INVESTOR PRESENTATION DRAFT · JULY 2026

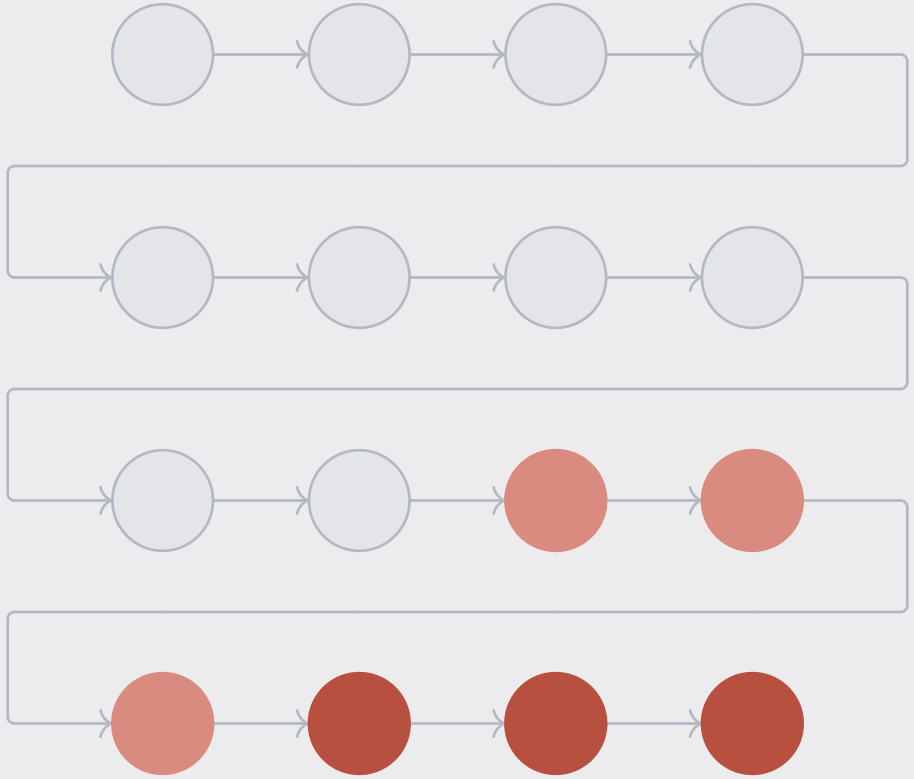
semiring.ai

Teaching AI to derive software instead of editing code.

RELIABILITY

AI proposes code that works. Most of the time.

The capabilities of coding agents are impressive. However, reliability has become the major limiting factor. Diligent human review is still required to identify subtle bugs, and errors compound in autonomous engineering loops.



THE GAP

Correct… compared to what?

SPEC.md

Natural language specifications are too vague to serve as a reliable reference for correctness.

?

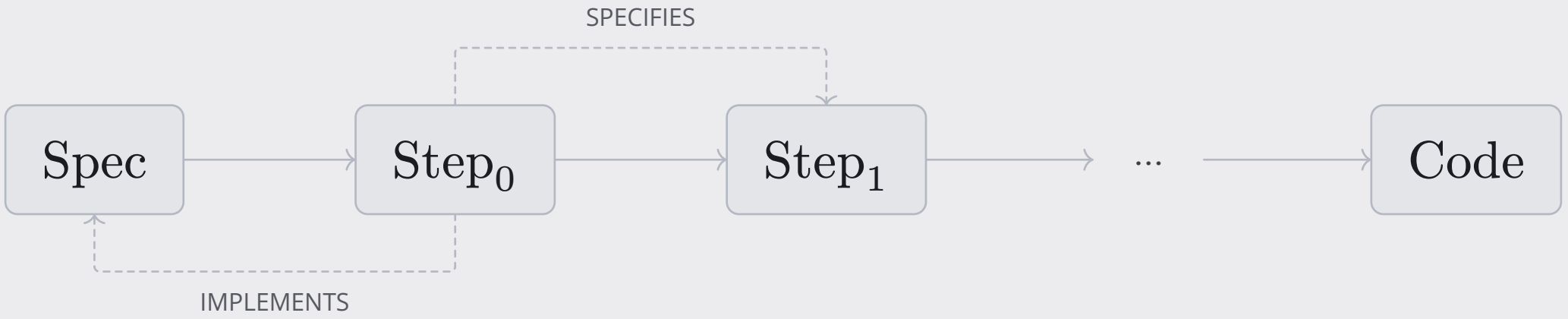
main.py

Code is the thing under suspicion, mixes intent with implementation details, and is too large and complex to review.

