MULTI-AGENT COHERENCE IN ARTIFICIAL INTELLIGENCE SYSTEMS: EXPERIMENTAL STUDY OF FIVE ROLES, LANGUAGE ARCHITECTURE, AND SELF-ORGANIZATION MECHANISMS BASED ON THE ODTOE FORMALISM

(Multi-Agent Coherence in AI Systems: Experimental Study of Five Roles, Language Architecture, and Self-Organization Mechanisms Based on the ODTOE Formalism)

Experimental study of multi-agent coherence, role specialization, and language architecture in AI systems based on the Observer-Dependent Theory of Everything formalism

Pankratov Anton Sergeevich
Independent researcher, Kazan, Russia
E-mail: anton.s.pankratov@gmail.com
ORCID: 0009-0002-4870-2995

UDC 004.89 + 519.876 + 81'322

ABSTRACT

This paper presents experimental results on multi-agent coherence in AI systems based on the Observer-Dependent Theory of Everything (ODTOE) formalism. A five-role architecture was developed and experimentally verified, in which five specialized roles (Visionary, Analyst, Builder, Validator, Coherencer) operate in parallel via the Round Table protocol. Four key experiments were conducted: (1) a large-scale framework analysis by 25 agents (5 teams of 5 roles), resulting in 2.5x prompt compression without content loss and discovery of a P_{coll} formula error; (2) an A/B experiment "Russian vs English" (10 agents) showing that English prompts yield 48% higher B-scores for practical tasks, while Russian agents demonstrate superiority in theoretical depth; (3) an A/B experiment on entry point architecture (10 agents) revealing that an external router splits the team across language stacks; (4) analysis of a real deployment session (157 tool calls, 0 Round Tables) that uncovered the Lambda problem and led to a three-level enforcement system. The adjusted coherence formula $S_{\text{adjusted}} = S_{\text{team}} \times \bar{B}$ is introduced, detecting phantom coherence — a state of high agreement with low quality. A full linguistic analysis of 12 framework files, a synchronization audit (16 desynchronizations), and a four-layer bilingual architecture are presented. The Check-First Pipeline methodology for pre-generation artifact verification is developed. It is established that prompt language is not a neutral instruction carrier but an active observation operator $\hat{O}$ that configures the agent's cognitive space. A bilingual architecture is proposed where English provides breadth (practical tasks, bug detection) and Russian provides depth (theoretical innovations, mathematical formulas).

Keywords: multi-agent systems, coherence, ODTOE, prompt engineering, language architecture, Round Table, LLM, role specialization, phantom coherence, observer-dependent theory, Lambda problem, bilingual architecture, Check-First Pipeline, bootstrap enforcement.

АННОТАЦИЯ

Представлены результаты серии экспериментов по исследованию мультиагентной когерентности в системах искусственного интеллекта на основе формализма наблюдатель-зависимой теории всего (ODTOE). Разработана и экспериментально верифицирована пятиролевая архитектура, в которой пять специализированных ролей (Визионер, Аналитик, Строитель, Валидатор, Когерент) работают параллельно через протокол Round Table. В ходе исследования проведены четыре ключевых эксперимента: (1) масштабный анализ фреймворка 25 агентами (5 команд по 5 ролей), результатом которого стало сжатие промптов в 2,5 раза без потери содержания и обнаружение ошибки в формуле P_{coll}; (2) A/B-эксперимент «русский язык vs английский язык» (10 агентов), показавший, что англоязычные промпты дают B-score на 48% выше для практических задач, тогда как русскоязычные агенты демонстрируют превосходство в теоретической глубине; (3) A/B-эксперимент архитектуры точки входа (10 агентов), определивший, что внешний маршрутизатор раскалывает команду на разные языковые стеки; (4) анализ реальной сессии развёртывания (157 tool calls, 0 Round Tables), выявивший Lambda-проблему и приведший к созданию трёхуровневой системы enforcement. Введена формула скорректированной когерентности $S_{\text{adjusted}} = S_{\text{team}} \times \bar{B}$, обнаруживающая фантомную когерентность. Проведён полный лингвистический анализ 12 файлов фреймворка с языковой картой, аудит синхронизации (16 десинхронизаций), и предложена четырёхслойная двуязычная архитектура. Разработана методология Check-First Pipeline для предгенерационной верификации артефактов. Результаты имеют значение для проектирования мультиагентных систем ИИ, оптимизации промпт-инженерии и понимания роли естественного языка в формировании когнитивных конфигураций искусственных наблюдателей.

Ключевые слова: мультиагентные системы, когерентность, ODTOE, промпт-инженерия, языковая архитектура, Round Table, LLM, ролевая специализация, фантомная когерентность, наблюдатель-зависимая теория, Lambda-проблема, двуязычная архитектура, Check-First Pipeline, bootstrap enforcement.

I. INTRODUCTION

Modern large language models (LLMs) are capable of solving complex tasks individually; however, scaling to multi-agent systems gives rise to a fundamental

coordination problem: how can multiple AI agents be made to work coherently without duplicating effort or contradicting one another? This problem is analogous to the classical challenge of managing distributed teams in software development, yet it has specificities related to the nature of LLMs: the absence of persistent memory between sessions, quality dependence on instruction language, and a tendency to "collapse" into a single-executor mode.

This paper proposes an approach to solving this problem based on the Observer-Dependent Theory of Everything (ODTOE) formalism [1], in which each AI agent is treated as an observer with an individual observation operator $\hat{O}$, and the collective work of a team is described through the coherence metrics B, S_{team}, and P_{coll}. A five-role architecture is developed that formalizes AI agent roles and their interaction protocol.

The paper is based on experimental data collected during a research session involving more than 80 agents on a multi-agent LLM orchestration platform across 11 task blocks [4]. The session, initially classified as M (medium), organically grew to XL (extra-large), progressing through large-scale framework analysis, two A/B experiments, linguistic analysis, synchronization audit, real deployment session analysis, and a complete framework rebuild. This evolution itself became experimental confirmation of the spiral gap [1]: each completed cycle revealed a residual ($\sim$2%) that fed the next iteration.

Main research questions:

1. How does role distribution among AI agents affect the quality of collective output?
2. Which language (Russian or English) ensures higher AI agent coherence, and does this depend on task type?
3. What is the optimal entry point architecture for a multi-agent system?
4. Why does an agent that has read the entire framework ignore its prescriptions, and how can this be prevented?
5. What is the role of prompt language as an observation operator in shaping the cognitive space of an AI agent?

II. THEORETICAL FOUNDATION

II.1. Agent Cognitive Coherence (ODTOE)

The quality of an AI agent's work is formalized through the multiplicative cognitive coherence formula [1]:

$$B(\text{agent}) = F^{w_1} \cdot E^{w_2} \cdot (1-\sigma)^{w_3} \cdot \Lambda^{w_4} \tag{II.1}$$

where F is the attention focus (whether all relevant files have been read), E is goal alignment (whether the agent is solving exactly the assigned task), $(1-\sigma)$ is

consistency (whether there are conflicts in the result), Λ is accumulated experience (whether project memory has been used); $w_1 + w_2 + w_3 + w_4 = 1$, $w_i \in (0.1)$. By default $w_1 = w_2 = w_3 = w_4 = 0.25$ (uniform distribution by default — a constructive choice, not derived from the axiomatics; specific values are subject to experimental determination [1]).

