prioritize equipment with a critical alarm or loss of communication.
Controlled action	Create incident, notify technician or run authorized command.
Audit	Record of user, agent, model, sources, tokens and result.

3. Automation with workflows and MCP

Verifiable objective: Coordinate processes that combine rules, agents, APIs and external systems.

Technical:

- Flow: an event, rule or request starts the workflow.
- Agent: interprets context if the step requires it.
- MCP: runs authorized calls to external tools.
- Control: validates permissions, limits, errors and result.
- Observability: records execution, model, tokens, estimated cost and traceability.

Technical example:

Step	Example
Event	Critical alarm at an installation.
Evaluation	Queries history and escalation rules.
Integration	Creates a ticket, notifies the team and updates an external system.
Control	Verifies permissions and records each action.
Result	Traceable incident with technical context and sources.

Technical link

- AITHANEX Things**.** Provides telemetry, alarms and states as context for agents; receives controlled actions on devices.
- AITHANEX Customer service**.** Uses agents for bots, classification, assisted replies and handoff to human operators.
- AITHA Core**.** Publishes data and events that serve as operational context for agents and workflows.
- AURA X36**.** Provides operational events, security and industrial integration for analysis and automation.
- **External tools.** APIs, webhooks and connectors managed through MCP with permissions and traceability.
- AITHANEX **base.** Common permissions, history, auditing, users and traceability of the system.

References

- AITHANEX: https://aitha.es/en/platform/
- AITHANEX AI: https://aitha.es/en/platform/aitha-ai/
- AITHANEX Things: https://aitha.es/en/platform/things/technical/
- AITHANEX Customer service: https://aitha.es/en/platform/customer-service/technical/
- AITHA Core: https://aitha.es/en/aitha-core/technical/
- AURA X36: https://aitha.es/en/products/aura-x36/technical/
- AITHANEX: https://aitha.es/en/platform/technical/