

USER MANUAL

KlastroHeron Decision Intelligence

4Councilmen · Prudentia

Multi-Agent Convergence Decision Support — MCP Server Edition (Amazon Bedrock)

Version 3.0 · Container product for AWS Marketplace

Klastrovanie Co., Ltd.

Recommendations for human oversight, not automated decisions. PhD Dissertation, 2011. Prototype, 2026.

Contents

1. What This Product Is
2. How It Works — Convergence
3. Architecture & Delivery
4. Deployment on AWS
5. Language Models (Amazon Bedrock)
6. Using the MCP Tools
7. Document & Image Input
8. Answer Languages
9. Live Web Search (optional)
10. Licensing & Entitlement
11. Health & Monitoring
12. Responsible Use & Compliance
13. Support

1. What This Product Is

KlastroHeron Decision Intelligence (4Councilmen / Prudentia) is a **multi-agent decision-support** tool. It runs **four structurally independent AI agents** — each locked to a distinct perspective (surveillance, ethics, audit, transparency) — that analyze the same question **without negotiation**, over several rounds. A separate judge measures whether the four perspectives independently reach the same conclusion. When four minds designed to disagree nonetheless converge, that agreement is a strong, high-confidence signal (“the phone rings”).

It is delivered as a **stateless streamable-HTTP MCP server** for AWS Marketplace’s “AI Agents and Tools” product type. All inference runs on **Amazon Bedrock**. There are no API keys to manage — authentication is through the deployment’s AWS execution role (IAM).

⚠ HUMAN OVERSIGHT, NOT AUTOMATION

Every output is a **recommendation for a human reviewer**, never an automated decision or approval. The product surfaces the rare moment of agreement between independent perspectives; the human decides what to do with it. Do not use it as the sole basis for any decision affecting an individual.

2. How It Works — Convergence

The four agents do not move toward each other. Each stays fixed at its own extreme perspective; what moves is a shared *convergence point*, driven by how closely their conclusions and reasoning align. When independent reasoning collapses onto the same attractor, a **singularity** is detected — the run reports it as high confidence.

Model diversity is the core of the method. The same agent persona (a fixed system prompt) is embodied by a *different underlying Bedrock model each round*, rotating across the configured model list. Because the identity is only a prompt and the model rotates, any convergence cannot be attributed to one model’s bias. A high convergence signal means four orthogonal viewpoints, on rotating models, independently landed on the same conclusion.

Result state	Meaning
singularity (phone rang)	All four independently converged — strong candidate for human review.
dominant_compatible_proposal	A dominant proposal most agents are compatible with, short of full singularity.
partial_convergence	A coalition agrees; one or more dissent.
no_singularity	No convergence — the perspectives did not align.

3. Architecture & Delivery

The product ships as a single OCI container image distributed through an AWS Marketplace-managed Amazon ECR repository. It exposes a stateless MCP endpoint at `/mcp` on port **8000**, plus a small set of HTTP routes for document upload and health. There is no web UI.

Component	Role
Four orthogonal agents	Fixed perspectives (surveillance / ethics / audit / transparency), each a system prompt.
Semantic judge	A single fixed Bedrock model that scores convergence each round (stable across rounds).
Torus convergence engine	Projects the four positions and detects a singularity (2011 dissertation mathematics).
Embedding model	Sentence-Transformer (local, in-image) for drift detection — no external call.
Document parser	Extracts text/tables/OCR from uploaded files for grounding.
Amazon Bedrock (Converse API)	All agent + judge inference; model-family agnostic; IAM-authenticated.

4. Deployment on AWS

Where it runs

The buyer deploys and runs the container in their own environment: **Amazon ECS** or **Amazon EKS**. The container is stateless and writes nothing durable.

NETWORK REQUIREMENT (OUTBOUND TO AWS)

Inference and licensing are AWS cloud APIs, so the container needs **outbound network access to AWS** and valid AWS credentials via its execution role. Allow egress to `bedrock-runtime.<region>.amazonaws.com` and `license-manager.us-east-1.amazonaws.com`. A fully air-gapped (no-internet) deployment is not supported by this edition.