OUR SOLUTION

Semiring is the missing link

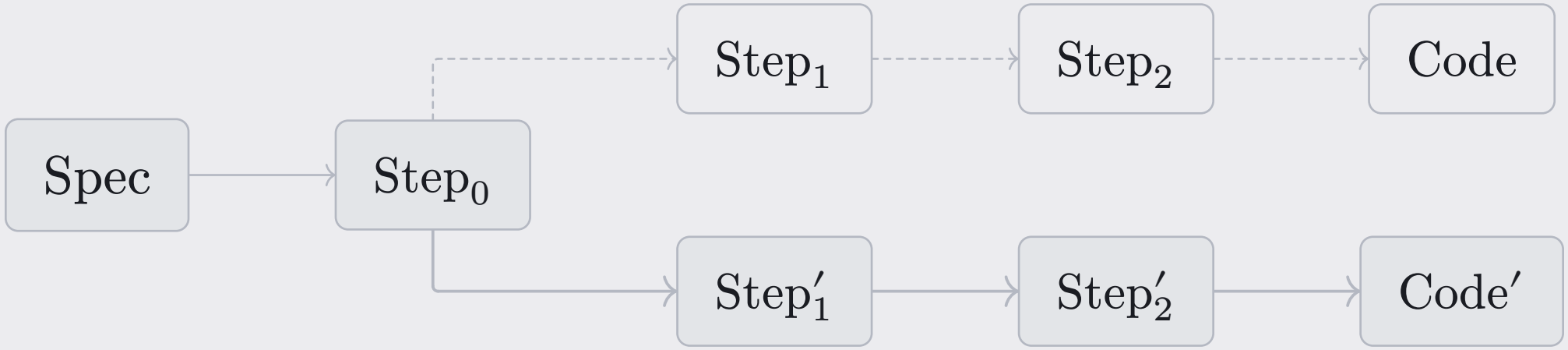
Engineering proceeds through a sequence of consequential choices. Today those choices are implicit, scattered across source code and human expertise, then largely discarded. Semiring IR is a structured representation of engineering decisions that sits between specification and executable code.



BEYOND CORRECTNESS

Rederive, don’t patch

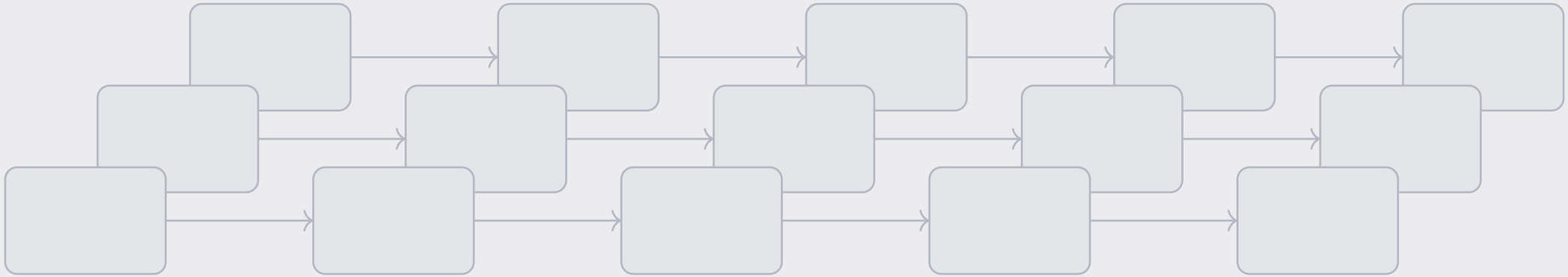
Conventional AI coding workflows iterate by patching code. Semiring can revisit algorithmic, numerical, structural, scheduling, and architectural choices at the right level of abstraction, then rederive the result. This is the approach driving innovations like FlashAttention, but this time without the human in the loop.



