Petr Korobeinikov

Founding Engineer & Architect - Go, Kotlin, Postgres, Cloud, DBaaS, Kubernetes

Target role: Principal Engineer
Target stack: Kotlin, Kubernetes, Postgres, Kafka

LinkedIn: linkedin.com/in/petr-korobeinikov

PROFILE

Principal-grade engineer with experience designing and running resilient cloud services: DBaaS for Postgres and Redis, Managed RabbitMQ, voice/real-time systems on Go and gRPC, event-driven integrations on Kafka, business-process orchestration on Temporal. Run architecture reviews, set engineering standards and DevEx practices at the unit level, mentor engineers in the internal mentoring program. Adapt AI tooling to the team's development workflow. Run a public engineering blog principal-engineering.ru, YouTube and Telegram channels on engineering practices.

EXPERIENCE

MTS, FinTech — Founding Engineer & Architect

Oct 2024 - Jul 2026

Stack: Kotlin, Spring Boot, Go, GraphQL, gRPC, protobuf, Kafka, PostgreSQL, Kubernetes, Kustomize, OpenTelemetry, Terraform / OpenTofu, k6

In 2 months designed and built the backend of a section in the "My MTS" mobile app. Within the first six months we exceeded our most ambitious financial targets. Specific numbers under strict NDA.

- Designed the architecture of the GraphQL backend for BDUI (Backend-Driven UI).
- Adapted AI tooling to the development process: configured custom skills and subagents in the development AI agent, connected MCP servers to the team's internal systems, formulated AI usage guidelines and prompt templates for the domain - routine tasks (test generation, service scaffolding, first-pass code review) moved into the AI loop.
- Developed a service template: GraphQL, OpenAPI, code generation from specifications.
- Built the integration with the financial partner via Kafka - strict message ordering, protobuf, idempotent consumer, dead-letter queue.
- Adopted Spring Boot + Kotlin in several microservices: thanks to well-tuned components and Kotlin's conciseness, time to develop and debug new services dropped from 1-2 weeks to a few days.
- Set up observability on OpenTelemetry: distributed traces, structured logs, metrics, alerts.
- Provisioned infrastructure via Terraform / OpenTofu.
- Built a reproducible local developer environment on colima / lima.
- Prepared technical onboarding - new developer ramp-up time dropped 5x.
- Mentored the team on infrastructure (Terraform) and load testing (k6).
- Led 8 mentees in the internal mentoring program.

MTS, VoiceTech — Founding Engineer & Architect

Jan 2024 - Sep 2024

Stack: Go, gRPC, WebRTC, Opus (audio), Temporal, Kubernetes, Helm, Argo CD, OpenTelemetry, D2

Voice ecosystem product team: real-time phone-call processing, AI assistant inside the call, post-call processing of recordings.

- Designed and built the core of the voice ecosystem and the loop adapter: Go channels, gRPC streams, non-blocking channel work, `try-send` semantics for real-time processing.
- Launched the "Marvin" voice AI assistant inside an active phone call - the user addresses it by name and gets a voice response from the model; processing happens on the operator side (works from any phone, including push-button and grandma's rotary). Demonstrated to a wide audience at the `platforma.mts.ru` conference.
- Built browser-based developer tools for the voice model on Opus + WebRTC - new team members ramped up on skill development and debugging without being issued real handsets and corporate phone plans.
- Built phone-conversation summarization: recording processing, prompt engineering for the summarization model, Temporal for fault tolerance, Telegram bot as the user-communication channel.
- Set up observability on OpenTelemetry: distributed traces, structured logs, metrics, alerts.
- Introduced "diagrams as code" with `d2lang` (after evaluating PlantUML, C4 as code, Structurizr).
- Built a technical onboarding process - developer ramp-up time dropped 20x.
- Established a unified DoD (output / outcome).

- Launched structured Go-developer hiring, ran interviews per the established process.
- Consolidated team communication into a single channel instead of several chats.
- Set up a reproducible developer environment.
- Raised incident management and Security Championship at the unit level.
- A number of patches accepted upstream: `localmodule` in `gci` - included in `golangci-lint v1.58.0`; long-standing PR in `wreulicke/http-timeout`; x86 support in `asdf-eza`.
- Led two mentees in the internal mentoring program.

MTS, #CloudMTS Technical Lead
Aug 2021 - Jul 2023, Moscow
Stack: Go, Kubernetes, Helm, PostgreSQL, Redis, RabbitMQ, Ansible, Packer, OpenCensus
- Redesigned and shipped to production DBaaS for PostgreSQL (Managed Postgres) with a fault-tolerant configuration.
- Designed and built the control plane for launching new DBaaS services - shortened time-to-production for new products.
- On the same platform, built DBaaS for Redis (Managed Redis).
- On the same platform, built Managed RabbitMQ.
- Introduced tracing (OpenCensus), logging, and error reporting in team projects.
- Set up a reproducible developer environment: 20 minutes of onboarding and a person is ready to work.
- Set up two-stage VM provisioning: image builds via HashiCorp Packer and final VM configuration from the image via Ansible.
- Ran Architecture Reviews inside the team - influenced solution quality at the design stage.
- Promoted unified engineering approaches and architecture standards at the company level.
- Developed and rolled out Security Championship and Developer Experience practices within the team and the unit.

Avito Senior Software Development Engineer
Sep 2016 - Dec 2020, Moscow
Stack: Go, Kubernetes, Helm, GraphQL, Consul, Vault, Python, React, Tornado
- Researched moving functionality to a FaaS approach (kubeless, fission, knative, nuclio, openfaas). At the company level the decision was not to move in this direction.
- Took direct part in building the internal DBaaS: wrote a GraphQL API gateway over internal services (Consul, Atlas), developed a CLI for administrators and users. Recorded a screencast on working with GraphQL for colleagues. Took part in research on using PV in Kubernetes.
- Automated builds of database samples for the internal PaaS: working with Vault, meeting security requirements, dind for image builds, the matching service accounts and role bindings in Kubernetes. The personal-data masking mechanism received high marks from Roskomnadzor during an audit.
- Took part in improving the internal PaaS based on user feedback (React, Go, Python).
- Was on the Kubernetes administration on-call rotation - grew expertise.
- The "sagas" service - coordination of distributed transactions between microservices via the SAGA pattern. Took part in the full cycle - from deployment in Kubernetes to tests, code, and alerting. 100% test coverage (Tornado).
- Built a number of infrastructure microservices in Go with high test coverage - work with queues and the data bus, search mechanisms (the most critical one - delivery of listings into search).
- Built a new database-sampling tool with a YAML config under a strict schema, tests, and examples; correctness is verified by regular CI runs.

RDW-media Senior Software Development Engineer
Aug 2010 - Oct 2014, Moscow
- Launched the paid model for job seekers (May 2014, team result, owned the area personally).
- Laid a strong foundation for the "Professional Communities" project `pro.rabota.ru` (Symfony2).
- Introduced Composer for dependency management and stood up an internal mirror of the packages in use.
- Launched Rabota.ru iOS and Android apps from scratch, handed development off to a team.
- Reproducible developer environment via SaltStack and Vagrant; migrated part of the data from NFS to Riak; introduced RabbitMQ for heavy workloads.

PUBLIC PRESENCE

- Engineering practices blog: principal-engineering.ru
- YouTube channel: @principal-engineering
- Telegram channel: @principalengineering