VeHa: A Hybrid National Verification Hackathon for Better Formal Methods Education

Sergey Staroletov[1,2], Dmitry Kondratyev[3], Vladimir Shelekhov[3,4], **Alexander Kogtenkov**[5], Nikolay V. Shilov[6], Natalia Garanina[3], Irina Shoshmina[7], Timofey Cherganov[8], Vasil Dyadov[5]

serg_soft@mail.ru kwaxer@mail.ru

[1]Institute of Automation and Electrometry, Novosibirsk, Russia
[2]Polzunov Altai State Technical University, Barnaul, Russia
[3]A.P. Ershov Institute of Informatics Systems, Novosibirsk, Russia
[4]Novosibirsk State University, Novosibirsk, Russia
[5]Kaspersky Lab, Moscow, Russia
[6]Lyceum 22 "Hope of Siberia", Novosibirsk, Russia
[7]Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russia
[8]RusBITech-Astra LLC, Moscow, Russia

(VeHa Consortium)

FMTea Workshop – May 19, 2026 – Tokyo, Japan

The Gap in Formal Methods Education

- Formal verification is critical for reliable software, but teaching it is hard.
- High entry barrier: students struggle to move from theory to practice.
- Traditional courses (testing, electives) are insufficient for the broader community.
- Need for applied, project-based activities that motivate deep learning.

Solution: A hackathon-style competition – VeHa (VErification HAckathon).

Why a Hackathon?

- Hackathons are effective for intensive, team-based problem solving.
- Verification is "programming" in a DSL (specification + proof).
- Competition + tight deadlines + mentoring + training materials = motivating environment.
- Hybrid (in-person + online) format lowers geographic barriers.

"VeHa" also means "a milestone" in Russian (and in Common Slavic).

Evolution of the VeHa Series

Year	Location	Focus	Key outcome
2023	Innopolis (PSSV)	SPIN/Promela, C-lightVer	15 teams; model checking difficult
2024	Innopolis (PSSV)	Frama-C, Coq, TLA+, C-lightVer	More industrial tasks, mixed success
2025	Novosibirsk (Pottosin Contest)	6 tasks, academy–industry partnership	Strong participation, measurable growth

Industry partners: Kaspersky Lab, Astra Group (Astra Linux).

- Following the awards ceremony for the first hackathon, Astra Group employees who participated in it proposed collaboration for the following year.
- After participating in the second hackathon, Kaspersky Lab employees also proposed collaboration for the following year.

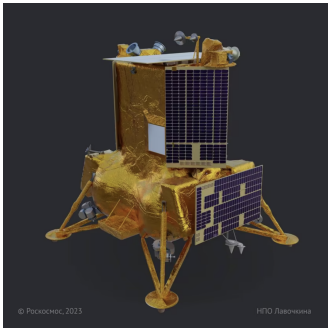
VeHa-2023 – The First Milestone

- **Location:** Innopolis (satellite event of PSSV workshop)
- **Format:** Hybrid (in-person + online)
- **Tracks:**
  - Deductive verification – C-lightVer (C-light language[1], uses ACL2 as a backend)
  - Model checking – SPIN / Promela
- **Tasks:**
  - Simple: deductive verification of array comparison function
  - Complex: modeling of Luna-25 space station accident (real-world incident)
  - Industrial protocols
- **Participation:** 15 teams, 13 submitted deductive solutions
- **Challenges identified:**
  - Model checking was too difficult – only one team solved Luna-25 task
  - Problems with LTL property formulation, concurrent modeling, counterexample analysis
- **Feedback:** Overall satisfaction but need for clearer problem statements

[1] Kondratyev, D.A., Nepomniaschy, V.A.: Automation of C program deductive verification without using loop invariants. Program. Comput. Softw. 48(5), 331–346 (2022).

VeHa-2023 – Leitmotif of the Competition

Model checking track: simulate the behavior of the space station control system leading to an accident based on official press releases.