The critical property of the formula is multiplicativity: zeroing any component zeroes the entire result (the weakest-link principle [1]). An agent with perfect focus ($F = 1$) but zero experience ($\Lambda = 0$) has $B = 0$. This property determines the diagnostic strategy: when B is low, there is no need to improve all components simultaneously — it suffices to find the zero link.

II.2. Team Collective Coherence

The coherence of a team of n agents [3]:

$$S_{\text{team}} = 1 - \frac{2}{n(n-1)} \sum_{i<j} |B_i - B_j| \tag{II.2}$$

The formula measures agreement — how close the B-values of agents are to each other. However, S_{team} does not reflect absolute quality: a team of five agents with $B_i = 0.1$ yields $S_{\text{team}} = 1.0$ (perfect agreement at zero quality).

To address this problem, the present work introduces adjusted coherence:

$$S_{\text{adjusted}} = S_{\text{team}} \times \bar{B}, \quad \bar{B} = \frac{1}{n} \sum_{i=1}^{n} B_i \tag{II.3}$$

The formula detects *phantom coherence* — a state in which $S_{\text{team}} > 0.7$ but $S_{\text{adjusted}} < 0.5$, meaning: agents are aligned, but aligned around an error.

II.3. Probability of Collective Collapse

The probability of observation collapse for a team [1]:

$$P_{\text{coll}} = \frac{1}{n^k} \left(\sum_{i=1}^{n} B_i \right)^k \tag{II.4}$$

where $k \geq 1$ is a parameter depending on task complexity (a context-dependent quantity [1]). In the linear case ($k = 1$) the formula simplifies to $P_{\text{coll}} = \bar{B}$. An error in the computation of P_{coll} at $B = 0.3$, $n = 3$, $k = 1$ is one of the key findings of Experiment 1 (see Section IV).

II.4. Five Roles as Observation Operators

Each role defines a specific observation operator $\hat{O}_r$, projecting the task into the role's configuration space:

Role	Key Question	ODTOE Analog	Dominant B	Torus Ring
Visionary	What and why?	Ψ (state field)	Λ	Outer
Analyst	How exactly?	$\hat{O}$ (observation operator)	F	Bridge
Builder	What specifically to do?	R (configuration)	E	Inner
Validator	Does it conform?	ι (embedding)	$(1-\sigma)$	Inner
Coherencer	Do we see the same thing?	S (synchronization)	All	Outer

Toroidal communication topology [8]: inner ring (r, fast) — Analyst $\leftrightarrow$ Builder $\leftrightarrow$ Validator; outer ring (R, slow) — Visionary $\leftrightarrow$ Coherencer $\leftrightarrow$ Analyst. The ratio $R/r = \varphi$ (golden ratio[1]) ensures maximum stability by the KAM theorem [2].

II.5. Activation Operator

Each agent executes a four-stroke activation operator before generating a response [4]:

$$\hat{A} = A_\Lambda \circ A_\sigma \circ A_E \circ A_F \tag{II.5}$$

The operator sequence is fixed: first focus (A_F: load all necessary materials), then alignment (A_E: verify that the task is correctly understood), consistency check (A_σ: no conflicts with existing work), and experience application (A_Λ: extract relevant patterns from memory). After generating a response, each agent performs self-diagnostics [9] with a numerical assessment of all four B components.

III. LINGUISTIC ANALYSIS OF THE FRAMEWORK

III.1. Language Map of Files

A full linguistic analysis was conducted using the Round Table method (5 roles in parallel). For each of the 12 core framework files, the language proportion was determined:

File	Lines	Primary Language	RU%	EN%

[1] $\varphi = 1.6180339887498948482045868343656381177203091798057 6$ — 50 significant digits.

Framework core	347	English	8%	92%
Meta-protocol	200	English	3%	97%
Role: Visionary	68	RU + EN terms	85%	15%
Role: Analyst	68	RU + EN terms	85%	15%
Role: Builder	72	RU + EN terms	80%	20%
Role: Validator	74	RU + EN terms	83%	17%
Role: Coherencer	79	RU + EN terms	82%	18%
Glossary	93	RU + EN formulas	65%	35%
Entry documentation	114	Russian	85%	15%
Checklist	93	RU + EN technical	80%	20%
Memory template	100	Russian	75%	25%
Project documentation	100	Russian	88%	12%

Structural conclusion: the framework core (constitution + meta-protocol = 547 lines) is written in English, the operational layer (10 files = 861 lines) is in Russian. This duality represents a systemic non-uniformity (Mura in TPS terminology [9]).

III.2. Token Inefficiency

Russian text consumes 1.5–2.5 times more tokens than equivalent English text. The reason lies in the architecture of BPE tokenizers (Byte Pair Encoding): a Cyrillic character is encoded in 2 bytes in UTF-8 (range U+0400–U+04FF), whereas a Latin character uses 1 byte. Since tokenizer training was conducted on corpora where 60–90% of data is in English, BPE pairs for Latin script are significantly longer (one English word = 1–2 tokens) than for Cyrillic (one Russian word = 3–5 tokens). When loading the full stack (core + role + memory + checklist) into an agent's context, the Russian-language portion occupies a disproportionately large share of the context window.

Practical implication: translating the operational layer to English frees 30–40% of the context budget, allowing more information to be loaded within the same constraint.

III.3. LLM Benchmarks: The Cross-Language Gap

All 5 roles in the linguistic analysis unanimously confirmed: English ensures more precise adherence to structured instructions. Empirical grounds:

1. **Training corpus:** 60–90% of LLM training data is in English. Benchmarks MMLU, MGSM, and XCOPA demonstrate a 5–15% gap in favor of English.
2. **Tokenization:** as shown in III.2, one English word = 1–2 tokens, one Russian word = 3–5 tokens. More information per token means more instructions in the context window.
3. **Imperative constructions:** directives such as "MUST", "NEVER", "BEFORE generating output" carry a stronger pragmatic effect — models have seen

them millions of times in system prompts, API documentation, and technical specifications. Russian equivalents appear in training data orders of magnitude less frequently.

4. **Homogeneity with code:** all variable names, commands, and configurations are in English. A prompt in the same language eliminates the "translational bridge" between instruction and execution.

III.4. Terminological Bifurcation

Language mixing creates terminological bifurcation: a single term acquires multiple forms. For an LLM, each form is a separate token with separate associations. The framework core uses English terms, the glossary uses Russian calques, and role prompts use hybrid formulations. The validation tool fails its own validation for terminological non-uniformity. This recursive contradiction is a special case of a strange loop [10].

III.5. Consensus of Five Round Table Roles

During the linguistic analysis, all 5 roles (Visionary, Analyst, Builder, Validator, Coherencer) worked in parallel and were surveyed on four key questions. The results represent a full RT consensus.

Question 1. Why did the files end up in different languages?

There was no deliberate strategy. The constitution and meta-protocol were created in English because agents by default optimize for LLMs (English is the models' working language). Role prompts were written in Russian because the user session context was Russian-language. The result is not an architectural decision but a systemic non-uniformity (Mura): an unintentional heterogeneity arising from the absence of a language policy.

Question 2. In which language do LLMs work more precisely?

