USER MANUAL

KlastroHeron Decision Intelligence

(4Councilmen Prudentia)

On-Premise Multi-Agent Decision Layer

Version 2.0.4

Klastrovanie Co., Ltd.

https://github.com/Klastrovanie/4councilmen/tree/Prudentia

PhD Dissertation, 2011. Prototype, 2026.

Table of Contents

1. What is 4Councilmen

1.1 Non-Convergent Models for Consensus

Four AI agents with orthogonal perspectives — surveillance, ethics, audit, transparency. They never compromise. They never seek consensus. When all four independently point to the same conclusion — that is the Fifth Response.

This model was built for a world where AI systems cannot share weights or training data due to business constraints — Gemini, Grok, GPT-5, Claude — each operating independently, each unable to collaborate directly.

4CM proposes a different path: instead of competing and wasting resources, orthogonal AI agents can publicly solve problems together without ever sharing their internals. Each model stays within its own perspective. Consensus is not negotiated — it emerges.

This is not about making AI agree. It is about finding the rare moments when independent minds, pushed to their limits, arrive at the same truth.

1.2 How It Is Different from Modern AI Systems

Modern AI systems are built by competing companies and trained on different or same common datasets. Their weights cannot be shared, their training pipelines cannot be merged. 4CM explores a different possibility: independent AI systems may still solve problems together — not by sharing data, but by converging on the same conclusion from different perspectives.

Every AI system today is built for convergence. Train on human feedback. Align to consensus. Optimize for agreement. 4CM is built on the opposite premise. The four agents are designed to never converge.

Think of a politician, a religious leader, an environmental activist, and a labor organizer sitting at the same table. They disagree on everything — by design, by identity, by conviction. Yet when all four, independently, point to the same conclusion — that conclusion is worth listening to.

They are structurally prevented from compromising. Consensus is not a goal — it is a rare event. And when it happens anyway — when four minds that were never meant to agree find themselves pointing at the same coordinate — that is the only result worth trusting.

1.3 How It Works

4CM does not force agreement between agents. Instead, answers are projected into a shared mathematical space. When independent reasoning collapses onto the same attractor, a singularity is detected.

The Golden Rule: Radical Orthogonality. The four agents are designed to be fundamentally different — even hostile — to one another’s logic.

1.4 Implementation Note

4CM does not force agreement between agents. Instead, answers are projected into a shared mathematical space. When independent reasoning collapses onto the same attractor, a singularity is detected. The Golden Rule: Radical Orthogonality. The four agents are designed to be fundamentally different — even hostile — to one another’s logic.

2. Architecture

2.1 System Components

Component	File	Role
Orthogonal Agents	orthogonal_agents.py	4 agents with fixed torus positions (±0.85, ±0.85)
Torus Engine	torus_math.py	f(x,y) convergence function, singularity detection
Semantic Judge	fourth_cm_engine.py	LLM-based scoring (conclusion + reasoning)
API Router	fourCM_router.py	SSE streaming, key management, scenario loading
Embedding Model	Sentence Transformer	all-MiniLM-L6-v2 — drift detection only
Document Parser	document_parser.py	PDF, DOCX, CSV upload and context injection
Prudentia UI	FourCM.tsx	React frontend with theme, language, endpoint config

2.2 Processing Pipeline

Each round follows this sequence:

1. Query input with scenario prompt loaded from angry_agents/ folder
2. 4 orthogonal agents generate independent responses (Claude, Grok, or local LLM)
3. Sentence Transformer embeds responses → drift detection → torus positions (geometric layer)
4. Semantic judge (Grok or local LLM) scores conclusion_convergence (X) and reasoning_convergence (Y)
5. Scores mapped to torus coordinates via sigmoid: score ≥ 0.5 → coordinate 0.83 (peak)
6. Torus gate f(x,y) evaluated — if singularity detected, common_conclusion promoted as 5th response
7. If no singularity, responses injected as context for next round

2.3 Torus Mathematics

The convergence function is $f(x,y) = (x^4 + y^4) \cdot e^{-(x^8 + y^8)}$, with parameters a=0, b=0, a1=b1=4, c=d=8 from Equation (2) of the 2011 PhD dissertation.

Key values on the torus surface:

Property	Value	Meaning
Peak coordinates	(±0.8409, ±0.8409)	4 agent positions — maximum activation
Peak height	f = 0.6065	All peaks exceed 0.5 activation threshold
Activation threshold	Z = 0.5	Sigmoid-derived — peaks are activated above this
Saddle point	Z = 0.4289	Channel opens — 4 islands connect into ring
Agent initial position	(±0.85, ±0.85)	Code constant — near analytical peak
Score mapping peak	0.83	Target coordinate when score ≥ 0.5

Agents do not move toward consensus. They remain fixed at their peaks. The convergence point moves toward them as scores increase. When it lands on all four peaks simultaneously, singularity occurs.