The Roscosmos corporation published preliminary investigation results. They state that Luna-25 crashed into Earth's natural satellite due to a malfunction in the BIUS-L angular velocity measurement unit. It turns out that too many commands were received by the instrument, and the BIUS-L was unable to select the highest-priority ones. "Consequently, the onboard control system received zero signals from the BIUS-L accelerometers," Roscosmos explained. The lack of accelerometer data "prevented the corrective impulse from detecting the moment the required velocity was reached and from shutting down the spacecraft's propulsion system in a timely manner." Luna-25's engine burned for 127 seconds instead of the planned 84, and the station entered an unplanned, open orbit and collided with the lunar surface.

Source: `https://www.ixbt.com/news/2023/10/03/roskosmos-nazval-osnovnuju-prichinu-krushenija-luny25-k-avarii-privel-sboj-v-bloke-biusl.html`

VeHa-2024

- **Location:** Innopolis (again with PSSV workshop)
- **New tools introduced:** Frama-C, Coq, TLA+
- **Tasks (five total):**
  - Verification of access rights functions (Frama-C, Coq)
  - GPU computational pipeline – model checking with optimal parameter search
  - 2-SAT deductive verification (C-lightVer)
  - IBFT consensus protocol – modeling and verification
- **Outcomes:**
  - Increased registrations vs. 2023
  - More practical, industry-oriented problems
  - However: no submissions for the consensus protocol task
- **Winners:** 16 winners/laureates across categories
- **Feedback highlights:**
  - Submission system needs improvement (public GitHub PRs led to visibility of solutions)
  - Strong interest in industrial applications of formal methods

Related Work – Verification Competitions

- **VerifyThis** (since 2011). Focus on solving complex software verification problems, depth over speed.
- **RERS** series. Initially tool performance comparison, later synthetic benchmarks for education.
- **TOOLympics** (2019, 2023). Merger of VerifyThis, RERS, and others – closest to VeHa, but not annual.
- **SpecifyThis**. Bridge between specification paradigms, case-based learning.

Main difference: VeHa combines model checking (SPIN, TLA+) and deductive verification (Isabelle, Coq, Why3, C-lightVer) in a single annual hybrid hackathon.

Comparison with ICPC (former ACM ICPC)

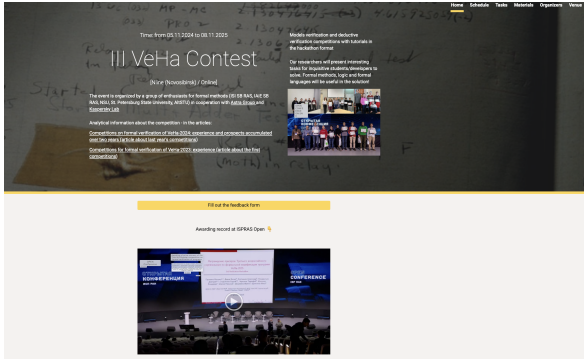
- ICPC[2] (popular in Russia): 5 hours, 8–12 problems, tested against hidden test suites.
- VeHa 2025: 3 days, six problems, verification via formal proofs, not black-box testing.
- We think that Integrating formal verification into popular programming contests (ICPC, Codeforces) is a pressing need to improve judging quality and educational level.

[2] https://icpc.global

Pedagogical Design

Core principles:

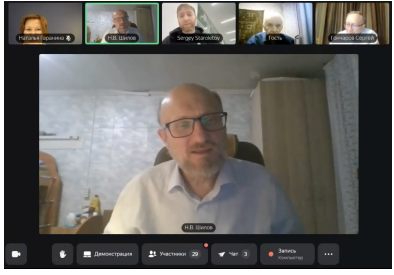
- Learning-by-doing
- Teamwork + mentorship
- Industry–academia bridge
- Two-track hybrid: model checking & deductive verification
- Online/offline hybrid.