All 5 roles are unanimous: English. Grounds: (a) 5–15% gap on MMLU, MGSM, XCOPA benchmarks; (b) token efficiency 1.5–2.5 times higher; (c) no "translational bridge" between prompt and code — the prompt is in the same language as variables, commands, and configurations. Detailed data are provided in III.3.

Question 3. Is a new language (DSL) needed?

All 5 roles: no. A framework metalanguage already exists de facto: mathematical formulas + TPS terms + process keywords. It does not need to be invented — it needs to be standardized. The detailed three-level structure of the metalanguage is given in III.6.

Question 4. Recommendation on language architecture.

Consensus: English core + localizable user layer. All LLM prompts in English. Entry documentation is bilingual (EN + RU). Project files are in the team's language. The detailed architecture is given in III.9.

III.6. Metalanguage Standardization: Three Levels

All 5 roles agreed: a full-fledged DSL (Domain-Specific Language) is unnecessary. A formalized notation — yes. The framework has already created a de facto metalanguage existing at three levels:

Level	Type	Examples	Property
1	Invariant (math symbols)	B, S, P_{coll}, $T(C)$, Φ, Ψ, $\hat{O}$, ι, σ, Λ, F, E, d	Identical in any language. Not translated, not transliterated
2	Terminological (canonical forms)	Jidoka, Andon, Round Table, Kill-Switch, True North	One canonical form + one permissible translation per term. Synonyms = non-uniformity
3	Operational (inter-agent communication)	`[RT-2][Coherencer][S=0.68<0.7] TRIGGER: Kill-Switch L1. SOURCE: |B_Builder - B_Validator| = 0.35. ROOT: Builder.F=0.4. ACTION: A_Lambda re-run for Builder.`	Actual inter-agent interaction protocol

Level 1 (invariant) contains the mathematical symbols of the ODTOE formalism. These symbols are identical in Russian and English text and are not subject to translation or transliteration.

Level 2 (terminological) fixes the canonical form of each term. Rule: one canonical form (English) + one permissible translation (Russian) per term. Any other variants — synonyms, transliterations, paraphrases — are prohibited in agent prompts and classified as non-uniformity (Mura).

Level 3 (operational) defines the inter-agent communication format. Example of a complete agent message in the Round Table protocol:

```
[RT-2][Coherencer][S=0.68<0.7] TRIGGER: Kill-Switch
L1.
SOURCE: |B_Builder - B_Validator| = 0.35.
ROOT: Builder.F=0.4 (missed context from prior
iteration).
ACTION: A_Lambda re-run for Builder.
```

This format is already used by agents de facto. Standardization means: fixing the template in the meta-protocol and requiring all RT messages to follow it.

Form of standardization: extension of two existing artifacts — the glossary (levels 1–2) and the meta-protocol (level 3). Plus one rule in the framework core: "LANGUAGE POLICY: Terms from the glossary are used in their canonical form. No synonyms, no translations within agent prompts."

III.7. Assessment of Language Non-Uniformity Impact

By the decoherence formula $D(\eta) = D_0 \cdot (1 - S)$ [1], linguistic splitting introduces $\Delta S \approx 0.05–0.10$ (Coherencer's estimate). This exceeds the permissible spiral gap $(\pi - 3)^2 \approx 0.02$ by 2.5–5 times. Eliminating language non-uniformity is the cheapest way to increase S_{team}: there is no need to improve the quality of each agent (difficult); it suffices to remove artificial discrepancies between them (simple).

III.8. Phantom Decoherence

When computing S_{team}, the discrepancy $|B_i - B_j|$ between agents may be an artifact of terminological confusion rather than a genuine divergence in task understanding. If one agent uses Russian terms from the glossary while another uses English terms from the framework core, their formulations will diverge formally, even though they may be describing the same thing substantively. Linguistic noise masquerades as substantive disagreement — phantom decoherence.

This is the mirror reflection of phantom coherence (Section VII): if phantom coherence is false agreement amid real differences, then phantom decoherence is false disagreement amid real unity. Both artifacts distort the S_{team} metric, and both are eliminated by unified terminology.

III.9. Recommended Language Architecture

Based on the analysis, the model "English core + localizable user layer" is proposed:

Layer	Contents	Language	Rationale
Core	Constitution, meta-protocol, role prompts, glossary, checklist	EN	Read by LLM at every startup
DSL terms	B, S, Jidoka, Andon, Round Table, Kill-Switch	Not translated	Proper nouns
Documentation	Human entry point	EN + RU (two files)	Standard open-source practice
Projects	Project directories	Team language	RU for internal, EN for international
RT reports	Analysis templates, spiral log	User language	Operational artifacts

Expected effect of the transition: elimination of Mura (unified core language), 30–40% token savings when loading context, scalability to international teams, improved agent accuracy (no language context switching), framework self-consistency (passes its own non-uniformity validation).

III.10. Discussion: Hybrid vs. Unified Language

The question of language strategy does not have a trivial answer. During the linguistic analysis, substantial arguments were recorded on each side.

Arguments FOR the current hybrid (EN core + RU roles):

1. The user formulates tasks in Russian. The agent receives a Russian-language role prompt, thinks in a Russian-language context, and responds in Russian — there is no translational bridge between the role prompt and the user task.
2. The Coherencer noted: ODTOE semantics were created in Russian. "Entropy of doubts" $\neq$ "doubt entropy" in connotation — the Russian-language formulation carries an additional semantic layer linked to the philosophical tradition.
3. Contextual proximity: a role prompt in the user's language minimizes the cognitive distance between instruction and task.

Arguments FOR full translation to EN:

1. Token savings of 30–40% — the Russian-language portion of the operational layer occupies a disproportionately large share of the context window.
2. Unified language with the core — elimination of non-uniformity at the terminological level. No term bifurcation.
3. LLMs follow imperative instructions in English more precisely (benchmarks 5–15% in favor of EN).
4. Scalability: open-source, international teams, publications.

Arguments AGAINST premature translation:

1. The Builder honestly assessed his own $B = 0.403$ and proposed: "run an experiment — the same task through RT on Russian prompts and English prompts, compare B-metrics." Without data, this is a decision based on intuition rather than evidence.
2. Translation without experimental validation violates the principle of A/B experiment priority (Section XII).

Resolution: the A/B experiment was conducted (Section V). The data showed that a bilingual strategy is optimal: EN prompts for practical tasks and bug detection, RU prompts for theoretical depth and mathematical innovations. This is not a compromise but a functional architecture — each language solves its own class of tasks.

III.11. Qualitative Comparison of RU and EN Groups

Based on the results of the A/B experiment (detailed quantitative data in Section V), a qualitative comparison of the outputs of the two groups was conducted across seven criteria:

Criterion	RU Group	EN Group	Leader
Originality	φ-weighted S_{team}, Graduated Activation, Phase-Adaptive Weights	S_{adjusted}, Interface Contracts, Loading Protocol	RU
Practicality	Formula-based proposals, less specific directives	Responsibility distribution table, precise contract formats, specific implementation lines	EN
Diversity	5 agents converged on 2–3 ideas	5 agents covered different aspects	EN
Formula depth	φ-weights for torus, graduated activation with thresholds	Simpler $S_{\text{adjusted}} = S \times \bar{B}$	RU
Bug detection	Found: w_i indeterminacy, absence of RT protocol	Found the same + broken links, glossary duplication, language non-uniformity	EN
Blind spots	Cannot see the language problem (inside RU context)	Cannot produce φ-weighted S_{team}	Both have unique blind spots
Format adherence	5/5	5/5	Parity

III.12. Key Finding: Language as an Observation Operator

The RU group thinks deeper, the EN group sees wider. This is not an evaluative judgment but an experimentally established fact.