Required IAM permissions

Grant the task/pod execution role permission to run Bedrock inference and to maintain the AWS Marketplace license entitlement:

```
{
  "Version": "2012-10-17",
  "Statement": [
    { "Effect": "Allow",
      "Action": ["bedrock:InvokeModel", "bedrock:Converse"],
      "Resource": "*" },
    { "Effect": "Allow",
      "Action": [
        "license-manager:CheckoutLicense",
        "license-manager:ExtendLicenseConsumption",
        "license-manager:CheckInLicense",
        "license-manager:ListReceivedLicenses"
      ],
      "Resource": "*" }
  ]
}
```

On first use of AWS License Manager in an account, a one-time service-linked role may be required: `aws iam create-service-linked-role --aws-service-name license-manager.amazonaws.com`. Also enable access to your chosen models in the Bedrock console ("Model access"). No API keys or `AWS_PROFILE` are configured — the execution role is used automatically.

Configuration (environment variables)

Variable	Default	Description
<code>BEDROCK_MODELS</code>	(set by seller)	Comma-separated Bedrock model IDs the four agents rotate across. Model diversity is the core of the method — use two or more, ideally different families.

<code>BEDROCK_JUDGE_MODEL</code>	first of <code>BEDROCK_MODELS</code>	The fixed model used by the convergence judge.
<code>BEDROCK_REGION</code>	us-east-1	Region for Bedrock inference.
<code>BEDROCK_TEMPERATURE</code>	0.7	Sampling temperature for agents.
<code>CM_VISION</code>	on	When on, uploaded images are also sent to the model to be seen directly (requires a vision-capable model). Set to off for OCR-only.
<code>CM_WEB_SEARCH_MODE</code>	off	off, or agentcore to use a buyer-configured Amazon Bedrock AgentCore Browser Tool (see Section 9).
<code>FOURCM_MAX_FILE_BYTES</code>	26214400 (25 MB)	Maximum size per uploaded file.
<code>FOURCM_MAX_TOTAL_BYTES</code>	104857600 (100 MB)	Maximum total upload size per session.
<code>FOURCM_MAX_FILES</code>	10	Maximum files per upload session.
<code>FOURCM_UPLOAD_TTL_SECONDS</code>	3600 (1 hour)	Uploaded files are auto-deleted after this.

5. Language Models (Amazon Bedrock)

The models used are configuration, not code: set `BEDROCK_MODELS` to a comma-separated list of Bedrock model IDs. As new models become available on Bedrock, add their IDs — no code change. All are called through the uniform Bedrock **Converse API**, so any Converse-capable, system-prompt-supporting model works.

```
BEDROCK_MODELS="us.amazon.nova-lite-v1:0,us.anthropic.claude-3-5-haiku-20241022-v1:0,us.meta.llama3-1-8b-instruct-v1:0"
```

Exact model IDs vary by account and region — confirm them in the Bedrock console (Model catalog / Cross-region inference). A model must have “Model access” granted, or calls fail with `AccessDenied`.

Using your own fine-tuned or imported Bedrock models

The four agents can run on your own models, not only Amazon’s foundation models. Because model selection is configuration, you simply put the model’s identifier or ARN into `BEDROCK_MODELS` (and optionally `BEDROCK_JUDGE_MODEL`). This works with:

Your model type	What to set in <code>BEDROCK_MODELS</code>
Bedrock fine-tuned / customized model	The custom model’s Provisioned Throughput ARN.
Custom Model Import (your own weights)	The imported model’s ARN.

Cross-region inference profile	The inference profile ID or ARN.
--------------------------------	----------------------------------

The container never contains or downloads model weights. It only calls the identifier or ARN you provide, through the Converse API, using the deployment’s IAM role. You can mix your own model with foundation models in the same rotation, or run all four agents on your own model.