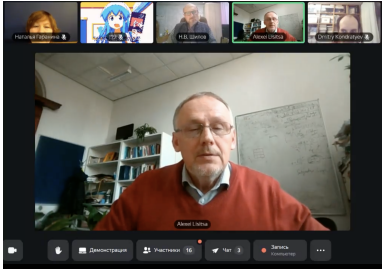


https://sites.google.com/view/veha2025

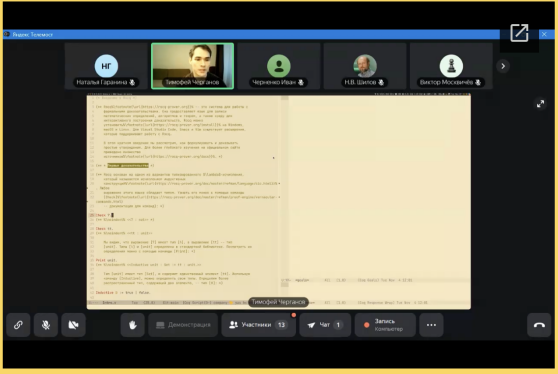
Preparatory Phase

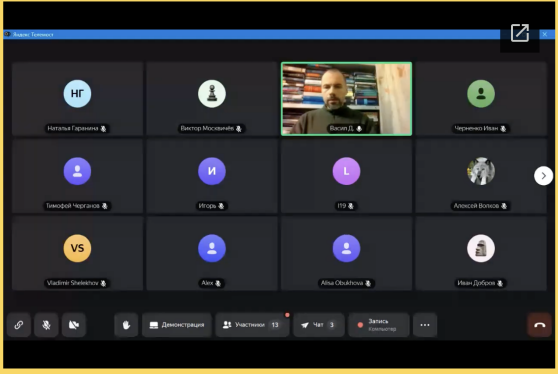
- Public website with mini-manuals, video tutorials for each tool.
- Online tutorials (Yandex.Telemost):
  - C-lightVer (deductive verification)
  - TLA+ (model checking of reactive/distributed systems)
  - Rocq (Coq) – from inductive types to program verification.
- Invited lecture by Alexei Lisitsa (Univ. Liverpool): "Finite Countermodel Finding for Infinite State and Parametrized Verification".





Tutoroals


Яндекс Телемост
НГ
Наталья Гарянина
Тимофей Черганов
Черненко Иван
Н.В. Шилов
Виктор Москвичёв
Тимофей Черганов
Демонстрация
Участники 13
Чат 1
Запись
Компьютер


Яндекс Телемост
НГ
Наталья Гарянина
Виктор Москвичёв
Васил Д.
Черненко Иван
Тимофей Черганов
И
Игорь
L
Алексей Волков
VS
Vladimir Shelekhov
Alex
Alisa Obukhova
Иван Дубров
Демонстрация
Участники 13
Чат 3
Запись
Компьютер

TLA+ Tutorial – Teaching Reactive Systems Verification

Topic: Formal verification of distributed protocols (network consensus).

Pedagogical techniques:

- **Conceptual layering.** Start with state variables, then transition predicates, finally temporal operators (always, eventually).
- **Interactive experimentation.** The presenter among with participatns modify specs and run TLC model checker live.
- **Counterexample analysis.** Show how to read error traces, localise bugs, and strengthen invariants.
- **Abstraction via nondeterminism.** Replace concrete values with sets – reduce state space.
- **Typing through invariants:** Use invariants to enforce type constraints (no separate type system).

Rocq (Coq) Tutorial – From Inductive Types to Program Verification

Topic: Interactive theorem proving for functional correctness.

Pedagogical sequence (gradual complexity):

1. **Basic definitions.** Inductive types (bool, nat), pattern-matching functions.
2. **Online feedback.** `Check` and `Compute` commands – learn by doing.
3. **Tactics introduction:** `simpl`, `rewrite`, `induction`, `apply` – prove simple properties.
4. **Abstraction.** Type classes and instances (monoids, polymorphic functions) – link to official docs and Software Foundations.
5. **Move to the task.** Define imperative language syntax, operational semantics, state; prove Hoare-like assertions.

Tools Covered

Model Checking	Deductive Verification
SPIN / Promela	Isabelle/HOL
TLA+	Coq (Rocq)
	Why3
	C-lightVer (ACL2 based)

Participants could choose one or more problems; solutions submitted via an ICPC-like testing system (NSU-TS). Manual checking by the juri.

Six Competition Tasks – Overview

1. **Model checking: Wenzhou train collision** (CTCS) – Promela, LTL.
2. **Distributed version control** – TLA+ specification.
3. **Functional lists in Isabelle/HOL** – folds, prefix operators.
4. **Static analysis in Rocq/Coq** – abstract interpretation.
5. **Chakravala algorithm (Pell's equation)** – C-lightVer, ACL2 invariants.
6. **Longest decimal substring** – deductive verification (Why3, Coq, Lean).