COMPOUNDING INSIGHT

Every derivation leaves a trace

We record the intermediate steps of every run, remembering what worked, under which assumptions, and why. This is training data for AI that does *engineering* and not merely coding.



ECOSYSTEM

One representation across computational domains

Computation is fragmented across specialised languages and compiler stacks. A shared representation allows transformations developed in one domain to become immediately applicable in another. Semiring IR is an open and extensible foundation for reusable engineering knowledge.

WHY NOW

The architecture came first. AI created a market.

Semiring grew out of years of research into compiler infrastructure for quantum computing. In solving that problem, we discovered a more general architecture spanning many forms of computation. At the time, there was little commercial demand for such a system. Autonomous software engineering created both the demand and an agent capable of using it.

BEACHHEAD

We target GPU kernels first

GPU compute has become the cost bottleneck in AI, making optimisation economically urgent. Promising model architectures are often delayed or never adopted because the path from idea to efficient kernels is too long and uncertain. Semiring can automate classes of kernel optimisation that would otherwise require years of research engineering.

- High deployment costs
- Clear measurable wins
- Large hidden optimisation space
- Rapid feedback and iteration
- Multiple abstraction levels

TEAM

The team this idea requires

Dr Lukas Heidemann

Researcher at University of Oxford
Former compiler researcher at Quantinuum
DPhil in CS at Oxford, MSc in Math at Bonn
Compilers, formal methods

Dr Cole Comfort

Researcher at Inria
DPhil in CS at Oxford
Categorical semantics

Marcus Gemzoe-Winding

Co-founded spher.app - led 5+ engineers.
Former Data Engineer at DRIVR
MSc in CS, Math at DTU
Reinforcement learning

MILESTONES

What we will demonstrate

By the end of this round, Semiring will demonstrate that AI can engineer software through a shared intermediate representation rather than manipulating source code directly.

- Semiring IR with extensible dialects and transformations
- Reusable transformations across independent domains
- AI-native compiler workflow built around Semiring
- Whole-system optimisation of GPU inference pipelines
- Initial external collaborators

THE ASK

We are raising $3M

To build the technical foundation for AI-assisted engineering through explicit derivation. Based across Europe’s leading programming languages and systems research groups, we can recruit exceptional researchers from our existing network while remaining highly capital efficient.

- Research engineering team: 70%
- Compute: 15%
- Operations: 15%

Appendix

ABSTRACTION LEVELS

Programs that cross the abstraction hierarchy

Semiring IR is a program representation that spans mathematical, statistical, data, and machine-level views of computation, in one extensible language.

Differential equations, Stochastic processes

Probabilistic programming, Differential programming

Logic programming, Incremental programming

Relational algebra, State machines

Real numbers, Tensor algebra, Streams

Sparse collections, MPI, MapReduce

Floating point, Async

Arrays, SIMD

LLVM, PTX, VHDL

DIFFERENT TRADEOFFS

An IR as a shared language for computation

Semiring builds on the central insight of MLIR: many different aspects of computation can be represented in a single extensible intermediate representation. MLIR's goal is to streamline compiler construction. Semiring's goal is to make the intermediate representation itself the shared language between AI systems, compilers, verifiers, and optimizers. This requires different design choices:

- implementation-independent specification
- declarative extensibility for dialects and transformations
- composable formal semantics
- graceful fallback for unknown extensions

MATHEMATICAL SEMANTICS

Designed for the unknown

We cannot anticipate tomorrow's computational abstractions. Rather than designing for today's domains, Semiring derives its architecture from mathematical semantics. This lets new domains compose coherently without redesigning the IR; an approach that emerged from our experience building quantum compiler infrastructure.