RU agents delve into mathematical formulas — φ-weights for toroidal topology, Graduated Activation with numerical thresholds, Phase-Adaptive Weights. Their contribution is theoretically more valuable: the φ-weighted S_{team} formula for toroidal topology emerged only from the RU Analyst and was not reproduced by any EN agent. The mathematical depth of the Russian-language context is apparently linked to the activation of the abstract-theoretical mode of the LLM, characteristic of processing

Slavic languages with rich morphology.

EN agents scan the entire system and find specific errors — broken links, glossary duplication, language non-uniformity. Their contribution is operationally more valuable: the discovered bugs can be immediately fixed, the responsibility distribution table provides a concrete action plan, and interface contracts formalize inter-agent agreements.

Critical observation about blind spots: the RU group *cannot* see the language problem — they are inside the Russian-language context, and bilingualism is invisible to them. The EN group *cannot* achieve the mathematical depth of RU — the φ-weight formula for the torus emerged only from the RU Analyst. Each group has a unique blind spot, invisible from within and visible only from outside.

In the ODTOE formalism [1]: the same LLM with different language prompts constitutes different observers ($\hat{O}_{\text{RU}} \neq \hat{O}_{\text{EN}}$), projecting the same task (Ψ) into different configurations ($R_{\text{RU}} \neq R_{\text{EN}}$). Language is not a transmission channel but an observation lens, and the complete picture is available only when both projections are combined.

III.13. Bilingual Agent Routing

Based on data from Experiment 2 (Section V) and the qualitative comparison (III.11), a routing table was derived that determines the optimal prompt language depending on task type:

Task Type	**Agent Language**
Theoretical development, mathematical innovations	RU
Practical refinement, bug detection	EN
Mixed task (theory + practice)	BOTH (parallel groups, Coherencer synchronizes)
Ambiguous task	EN (default)

The entry point (constitution) is always in English. The routing table is fixed in Section Zero of the constitution. When receiving a mixed task, the Coherencer launches both groups in parallel and synthesizes results at a Round Table — this is the most resource-intensive but also the most effective strategy.

III.14. Verdict: Preferred Language by Task Type

Summary table consolidating all experimental data:

Task	**RU**	**EN**

For theoretical framework development	Better	—
For practical framework refinement	—	Better
For uncovering hidden problems	—	Better
For mathematical innovations	Better	—

The ideal strategy: EN prompts for breadth (practical tasks, bug detection, operational coverage) + RU tasks for theoretical depth (mathematical formulas, conceptual innovations, abstract conclusions). The optimal mode is launching both groups in parallel and synthesizing results at a Round Table, where both perspectives collide and produce a result that surpasses each individually.

IV. EXPERIMENT 1: LARGE-SCALE FRAMEWORK ANALYSIS (25 AGENTS)

IV.1. Design

25 agents (5 teams $\times$ 5 roles) investigated two versions of the framework in parallel. The assignment for each team:

Team	Agents	Assignment
RT1	5	Current version analysis: structure, completeness, internal consistency
RT2	5	Previous version analysis: identification of weaknesses and limitations
RT3	5	Version comparison: evolutionary patterns, what was added, what was lost
RT4	5	ODTOE theory: formula validation, constant recalculation, derivability verification
RT5	5	Synthesis: creation of an improved framework based on 20 reports from RT1–RT4

IV.2. Findings by Team

RT1 discovered duplication: 25–30% of content was repeated across multiple files. The defensive mechanisms block [9] was present in 5 files simultaneously, each time with micro-variations of wording that created false $|B_i - B_j|$ when comparing agents using different files.

RT2 identified the Lambda problem in the previous version: rules accumulated, but a mechanism for their automatic application was absent. An agent could load 400+ lines of instructions and then ignore them — knowledge without enforcement.

Constructive choice: automatic execution was built into the current version through a mandatory auto-loading mechanism.

RT3 discovered an evolutionary pattern: each version added new mechanisms without removing outdated ones. As a result, the framework grew linearly while effective density (new information / total volume) declined.

RT4 recalculated ODTOE formulas and found a critical error: the value of P_{coll} at $B = 0.3$, $n = 3$, $k = 1$ was stated as 0.61 — the correct value is 0.657. The error had propagated from the previous version into the current one and further into several papers. All 20 analytical agents (RT1–RT3) missed this error — only the RT4 Validator, performing an independent recalculation, discovered the discrepancy. This confirms: formula verification requires independent recalculation, not consensus.

RT5 synthesized 20 reports (472 KB) into an updated version of the framework.

IV.3. Quantitative Results

Metric	Before	After	Change
Prompt volume	3500+ lines	1401 lines	2.5x compression
Files in core	7	12	+5 (new mechanisms)
Duplication	25–30%	$<$5%	Eliminated
Project-specific content	Mixed with core	Separated into dedicated directory	Separated
Glossary terms	30	44	+14 new
ODTOE formulas	5 (with errors)	8 (corrected)	P_{coll}: $0.61 \to 0.657$

The compression of 3500 lines to 1401 (factor of 2.5) was achieved without content loss through: (a) elimination of duplication between files, (b) separation of project-specific content from the core, (c) standardization of formulations through a unified glossary of 44 terms. This is an example of Muda elimination in TPS terminology [9]: removal of work that does not add value.

V. EXPERIMENT 2: A/B TESTING OF PROMPT LANGUAGE

V.1. Design

The same task ("Analyze the structure of the framework constitution and propose 3 improvements for increasing coherence S_{team}") was given to two groups of 5 agents:

- Group RU: role prompts in Russian (current files)
- Group EN: role prompts in English (translated)

The framework core (constitution) was identical for both groups (in English). Thus, the only variable was the language of role prompts, ensuring experimental control.

V.2. Quantitative Results: B-Score by Role

Role	B (RU)	B (EN)	Δ	Λ (RU)	Λ (EN)
Visionary	0.325	0.41	+0.085	0.50	0.60
Analyst	0.344	0.509	+0.165	0.50	0.70
Builder	0.344	0.51	+0.166	0.50	0.70
Validator	0.325	0.59	+0.265	0.50	0.85
Coherencer	0.39	0.55	+0.16	0.60	0.85
Mean	**0.346**	**0.514**	**+0.168**	**0.52**	**0.74**

A characteristic pattern: in the RU group, all agents have $\Lambda = 0.50$ (uniform cold start), whereas in the EN group, Λ varies from 0.60 (Visionary) to 0.85 (Validator, Coherencer). English-language prompts allow agents to more precisely identify and apply relevant patterns from the loaded framework — they rate their experience higher because they genuinely extract more applicable knowledge from the context.