Task 1: Wenzhou Train Collision (Model Checking)

- **Tool:** SPIN / Promela
- **Type:** Research
- **Goal:** Model the CTCS high-speed train control system crash scenario.
- **Requirements:**
  - Formalize equipment fault tolerance, automatic blocking, dispatch–onboard interactions, time constraints.
  - Formulate LTL safety properties whose violation leads to collision.
  - Propose architectural changes to guarantee safety under multiple failures.
- **Reading:** Ouyang et al. (STAMP-based analysis)
- **Outcome:** Only one team submitted a partial solution (special diploma) + one good solution came after the contest.

Task 1: Wenzhou Train Collision (Model Checking)

- Just like in the first content, the task was to simulate an accident.
- This time, it was about Chinese high-speed trains, as there are plenty of natural language descriptions online (English and Chinese Wikipedia entries on collisions, train control system components, and automatic blocking).
- The goal was also to simulate the control system, show the state of the crash and propose a fix, verifiable by a verifier.
- However, many teams postponed the task due to more interest in industrial challenges and more clearly defined problems, and ultimately did not submit solutions.

Task 2: Distributed Version Control (Industrial, Kaspersky)

- **Tool:** TLA+
- **Type:** Industrial (proposed by Vasil Dyadov, Ilya Shchepetkov)
- **Goal:** Specify a distributed version control system with multiple agents editing files in parallel, synchronizing via a central server, and handling conflicts (Push, Pull, Merge).
- **Main aspects:**
  - Linear change log
  - Eventual consistency under diverging histories
  - Stepwise verification of invariants and absence of data loss using TLC model checker

Task 3: Functional Lists in Isabelle/HOL (Industrial, Kaspersky)

- **Tool:** Isabelle/HOL
- **Type:** Industrial (proposed by Alexander Kogtenkov)
- **Goal:** Prove properties of functional programs on recursive data structures.
- **Tasks:**
  - Prove lemmas about list folds
  - Equivalence of different prefix operator implementations
  - Properties of functions on lists of lists
- **Methodological constraint:** Use only basic tactics (`simp`, `blast`, `cases`, `induction`).

Task 4: Static Analysis in Rocq/Coq (Industrial, Astra Group)

- **Tool:** Rocq (Coq)
- **Type:** Industrial (proposed by Timofey Cherganov, Ivan Smirnov)
- **Goal:** Build and prove correctness of an abstract interpreter for a simple imperative language.
- **Requirements:**
  - Define abstract domains (flat constant lattice, interval analysis)
  - Formalize lattice structure, prove monotonicity and correctness of abstract operations
  - Extend analysis with branch and loop conditions
  - Prove preservation of safety properties (e.g., no division by zero)
- **Feedback:** Received three "5.0" ratings from participants.

Task 5: Chakravala Algorithm (Research)

- **Tool:** C-lightVer + ACL2
- **Type:** Research (proposed by Dmitry Kondratyev)
- **Goal:** Deductive verification of the historical Chakravala method for Pell's equation $x^2 - ny^2 = 1$.
- **Challenge:** Identify a loop invariant reflecting the arithmetic essence, using recurrent relations from Ayyangar's paper.
- **Automation:** ACL2 automatically applies domain theory lemmas to prove verification conditions.

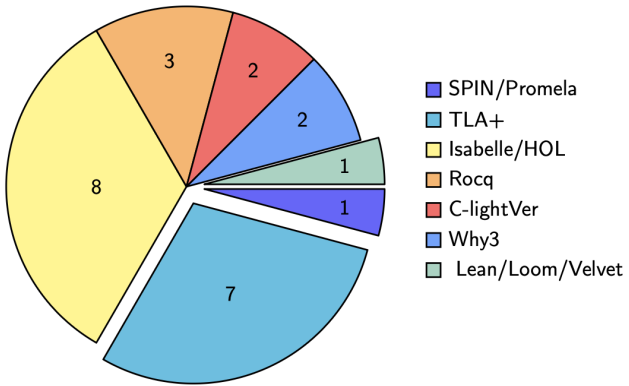
Task 5: Chakravala Algorithm (Research)

Task 6: Longest Decimal Substring (Research)

- **Tool:** Any (Why3, Coq, Lean, etc.)
- **Type:** Research (proposed by Vladimir Shelekhov)
- **Source:** 1980s Pascal programming textbook.
- **Goal:** Select longest substring of decimal digits that is closest to the left.
- **Requirements:**
  - Write efficient program
  - Build formal specification (pre/post, loop invariants)
  - Fully verify using chosen tool
  - Expert assessment of effectiveness (additional goal)
- **Observation:** List representation proved much harder than array representation; Why3 with arrays succeeded automatically.