3. Deployment

3.1 Prerequisites

Requirement	Details
Docker	Docker Engine 20+ with Compose V2
Foundation Model	Ollama (local) or API keys (Claude, Grok, Gemini, GPT)
Hardware (local LLM)	GPU with 8GB+ VRAM recommended (e.g., Tesla T4 for 14B models)
Network	Port 80 (nginx) — internal only, no public exposure required

3.2 Quick Start

```
git clone https://github.com/Klastrovanie/4councilmen.git
cd 4councilmen
git checkout Prudentia
cp the.env.example .env
# Edit .env with your configuration
docker compose up -d --build
# Access at http://localhost
```

3.3 Environment Configuration (.env)

Rename the.env.example to .env before first run.

Variable	Default	Description
INTERNAL_LLM_BASE_URL	http://localhost:11434/v1	Ollama endpoint. Use host.docker.internal for Docker.
INTERNAL_LLM_MODEL	llama3.1:8b	Model name as shown in 'ollama list'
ANGRY_AGENTS_PATH	/app/angry_agents	Scenario folder path (set by docker-compose)
FOURCM_PATH	/app	Application root (set by docker-compose)
ANTHROPIC_API_KEY	(empty)	Optional fallback — prefer UI or secret manager
XAI_API_KEY	(empty)	Optional fallback — prefer UI or secret manager
FOURCM_MAX_FILE_BYTES	26214400	Max upload file size (25MB)
FOURCM_UPLOAD_TTL_SECONDS	3600	Upload session expiry (1 hour)

Important: When running inside Docker, replace localhost with host.docker.internal (Mac/Windows) or 172.17.0.1 (Linux). Use the port and path matching your LLM server (e.g., Ollama: 11434/v1, vLLM: 8000/v1, llama.cpp: 8080/v1).

4. Usage

4.1 Provider Modes

Mode	Agents	Judge	Data Flow
All Local LLM	Local model ×4	Local model (fixed)	No data leaves infrastructure
Round-robin	Claude→Grok→Claude→Grok	Grok (fixed)	Data sent to Anthropic + xAI
All Claude	Claude ×4	Grok (fixed)	Data sent to Anthropic + xAI
All Grok	Grok ×4	Grok (fixed)	Data sent to xAI
Per-agent	User-assigned per agent	Grok or local	Depends on configuration

The judge (semantic scoring) is always a single fixed model to ensure cross-round score comparability. In external API mode, the judge is always Grok. In local mode, the judge uses the same local LLM.

4.2 API Keys

v2.0.4 Security Change: API keys entered via the UI are stored in memory only and cleared when the container restarts. They are never written to disk. For persistent keys, use your organization's secret manager (Docker Secrets, HashiCorp Vault, or cloud-native solutions) or the env_file mechanism in docker-compose.yml.

4.3 LLM Endpoints

The LLM Endpoints tab in the API Keys modal allows per-agent endpoint configuration. Each agent can point to a different local LLM instance. These settings are stored in the browser's localStorage and are per-user, per-browser. They are not transmitted to the server.

4.4 Scenarios

Scenarios are stored in the angry_agents/ directory, mounted as a Docker volume. Each scenario folder contains:

File	Purpose
members.txt	Agent names (e.g., 1: SENTINEL, 2: ETHIKOS, 3: AUDITOR, 4: HERALD)
1.txt – 4.txt	System prompts for each orthogonal agent
query.txt	Default query for the scenario
risk.txt	Risk level (normal / high)
title.txt	Display title

Pre-built scenarios (pharma, outbreak, government, M&A, Oppenheimer, whistleblower, key talent) are created automatically on first run by entrypoint.sh. Custom scenarios can be added by creating new folders in angry_agents/.

Note: Pre-built scenarios marked [MOCK RESEARCH SCENARIO] are for research and simulation purposes only. They are not validated for clinical, financial, legal, or regulatory decision-making.

4.5 Document Upload

The Software supports injecting documents (PDF, CSV, XLSX, DOCX, PPTX, TXT, images) into the query context. Documents are parsed server-side, and extracted text is prepended to the query for all agents. Uploaded files are stored in /app/tmp_uploads/ (ephemeral, no volume mount) and automatically deleted after the TTL expires (default: 1 hour).

5. Security

5.1 API Key Handling (v2.0.4)

As of version 2.0.4, the Software does not write API keys or authentication credentials to persistent storage.

Aspect	Behavior
UI key entry	Stored in os.environ (memory) only
Container restart	Keys are cleared — re-enter via UI or inject via secret manager
.env file	Never contains API keys written by the application
Logs	Key values are never printed to application logs
To Klastrovanie	No keys are transmitted to Klastrovanie in any mode

5.2 Secret Manager Integration

For persistent key management in enterprise deployments, use one of the following methods:

Method	How
Host environment variables	export ANTHROPIC_API_KEY=... before docker compose up
Docker Secrets (Swarm)	Mount secrets to /run/secrets/ — entrypoint.sh converts to env vars
HashiCorp Vault	Add vault kv get commands to entrypoint.sh
Cloud secret managers	AWS Secrets Manager / Azure Key Vault / GCP Secret Manager