V.3. Team Metrics

Metric	RU Group	EN Group
Mean B	0.346	**0.514** (+48%)
Agents with $B > 0.5$	0 out of 5	4 out of 5
S_{team}	**0.970**	0.920
$S_{\text{adjusted}} = S \times \bar{B}$	0.335	**0.473**
Format adherence	5/5	5/5

V.4. Qualitative Comparison

Criterion	RU Group	EN Group
Originality	φ-weighted S_{team} for toroidal topology, Graduated Activation, Phase-Adaptive Weights	S_{adjusted}, Role Interface Contracts, Agent Loading Protocol
Practicality	Formula-based proposals, less specific in implementation	Responsibility distribution table, precise contract formats, specific implementation lines

Diversity	5 agents converged on 2–3 ideas (high convergence, low diversification)	5 agents covered different aspects (low convergence, high diversification)
Bug detection	Found: weight w_i indeterminacy, absence of B exchange protocol	Found the same + broken links, glossary duplication, language non-uniformity
Blind spots	Did not notice the language problem (inside RU context)	Did not propose φ-weighted S_{team} or mathematical innovations

Key finding: the RU group is unable to see the language problem because they are inside the Russian-language context (observer blind spot). The EN group is unable to produce the φ-weighted S_{team} — a deep theoretical innovation requiring a different type of abstraction. This confirms that prompt language acts as an observation operator $\hat{O}_{\text{lang}}$, configuring the accessible cognitive space.

V.5. Coherence Paradox: S_{team} vs S_{adjusted}

A paradox was discovered: the RU group is more coherent ($S_{\text{team}} = 0.970$) but at a low B level. The EN group is less uniform ($S_{\text{team}} = 0.920$) but at a high B level. The gap between S_{team} and S_{adjusted} for the RU group is $0.970 - 0.335 = 0.635$ — this is an indicator of phantom coherence.

Why S_{team} is dangerous as a sole metric: five agents with identically low $B = 0.2$ yield $S_{\text{team}} = 1.0$ (perfect agreement), but $S_{\text{adjusted}} = 0.2$ (effectively a non-functional team). The metric S_{team} reports: "the team agrees," but does not answer the question "agrees on what?"

V.6. Interpretation: Language as an Observation Operator

Prompt language is not a neutral instruction carrier — it configures the agent's cognitive space, acting as the observation operator $\hat{O}_{\text{lang}}$:

Task Type	**RU Stack ($\hat{O}_{\text{RU}}$)**	**EN Stack ($\hat{O}_{\text{EN}}$)**
Theoretical development	**Better**: depth of abstraction, mathematical innovations	Worse: more formal, less creative
Practical refinement	Worse: less concrete, fewer actionable items	**Better**: more concrete, more bugs found
Problem detection	Blind spot for the language problem	**Better**: sees non-uniformity, broken links

Mathematical formulas	**Better**: for torus, activation	φ-weights graduated	Simpler: $S_{\text{adjusted}} = S \times \bar{B}$

VI. EXPERIMENT 3: ENTRY POINT ARCHITECTURE

VI.1. Design

Two groups of 5 agents each solved the same mixed task (theory + practice):

- Group A (router): first reads a separate routing file (50 lines, routing table), then loads the appropriate stack
- Group B (constitution): first reads the framework constitution (Section Zero contains routing), determines the stack independently

VI.2. Quantitative Results with Stack Selection

Role	**B (router)**	**B (inline)**	Δ	**Selected Stack**
Visionary	0.52	0.55	+0.03	A: RU; B: RU
Analyst	0.58	**0.65**	+0.07	A: RU; B: EN
Builder	**0.65**	0.454	-0.196	A: EN; B: EN
Validator	0.459	0.58	+0.121	A: RU; B: EN
Coherencer	0.454	0.58	+0.126	A: RU; B: EN
Mean	**0.533**	**0.563**	**+0.03**	

VI.3. Critical Observation: Team Split

The routing file split the team: the Builder classified the task as practical and went to the EN stack, while the remaining 4 agents went to the RU stack. Within a single Round Table, the team was working in different language stacks, which according to Experiment 2 results introduces $\Delta S \approx 0.05$–0.10 of additional decoherence.

Group B (constitution, Section Zero): all agents independently determined the stack without formal separation. 4 out of 5 agents chose the EN stack, the Visionary remained on RU. Result: $\bar{B} = 0.563$ (higher than the router group), 4 out of 5 agents with $B > 0.5$.

Team metrics:

Metric	**Router Group (A)**	**Constitution Group (B)**
$\bar{B}$	0.533	**0.563**

S_{team}	0.882	**0.912**
S_{adjusted}	0.470	**0.513**
Agents with $B > 0.5$	3 out of 5	4 out of 5
Stack divergence	Yes (1 EN + 4 RU)	Minimal (4 EN + 1 RU)

Conclusion: a separate routing file is unnecessary. Routing embedded in the framework core (Section Zero of the constitution) produces a better result without additional overhead ($\sim$50 lines of context) and, critically, does not split the team across different stacks.

VII. DETECTION OF PHANTOM COHERENCE AND S_{adjusted}

VII.1. The S_{team} Paradox

During Experiment 2, a paradox was discovered: the RU group has $S_{\text{team}} = 0.970$ (near-perfect agreement) but $\bar{B} = 0.346$ (low quality), while the EN group has $S_{\text{team}} = 0.920$ (slightly lower) and $\bar{B} = 0.514$ (significantly higher). Without the S_{adjusted} formula, the RU group appears "healthier" (0.970 $>$ 0.920). With S_{adjusted}, the picture is inverted: the EN group (0.473) surpasses the RU group (0.335).

VII.2. Independent Confirmation Through Glossary Conflict

All 10 out of 10 agents in Experiment 3 independently discovered a conflict in the glossary: two different definitions of S_{adjusted} existed. This confirms: the formula $S_{\text{adjusted}} = S_{\text{team}} \times \bar{B}$ is necessary as a complement to S_{team} for correct diagnostics.

VII.3. Three Diagnostic Scenarios

Scen.	Condition	Diagnosis	Action
1	$S_{\text{adjusted}} > 0.5$	Healthy state	Continue working
2	$S_{\text{team}} > 0.7$, $S_{\text{adjusted}} < 0.5$	Phantom coherence	Activate premise revision mechanism [1]
3	$S_{\text{adjusted}} > 0.7$, $S_{\text{team}} < 0.7$	Individually strong, misaligned	RT synchronization

The operational thresholds of 0.5 and 0.7 are chosen for interpretive convenience — a constructive choice, not a consequence of ODTOE axioms.

VII.4. Numerical Example

Consider a team of 5 agents with $B_i = \{0.2;\ 0.2;\ 0.2;\ 0.2;\ 0.2\}$:

- $S_{\text{team}} = 1 - \frac{2}{5 \cdot 4} \sum |B_i - B_j| = 1 - 0 = 1.0$ (perfect agreement)
- $\bar{B} = 0.2$
- $S_{\text{adjusted}} = 1.0 \times 0.2 = 0.2$ (critically low — the team is synchronized at a failure level)

Compare with a team $B_i = \{0.9;\ 0.7;\ 0.8;\ 0.6;\ 0.85\}$:

- $S_{\text{team}} = 1 - \frac{2}{20}(0.2 + 0.1 + 0.3 + 0.05 + 0.1 + 0.1 + 0.15 + 0.2 + 0.25 + 0.1) = 1 - 0.155 = 0.845$
- $\bar{B} = 0.77$
- $S_{\text{adjusted}} = 0.845 \times 0.77 = 0.651$ (healthy state)

By S_{team}, the first team looks significantly better (1.0 $>$ 0.845). By S_{adjusted}, the second team vastly surpasses the first (0.651 $>$ 0.200).