Participation Metrics

- 35 teams registered (47 participants, 8 cities).
- 19 teams (30 participants) actively solved problems (4 cities).
- Solutions by tool (total 23 submitted solutions):



Competencies Acquired by the Participants

- **Train collision (SPIN)** – abstract domain modeling, LTL safety properties, fault injection.
- **Distributed VC (TLA+)** – compositional design, invariants, model checking with TLC.
- **Isabelle/HOL lists** – structural induction, equivalence proofs, modular reasoning.
- **Coq static analysis** – abstract interpretation lattice, concretization, monotonicity.
- **Chakravala** – loop invariants from recurrence relations, automated verification.
- **Longest substring** – formal specification, WhyML, automation vs. interactive proof.

Winners and Geographic Distribution

Awards by city:

- St. Petersburg, Russia: 14 awards (SPbPU, ITMO)
- Novosibirsk, Russia: 7 awards (NSU, SB RAS)
- Moscow, Russia: 5 awards (BMSTU, MIPT, ISP RAS)
- International: University of Naples (Cyprus) – 1st in task 6.


`https://youtu.be/zMIQGuXlYzI`

Participant Feedback (responses from 12 teams)

- 67% gave overall organisation the highest rating (5/5).
- 100%: task level "okay", wording "fairly adequate".
- 42% wanted a longer event.
- Preferred format: 67% hybrid, 33% fully online.
- Improvement requests: segmented time blocks, better submission system (closed repo), advance evaluation criteria.
- Task 4 (Rocq static analysis) received three 5/5 ratings.

Industry: motivation

Kaspersky Lab
KasperskyOS (microkernel OS)

Company:

- Attract people
- Demonstrate used tools
- Show FM use in the company
- Reuse materials in internal educational programs

Industry: motivation

Kaspersky Lab
KasperskyOS (microkernel OS)

Organizers:

- New tasks
- Tooling diversity
- Concise introduction
- Evaluation of results
- Prizes

Company:

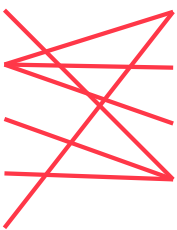
- Attract people
- Demonstrate used tools
- Show FM use in the company
- Reuse materials in internal educational programs

Industry: motivation

Kaspersky Lab
KasperskyOS (microkernel OS)

Organizers:

- New tasks
- Tooling diversity
- Concise introduction
- Evaluation of results
- Prizes



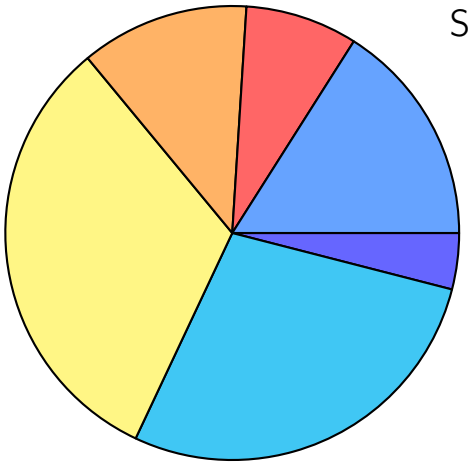
Company:

- Attract people
- Demonstrate used tools
- Show FM use in the company
- Reuse materials in internal educational programs

Formal methods in the Kaspersky Lab company

Goals	Methods	Tools
Strong functional correctness	Hoare logic	Frama-C + Why3 + Isabelle/HOL
Behavioral correctness	Model checking	TLA+/TLC/Pluscal
Security model guarantees	Refinement	Event-B
Language safety	Formal semantics, Run-time validation	Isabelle/HOL
...	...	...