BUYER SETUP AND COST

Fine-tuned and customized models require **Provisioned Throughput**, and Custom Model Import uses **Custom Model Units**. Provisioning, capacity, and the associated Amazon Bedrock charges are configured in, and billed to, your own AWS account. This product references the model; it does not provision it. The model must be Converse-capable and support a system prompt.

Vision-capable vs text-only models

Image understanding (Section 7) requires a **multimodal** model. If an image is uploaded and a text-only model is in the rotation, that agent’s call for that image will error — choosing appropriate models is the deployment’s responsibility.

Capability	Examples (verify current availability in the Bedrock console)
Vision (sees images)	Amazon Nova Lite, Amazon Nova Pro, Anthropic Claude 3 / 3.5 (Sonnet, Haiku), Meta Llama 3.2 Vision, and other multimodal models.
Text-only (OCR only)	Amazon Nova Micro, and other text-only models.

If you enable image input, configure `BEDROCK_MODELS` with vision-capable models only, or set `CM_VISION=off` to use OCR text extraction instead.

6. Using the MCP Tools

Connect any MCP client to `http(s)://<host>:8000/mcp` (stateless streamable-HTTP). The server exposes four tools.

Tool	Purpose
<code>run_council</code>	Run the four councilmen over N rounds and return the convergence result.
<code>list_agent_sets</code>	List available agent scenario sets.
<code>get_agent_set</code>	Load one set’s four agent definitions.
<code>validate_agents</code>	Orthogonality check for a custom agent prompt.

run_council — parameters

Parameter	Description
-----------	-------------

<code>query</code>	The question / scenario to analyze (required).
<code>rounds</code>	Deliberation rounds, 1–5 (default 3).
<code>agent_set</code>	Scenario set (default “government”).
<code>risk_level</code>	“normal” or “high”.
<code>lang</code>	“auto” (answer in the question’s language) or a specific language — see Section 8.
<code>upload_session_id</code>	Ground the agents on uploaded documents/images — see Section 7.
<code>agents</code>	Per-call override of the four perspectives (ephemeral; not persisted).
<code>no_context</code>	Blind mode — agents do not see other agents’ prior-round findings.

run_council — result (key fields)

```
{
  "phone_rang": true,
  "convergence_state": "singularity",
  "first_singularity_round": 1,
  "conclusion": "...the converged (or dominant) conclusion...",
  "rounds": [ { per-round judge payload incl. each agent's response } ],
  "lang": "en",
  "web_search": "disabled ...",
  "error": null
}
```

7. Document & Image Input

Documents and images can be attached as evidence. Upload them first, then pass the returned `upload_session_id` to `run_council`.

Type	Extensions	How it is processed
PDF	.pdf	Text + tables (pdfplumber); OCR fallback for scanned / image-only pages.
Word	.docx, .doc	Document text extraction.
Spreadsheet	.xlsx, .xls, .csv	Per-sheet column/row summary.
Presentation	.pptx, .ppt	Slide text + OCR of embedded images.
Text	.txt	Raw text.
Image	.jpg, .jpeg, .png	OCR (Korean + English). With a vision-capable model, the image is <i>also</i> sent to the model to be seen directly.

```
# 1) upload (multipart field "files", repeatable) -> returns upload_session_id
POST /uploads

# 2) (optional) inspect what was extracted
GET /uploads/{session_id}/context

# 3) run the council on the uploaded evidence
run_council(query=..., upload_session_id="...")
```

HONESTY SAFEGUARD

When an image is uploaded, the agents are instructed to analyze it only if they can actually perceive it, and otherwise to state clearly that they cannot see it rather than fabricate visual details.

Uploads are ephemeral (stored in-container, no volume) and auto-deleted after the TTL (default 1 hour). Limits: 25 MB per file, 100 MB per session, 10 files (all configurable).