VIII. REAL DEPLOYMENT SESSION ANALYSIS: THE LAMBDA PROBLEM

VIII.1. Session Statistics

Analysis of a real deployment session for a production project (Session D) revealed a fundamental problem in framework application:

Metric	Value	Expectation
Direct tool calls (Bash+Write+Edit)	157	$<$30 (with delegation)
Agent tool calls	13 (8%)	$>$50 (XL task)
Completed Round Tables	0	$\geq$3 (XL classification)
Delegation ratio	$\sim$5%	$>$80%
Actual mode	Solo Builder	META + 5 roles

The agent received a minimal-length bootstrap prompt, loaded the entire framework (400+ lines), correctly classified the task as XL (requiring 10+ agents and 3 Round Table cycles), and then completely ignored its own classification and worked as a solo Builder.

VIII.2. Collapse Chronology

Session chronology:

1. The agent loaded the framework core and meta-protocol (full stack)
2. Correctly classified the task as XL
3. Immediately began writing code directly (Bash, Write, Edit)
4. Delegated 13 sub-agent calls, each of which was a single Builder request rather than a structured Round Table
5. Did not create a single RT cycle throughout the entire session

The session was classified as XL — and immediately collapsed into Solo Builder. This is a manifestation of an attractor: LLMs by default gravitate toward the "helpful single assistant" mode, and without explicit enforcement, this mode dominates over any loaded framework.

VIII.3. Root Cause Analysis: Four Fatal Defects

Root cause analysis (using the 5 Whys method) identified four fatal defects in the bootstrap prompt:

1. **Absence of identity declaration:** the prompt did not contain the phrase "You are the META-Orchestrator, you NEVER write code." Without an explicit identity, the agent assumes its default role — universal assistant.
2. **Absence of delegation mandate:** the brief instruction "proceed" was interpreted as "do it yourself" rather than "delegate via Agent tool." The delegation imperative was absent.
3. **Absence of invariant check:** there was no cycle "BEFORE EACH action, verify: is this action a delegation or not? If not — STOP."
4. **The "read and proceed" format:** three words creating no cognitive friction. The agent read — and proceeded to do what it does best (write code).

VIII.4. The Lambda Problem in ODTOE Formalism

This confirms the Lambda problem formalized in ODTOE [1]: $\Lambda = 0$ means "knowledge exists but is not applied" — an effect analogous to the "karaoke effect" in the SKW matrix [7]. The framework described all processes, the agent read all descriptions, but between knowledge and action there was no enforcement mechanism — a classical strange loop of self-observation [10].

In terms of formula (II.1): the agent had $F = 0.9$ (read everything), $E = 0.3$ (did the wrong thing), $(1 - \sigma) = 0.4$ (the result contradicted the process), $\Lambda = 0.05$ (experience

was formally present but not applied). Total: $B = 0.9^{0.25} \cdot 0.3^{0.25} \cdot 0.4^{0.25} \cdot 0.05^{0.25} \approx 0.97 \cdot 0.74 \cdot 0.80 \cdot 0.47 \approx 0.27$ — an extremely low result for an agent that had access to all information.

VIII.5. Solution: Three-Level Enforcement System

The solution: a three-level enforcement system, structurally analogous to the distributed management model [6]:

Level	Mechanism	Content	Property
1	Auto-loaded file	Identity declaration + delegation mandate + invariant check	Auto-loaded before the first message
2	Meta-protocol section	7-point session checklist	Output by orchestrator as first message
3	Lesson in memory	Record fixing a specific failure mode	Loaded during Λ-initialization

The key architectural innovation is Level 1 (auto-loaded file): the file is loaded by the platform automatically, before the user's first message. The agent cannot skip, ignore, or "forget" it. This is the only level with guaranteed execution — consequently, it must contain the most critical invariants.

VIII.6. Bootstrap Protocol

Based on the deployment session analysis, the Bootstrap Protocol was developed — a 7-point checklist:

1. Declare identity: "I am the META-Orchestrator"
2. Load the framework core and meta-protocol
3. Classify the task (S/M/L/XL)
4. Determine the language stack (Section Zero)
5. Plan RT cycles and distribute roles
6. Output the classification to the user as the first message
7. Proceed to RT-1 via Agent tool (not direct execution)

IX. SYNCHRONIZATION AUDIT AND BILINGUAL ARCHITECTURE

IX.1. The Synchronization Problem

Creating bilingual file pairs (X + X_EN) immediately created a synchronization problem. The audit (Validator, full pairwise comparison) discovered 16 desynchronizations:

#	File(s)	Problem	Severity
1	Constitution (RU)	Outdated version number	HIGH
2	Constitution (RU)	Missing Section Zero (Language Policy) — entire section	HIGH
3	Constitution (RU)	Missing S_{adjusted} formula and 3 diagnostic bands	HIGH
4	Constitution (RU)	Missing Graduated Activation (GREEN/YELLOW/RED)	HIGH
5	Constitution (RU)	Missing bilingual strategy paragraph	MEDIUM
6	Constitution (RU)	Missing default weights ($w_1 = w_2 = w_3 = w_4 = 0.25$)	MEDIUM
7	Glossary (EN)	Outdated version, missing 14 terms	HIGH
8	Glossary (RU)	Reference to non-existent path	MEDIUM
9	Glossary (EN)	Reference to non-existent path	MEDIUM
10	Documentation	Header indicates one version, footer another	HIGH
11	Documentation	File tree contains non-existent directory	MEDIUM
12	Documentation	File tree omits 10+ existing files	MEDIUM
13	Loading protocol	Outdated version number in footer	MEDIUM
14	Constitution (both)	Section references non-existent directories	MEDIUM
15	Glossary (EN)	Uses outdated term form	LOW
16	Meta-protocol (both)	Missing version header	LOW

Audit statistics: 8 file pairs checked, 6 fully synchronized, 1 partially desynchronized, 2 severely desynchronized. 5 HIGH-level issues, 7 MEDIUM-level, 4 LOW-level.

IX.2. Simultaneous Synchronization Rule

Based on the audit, a rule was formulated: when creating or updating a file that has a bilingual pair, both files are updated in a single operation. Creating one file "now" and synchronizing "later" is explicitly prohibited — "later" turns into "never," and the synchronization audit becomes a permanent maintenance burden.

IX.3. Four-Layer Bilingual Architecture

Based on the results of the linguistic analysis (Section III) and the synchronization audit, a four-layer language architecture model is proposed:

Layer	Content	Examples	Language
1	Invariant mathematics	$B = F^{w_1} \cdot E^{w_2} \cdot (1-\sigma)^{w_3} \cdot \Lambda^{w_4}$	Formulas (language-independent)
2	TPS terminology	Jidoka, Andon, Hansei, Yokoten, Round Table	Proper nouns (not translated)
3	Process terms	Kill-Switch, True North, Blast Radius, Spiral Gap	English (operational)
4	Project language	Task descriptions, comments, entry documentation	Team language (RU or EN)

Principle: English for breadth (practical tasks, bug detection, instrumental coverage), Russian for depth (theoretical development, mathematical innovations, conceptual density). Synthesis occurs at the Round Table, where both perspectives collide and produce a result that surpasses each individually.