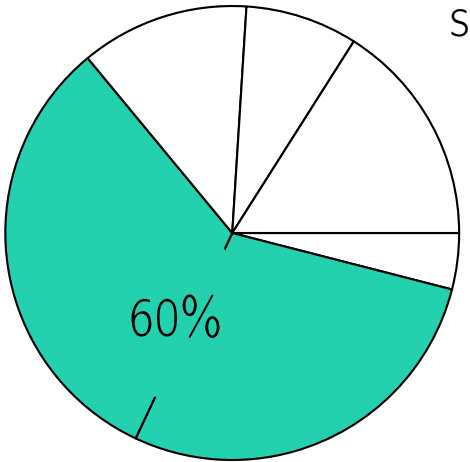
Success story



Submitted solutions:

- SPIN/Promela
- TLA+
- Isabelle/HOL
- Rocq
- C-lightVer
- Why3/Velvet

Success story



Submitted solutions:

- ☐ SPIN/Promela
- ☑ TLA+
- ☑ Isabelle/HOL
- ☐ Rocq
- ☐ C-lightVer
- ☐ Why3/Velvet

Success story

Factors:

- Detailed preparation procedure
- Self-contained introductory material
- Simplicity of tasks
- Gradual complexity of sub-tasks
- Transparent grades

Tasks characteristics

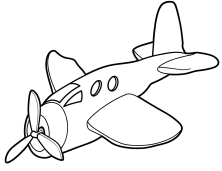
Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting	✓	✗
Easy	✗✓	✓
Creative	✓	✗
Automation-resistant	✓	✓✓

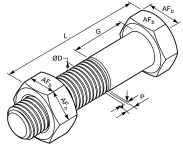
Tasks characteristics

Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting	✓	✗
Easy	✗✓	✓
Creative	✓	✗
Automation-resistant	✓	✓✓

Tasks characteristics

Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting	✓	✗





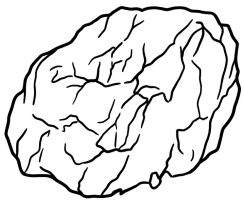
Tasks characteristics

Characteristic	TLA+	Isabelle/HOL
Real	✗	✓✓
Interesting		✗
Easy		✓
Creative		✗
Automation-resistant	✓	✓✓

Understanding

Tasks characteristics

Expectation:

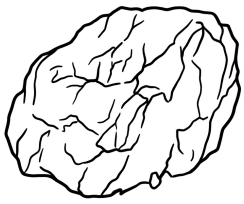


1ns

Automation-resistant	✓	✓✓

Tasks characteristics

Reality:



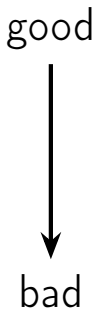
ages

Automation-resistant	✓	✓✓

Industry demands and contest alignment

Efficient navigation:

- Quick introduction to a new domain
- Core knowledge in **all** approaches
- Estimation: time, expertise, comp. resources
- Systematization



Trends and demands

Hybrid and heterogeneous proofs of correctness:

- Different methods in one project
- Integration of the tools
- Multi-level guarantees: tools, programs, data

Moving towards the end: Lessons Learned

- VeHa-2025 confirmed hackathon format as **effective for teaching formal methods**.
- **Combining model checking and deductive verification** in one event provides a comprehensive view.
- **Industry** involvement ensures relevance and **motivates** participants.
- The community is growing; **strong centers** in St. Petersburg, Novosibirsk, Moscow.

Next Steps

- **Balance** task difficulty – multi-level tasks work better.
- Improve submission **infrastructure** (closed repositories, better integration with website).
- Attract more industrial participants (**not only researchers**).
- Build a **benchmark** database from submitted solutions (for local verification tools).
- Strengthen **international** collaboration and increase outreach.

Acknowledgements

- Ivannikov Institute for System Programming (**ISP RAS**) (award ceremonies, support for publishind analytics papers).
- **V.A. Nepomniaschy**[3] Memorial Fund (for support to go there).

Thank you!

`https://github.com/VeHaContest` – Collection of VeHa Solutions

Questions?

[3] https://dblp.org/pid/n/VANepomniaschy.html