8. Answer Languages

The councilmen and the judge's conclusion are produced **directly in the target language by the model** (a system-prompt directive — no translation service). JSON field names always stay in English.

- **Automatic** (`lang="auto"` , default): the answer is produced in the language of the question (detected from the query).
- **Override**: pass a language to force it — e.g. `"en"` , `"ko"` , `"ja"` , `"de"` , `"fr"` .

The 31 supported languages: English, Korean, French, German, Spanish, Italian, Polish, Czech, Romanian, Hungarian, Croatian, Bulgarian, Portuguese, Portuguese (Portugal), Slovenian, Dutch, Greek, Norwegian, Swedish, Finnish, Danish, Icelandic, Estonian, Lithuanian, Latvian, Japanese, Russian, Ukrainian, Turkish, Arabic, Hebrew.

Output quality depends on the configured model's competence in that language; major languages are well supported by the mainstream Bedrock models.

9. Live Web Search (optional)

Live web search is **not bundled** — the Bedrock agent models have no built-in web access, and this product does not add a search dependency of its own. Without it, the councilmen reason from uploaded documents and model knowledge only.

A deployment may enable web search by configuring **Amazon Bedrock AgentCore Browser Tool** in its own account and setting `CM_WEB_SEARCH_MODE=agentcore` . AgentCore Browser is a separately billed AWS capability; enabling it is the buyer's choice and responsibility. When it is not configured, questions that require live web information (e.g. current events, schedules) simply cannot be verified, and the council will say so rather than guess.

10. Licensing & Entitlement

This product is offered under **AWS Marketplace contract pricing** and validates its entitlement continuously against **AWS License Manager**. No license keys or manual configuration are required — the container reads its entitlement automatically from the AWS account it runs in.

- While the contract is active, the server operates normally and `GET /health` returns **200**.
- If the contract expires or the entitlement is revoked, the MCP endpoint (`/mcp`) returns **HTTP 402 (Payment Required)** and `GET /health` returns **HTTP 503**. The process keeps running; renewing the subscription/contract restores service automatically, with no redeploy.
- Temporary AWS API disruptions are tolerated for a limited grace period before this takes effect.

Ensure the execution role has the license-manager permissions listed in Section 4, or the service stops responding after the grace period.

11. Health & Monitoring

Use `GET /health` on port 8000 as the container health check. It returns HTTP 200 with `{"status":"ok", ...}` while entitled, and HTTP 503 with `{"status":"license_expired", ...}` when the entitlement is not active. The response also reports the Bedrock region, number of configured models, and web-search status.

This image does not include `curl`. For an ECS/Kubernetes health check, use Python's stdlib, e.g. `python3 -c "import urllib.request,sys; sys.exit(0 if urllib.request.urlopen('http://localhost:8000/health').status==200 else 1)"`. Treat a 503 as unhealthy; if you do not want the task/pod restarted on entitlement loss, attach `/health` as a readiness (not liveness) probe.

12. Responsible Use & Compliance

The product is a multi-perspective decision-*support* tool. It does not automate any decision. The converged conclusion is a recommendation, not a binding determination; a human reviewer may reject, modify, or override any output at any time.

Topic	Position
EU AI Act — Transparency (Art. 50)	All agent outputs and the converged conclusion are AI-generated content and are identified as such.
EU AI Act — High-risk (Art. 6–49)	If deployed for an Annex III high-risk purpose, the deploying organization assumes the corresponding provider/deployer obligations.
GDPR	The container runs in the customer's account; inference calls go to Amazon Bedrock in the configured region. The customer is responsible for lawful basis, data-processing terms, and residency choices (region selection).

Human oversight

Human review is required before acting on any output.

⚠ NOT FOR INDIVIDUAL DETERMINATIONS

Do not use this product as the sole basis for decisions producing legal or similarly significant effects on a person. It is a research/oversight aid; outputs must be independently reviewed.

13. Support

For technical support, integration questions, or model-selection guidance, contact the seller through the support channel listed on the AWS Marketplace product page.