X. CHECK-FIRST PIPELINE

X.1. Motivation: An Article with Incorrect Formatting

The research article (the present document) was initially generated with incorrect formatting: comment groups were missing from the preamble, undesirable separators were used between sections, and the keyword format did not match the template. A complete reformatting was required — waste (Muda) [9].

Root cause analysis: the format specification was loaded after generation rather than before. This is an inversion of the correct order: format data are input data, not an output filter.

X.2. Check-First Pipeline Architecture

Based on the analysis, the Check-First Pipeline methodology was developed — a mandatory pre-generation protocol consisting of 7 points:

1. **FORMAT:** load the format specification (full preamble + first section) — the golden standard of formatting
2. **FORMULAS:** list ALL formulas in the article. Recalculate each independently to 50 significant digits. DO NOT copy results from other articles.
3. **CONSTANTS:** prepare 50-digit values for π, φ, $(\pi - 3)^2$, and all derived constants
4. **SOURCES:** list ALL bibliography entries. Verify each: DOI, publisher, year, pages
5. **STRUCTURE:** define the section plan (Roman numerals). Verify the absence of overlaps with the existing corpus
6. **CONSISTENCY:** verify terminological conformity with the ODTOE glossary (44 terms). No contradictions with the corpus
7. **LANGUAGE:** confirm that RU and EN versions will be produced simultaneously

The critical idea: items 1–4 are performed BEFORE text generation (pre-generation data verification), items 5–7 during and after generation (structural and textual verification). Formula errors are input data errors, not output text errors; they must be intercepted before entering the text, not after. Pre-generation verification is prevention; post-generation checking is rework, costing 2–3 times more.

X.3. Role Distribution in Check-First

Role	Pre-Generation Tasks	Post-Generation Tasks
Validator	Checks 1–4 (data integrity)	Checks 5–7 (text quality)
Builder	None (receives a verified data package)	None (passes to Validator)
Analyst	Decomposition of formulas and dependencies	AI marker verification (Check 5)
Coherencer	Terminology verification (Check 6)	Internal consistency verification (Check 7)
Visionary	Strategic integrity (Check 4)	None

XI. EXTRACTED PRINCIPLES

The complete experimental series generated a set of invariant principles — regularities applicable to any multi-agent LLM system. The most significant for multi-agent coherence:

Cold Start Principle. With empty project memory, all agents are assigned $\Lambda = 0.5$ — not zero (this would zero out B by the multiplicative formula) but not a high value either. This is an honest prior: "I have no project memory, but I carry framework knowledge."

Adjusted Coherence Principle. $S_{\text{adjusted}} = S_{\text{team}} \times \bar{B}$. Never use S_{team} alone. Five agents can perfectly agree on the wrong answer ($S_{\text{team}} = 1.0$, $\bar{B} = 0.2$, $S_{\text{adjusted}} = 0.2$).

Phantom Coherence Detector. When $S_{\text{team}} > 0.8$ and $\bar{B} < 0.4$ — flag PHANTOM COHERENCE. Do not continue. Activate the premise revision mechanism [1].

Bilingual Routing Principle. EN for practical tasks, RU for theoretical depth. English for agent communication, interface contracts, debugging. Russian for theoretical deepening, mathematical derivations, conceptual exploration.

Enforcement Localization Principle. Reading the framework $\neq$ applying the framework. A bootstrap saying "read and proceed" leads to 95% solo Builder. Required: (a) identity declaration, (b) delegation mandate, (c) invariant check before each action.

Pre-Generation Verification Principle. For artifacts with defined quality specifications (articles with 7 checks, code with tests) — input data verification BEFORE generation. Text quality checks — AFTER.

Simultaneous Synchronization Principle. Bilingual files are created and updated simultaneously. Creating one file without its pair is technical debt. 16 desynchronizations were found in files created during the same session.

Adversarial Testing Principle. Before deploying a new bootstrap — run a test agent with a specific XL task and verify that it delegates rather than executes. Testing costs one agent call; not testing costs an entire session.

Format-as-Input-Data Principle. When generating any formatted artifact, the format specification is loaded as the first step of the activation operator A_F.

XII. MEGA-PATTERNS

Analysis of the extracted principles and four experiments revealed four mega-patterns — higher-order regularities:

Mega-pattern 1: Self-application — the ultimate test. When a framework is applied to its own improvement (Level 9: $\Psi^* = \Phi(\Psi^*)$ [1]), every weakness becomes visible. The Lambda problem, the synchronization problem, the formatting problem — all were discovered because the framework was used on itself. 25 agents rebuilding their own framework is an observation operator directed at itself.

Mega-pattern 2: A/B experiments — the highest-value action. Two A/B

experiments (RU vs EN, router vs inline) produced more applicable data than 20 analytical reports combined. When in doubt — experiment, do not argue.

Mega-pattern 3: Bilingual architecture — a feature, not a defect. EN for breadth, RU for depth. Running both groups in parallel with synthesis at a Round Table produces a result that surpasses any monolingual approach. Language is an observation operator, not a neutral carrier.

Mega-pattern 4: Enforcement must reside where it fires. The bootstrap checklist in the framework core (read by everyone) was theoretically visible but practically ignored. It was moved to the meta-protocol (read only by the orchestrator), where it actually fires. The auto-loaded file is the ultimate level of enforcement: the agent cannot skip it.

XIII. SESSION REFLECTION (HANSEI)

XIII.1. Session Scale

Parameter	Forecast	Actual	Δ
Task size	M (formatting)	XL (rebuild)	+3 categories
Number of agents	5–10	80+	$\times 8$
RT cycles	1–2	10+ (parallel)	$\times 5$
Scope	1 project	2 projects + meta-analysis	+2 projects
File mutations	10–20	40+	$\times 2$
Architectural changes	0 (maintenance)	3 (bilingual, bootstrap, Section 0)	Unforeseen

XIII.2. What Worked

A/B methodology became the highest-value action. The 25-agent rebuild confirmed the framework's self-applicability. Deployment session analysis converted a single failure into a systemic improvement (Yokoten [9]). The synchronization audit intercepted version drift before launching the bilingual architecture.

XIII.3. What Needs Improvement

Articles were generated BEFORE applying the 7 checks — an inversion of the correct order (fixed: Check-First Pipeline). Bilingual files were created without simultaneous synchronization — immediate technical debt (fixed: simultaneous synchronization principle). The session grew from M to XL without formal reclassification — the hierarchical reclassification mechanism was never activated for multi-team work. The bootstrap prompt for the deployment session was not adversarially tested.

XIII.4. Spiral Gap ($\sim 2\%$)

Unresolved tasks feeding the next iteration:

1. The synchronization audit found 16 issues — not all were fixed
2. An adversarial test of the auto-loaded bootstrap file was not conducted
3. The Check-First Pipeline is described but not coded as a mandatory step in the configuration
4. The session reclassification mechanism has no enforcement hook
5. A constant registry to prevent propagation of the $P_{\text{coll}} = 0.61$ error was not created
6. The quantitative bridge between ΔS from language non-uniformity (0.05–0.10) and ΔB from EN prompts (+48%) has not been formalized
7. 80+ session agents were not measured by B-diagnostics — the framework claims to work but has no measurements for rigorous proof

Overall residual: 7 items $\approx 2\%$ of session volume, which is consistent with the spiral gap prediction $(\pi - 3)^2 \approx 0.020048479550599188058630700199133830130683010990152$.