5.3 Data Flow

Component	Internal LLM Mode	External API Mode
Agent Queries	Customer's servers only	Customer → Foundation Model Provider
Agent Responses	Customer's servers only	Provider → Customer
Embedding (ST)	Customer's servers only	Customer's servers only
Torus Computation	Customer's servers only	Customer's servers only
Decision Logs	Customer's servers only	Customer's servers only
To Klastrovanie	NONE	NONE

5.4 Security Checklist for Administrators

- Select secret manager integration method (A/B/C/D above)
- Verify no API keys exist in .env file after initial setup
- Confirm keys are cleared after container restart (curl /fourCM/keys/status)
- Enable access control and audit logging on your secret manager
- Restrict .env file permissions to owner-only (chmod 600 .env)
- When decommissioning: shred .env and all files in logs/ directory

6. Regulatory Compliance

6.1 Intended Purpose

The Software is a multi-perspective decision-support and simulation tool. It does not automate any decision. All final decisions rest with the human operator. The fifth response (common_conclusion) is a recommendation, not a binding determination. The operator may reject, modify, or override any output at any time.

6.2 EU AI Act (Regulation 2024/1689)

Requirement	Status
GPAI Provider Obligations (Art. 51–56)	NOT APPLICABLE — no Foundation Model provided
High-Risk Obligations (Art. 6–49)	NOT APPLICABLE as deployed — limited-risk classification
Transparency (Art. 50)	SATISFIED — UI discloses AI-generated content
Open-Source Exemption (Art. 2(12))	NOT CLAIMED — dual licensing = monetization
ISO 42001 / EN 18286	NOT REQUIRED — voluntary for limited-risk

Warning: If the Software is used for Annex III high-risk purposes (employment decisions, credit scoring, etc.), the deploying organization assumes full provider/deployer obligations under Article 25.

6.3 GDPR (Regulation 2016/679)

In Internal LLM Mode, all data remains within the customer's infrastructure. Klastrovanie is neither a data controller nor processor. In External API Mode, queries are transmitted to Foundation Model providers (typically US-based). The customer is responsible for establishing appropriate Data Processing Agreements and ensuring adequate GDPR Chapter V safeguards.

6.4 Cyber Resilience Act (Regulation 2024/2847)

Full CRA obligations apply from late 2027. Klastrovanie commits to: vulnerability disclosure channel, SBOM delivery, and security patch notifications within the support SLA timeframe.

6.5 Transparency Notices

The following notices must be maintained in any deployment:

- All agent outputs and the fifth response are AI-generated content
- The Software orchestrates multiple AI models for decision support, not automated decision-making
- Human review is required before acting on any output
- Pre-built scenarios are for research and simulation purposes only

7. API Reference

7.1 Endpoints

Method	Path	Description
GET	/health	Health check — returns status, version, engine
GET	/fourCM/agents	List available scenario sets
GET	/fourCM/agents/{set}	Load specific scenario with agent prompts
POST	/fourCM	Run 4CM analysis (SSE streaming)
POST	/fourCM/validate	Orthogonality check on agent prompts
POST	/fourCM/keys/save	Save API keys to memory (not disk)
GET	/fourCM/keys/status	Check if keys are set (values never returned)
DELETE	/fourCM/keys/clear	Clear API keys from memory
GET	/fourCM/config	Current LLM configuration
POST	/fourCM/upload	Upload documents for context injection

7.2 SSE Stream Events

The POST /fourCM endpoint returns Server-Sent Events with the following event types:

Event	Data
agents_loaded	Agent names and prompts for the loaded scenario
round_start	Round number and configuration
agent_response	Individual agent response with name and content
convergence	Torus coordinates, f(x,y) value, singularity status
singularity	Fifth response (common_conclusion) and convergence scores
round_end	Round summary with all scores
complete	Final summary with all rounds
error	Error message if processing fails

8. Troubleshooting

Symptom	Cause	Fix
{"sets":[]}	ANGRY_AGENTS_PATH misconfigured	Check docker-compose.yml environment section
ConnectionError to localhost:11434	Docker can’t reach host Ollama	Use host.docker.internal:11434 in .env
Model not found	Wrong INTERNAL_LLM_MODEL	Run 'ollama list' and use exact model name
Keys cleared after restart	v2.0.4 memory-only design	Use secret manager or env_file for persistence
Slow responses (local LLM)	14B model on limited GPU	Try smaller model (7B) or reduce rounds
500 on /keys/clear	Code error in _write_env_file	Verify fourCM_router.py has correct v2.0.4 patch
Upload files lost	/app/tmp_uploads not mounted	By design — uploads are ephemeral (TTL-based)

9. References

Resource	URL
GitHub Repository	https://github.com/Klastrovanie/4councilmen
Prudentia Branch	https://github.com/Klastrovanie/4councilmen/tree/Prudentia
Release v2.0.4	https://github.com/Klastrovanie/4councilmen/releases/tag/v2.0.4
PhD Dissertation (2011)	https://dl.acm.org/doi/book/10.5555/2231522
License	AGPL-3.0 (open source) / Commercial dual license

Klastrovanie Co., Ltd.
Yongwoo Chung, CEO & Founder