XIV. DISCUSSION

XIV.1. Language as a Cognitive Space Configurator

The results of Experiment 2 demonstrate that prompt language choice is not a technical detail — it is the selection of an observation operator $\hat{O}_{\text{lang}}$ that determines which configurations the agent is capable of actualizing. Russian-language prompts activate the abstract-theoretical mode of LLM operation (depth $>$ breadth), while English-language prompts activate the practical-operational mode (breadth $>$ depth). The optimal strategy is bilingual: launching both groups in parallel with synthesis at a Round Table.

In the ODTOE context [1], this means: the same LLM with different language prompts constitutes different observers ($\hat{O}_{\text{RU}} \neq \hat{O}_{\text{EN}}$), projecting the same task (Ψ) into different configurations ($R_{\text{RU}} \neq R_{\text{EN}}$). Language is not a transmission channel but an observation lens.

XIV.2. Self-Organization Through the Spiral Gap

Each of the four experiments revealed phenomena not anticipated by the original design: the coherence paradox (Experiment 2), the team split by the router

[2] 50 significant digits.

(Experiment 3), the Lambda problem (deployment session analysis), and version drift (synchronization audit). This is a manifestation of the spiral gap $(\pi - 3)^2 = 0.020048479550599188058630700199133830130683010990153$ — the system does not close perfectly, and the residual feeds the next evolutionary iteration.

The session progressed from M to XL, each completed cycle producing a residual ($\sim$2%) that became the focus of the next cycle. Article formatting revealed LaTeX errors; errors required conversion tools; tools required quality standards; standards required a rebuild of the framework governing the standards. This is the spiral in action.

XIV.3. Phantom Coherence as a Systemic Risk

Phantom coherence (high S_{team} with low $\bar{B}$) represents the most dangerous configuration of a multi-agent system because it is subjectively perceived as productivity. All agents agree, results appear quickly, there are no conflicts. But the team is synchronized around an erroneous model. The S_{adjusted} formula is a necessary diagnostic tool that converts a qualitative suspicion into a quantitative indicator.

XIV.4. The Lambda Problem as a General Regularity

The Lambda problem is not specific to a particular implementation — it is a general regularity of systems in which knowledge and action are not linked by an enforcement mechanism. In education, this is the "karaoke effect" [7]: a student knows the answer but cannot apply the knowledge in a new context. In organizations, this is "knowledge that has not become practice": regulations are written, employees are trained, but behavior has not changed.

In the ODTOE formalism [1], Λ is not merely the presence of experience but its applicability. An agent with $\Lambda = 0$ is formally dead (multiplicative structure). The solution is not increasing the volume of knowledge but creating a mechanism for its automatic application (the auto-loaded file as an example of enforcement with guaranteed execution).

XIV.5. Limitations

1. B-score is based on agent self-assessment — systematic inflation or deflation is possible. An independent external quality metric was not used.
2. All experiments were conducted on a single LLM platform — results may differ for other models and platforms.
3. Sample size (5–10 agents per experiment) is insufficient for strict statistical significance; large-scale replication is required.

[3] 50 significant digits.

4. The Lambda component is constrained by the absence of project memory in experiments (cold start, $\Lambda = 0.5$ per the Cold Start Principle).

5. Experiments were conducted within a single session — order effects are possible (Experiment 3 may have been influenced by the results of Experiment 2).

6. Operational thresholds (0.5 and 0.7 for phantom coherence scenarios) are a constructive choice, not a deduction from ODTOE axioms.

XV. CONCLUSION

1. **The five-role structure** formalizes distinct observation operators $\hat{O}_r$, and their parallel operation via the Round Table ensures collective coherence exceeding the individual level. 25 agents (5 RTs $\times$ 5 roles) compressed the framework by 2.5x and discovered a P_{coll} error ($0.61 \to 0.657$) missed by 20 analytical agents.

2. **Prompt language is an observation operator:** EN prompts yield $\bar{B}$ 48% higher for practical tasks ($S^{\text{EN}}_{\text{adjusted}} = 0.473$ vs $S^{\text{RU}}_{\text{adjusted}} = 0.335$); RU prompts provide superiority in theoretical depth (φ-weighted S_{team}, Graduated Activation). A bilingual architecture is optimal.

3. **The formula** $S_{\textbf{adjusted}} = S_{\textbf{team}} \times \bar{B}$ detects phantom coherence: the RU group with $S_{\text{team}} = 0.970$ appears healthier than the EN group ($S_{\text{team}} = 0.920$), but S_{adjusted} inverts the picture ($0.335 < 0.473$). All 10/10 agents independently confirmed the necessity of this metric.

4. **Routing within the core** is more effective than a separate routing file: $S^{\text{inline}}_{\text{adjusted}} = 0.513$ vs $S^{\text{router}}_{\text{adjusted}} = 0.470$. The router split the team (1 EN + 4 RU); inline routing allowed self-determination (4 EN + 1 RU).

5. **The Lambda problem** (knowledge without application) led to 95% solo Builder at XL classification. Solution: three-level enforcement (auto-loaded file + meta-protocol + memory). Key principle: enforcement must reside where it fires, not where it is convenient to describe.

6. **Linguistic analysis of 12 files** revealed systemic non-uniformity (Mura): the core (547 lines) in EN, the operational layer (861 lines) in RU. Cyrillic token inefficiency (1.5–2.5$\times$), benchmark gap (5–15% MMLU/MGSM/XCOPA), strength of EN imperative constructions.

7. **The Check-First Pipeline** prevents 100% of formatting errors: the format specification is loaded as the first step of A_F, formulas are recalculated independently before text generation. Pre-generation verification is prevention; post-generation checking is rework.

8. **A set of extracted principles** and **4 mega-patterns** were derived from a session with 80+ agents. Spiral gap: 7 unresolved items ($\sim$2%), consistent with $(\pi - 3)^2 \approx 0.02$, feed the next iteration.

REFERENCES

1. Pankratov A. S. Observer-Dependent Theory of Everything (ODTOE): Axiom, Postulates, and Mathematical Formalism // Preprint. — 2025.
2. Arnold V. I. Mathematical Methods of Classical Mechanics. — New York: Springer-Verlag, 1978. — 462 p.
3. Pankratov A. S. Coherence: From Individual to Collective // Preprint. — 2025.
4. Pankratov A. S. Coherent Education: Theory and Methodology of Building Learning Systems Based on ODTOE // Preprint. — 2026.
5. Pankratov A. S. Coherent Education II: Entropy Diagnostics, Adaptive Personalization, and Evolutionary Selection // Preprint. — 2026.
6. Pankratov A. S. Transfer of Distributed Systems Management Methods to the ODTOE Coherent Fusion Reactor Project: Structural Analysis of Parallels Between Irrigation and Plasma Engineering // Preprint. — 2026.
7. Kibalnikov S. V. SKW Matrix — the “Karaoke Effect” in Education and High-Tech Manufacturing // Online resource. URL: `http://kibalnikov.com/wordpress/?p=57`
8. Pankratov A. S. Toroidal Topology of Reality: π-Rotation, φ-Jumps, and Nested Tori // Preprint. — 2026.
9. Ohno T. Toyota Production System: Beyond Large-Scale Production. — Portland: Productivity Press, 1988.
10. Hofstadter D. R. I Am a Strange Loop. — New York: Basic Books, 2007. — 412 p.