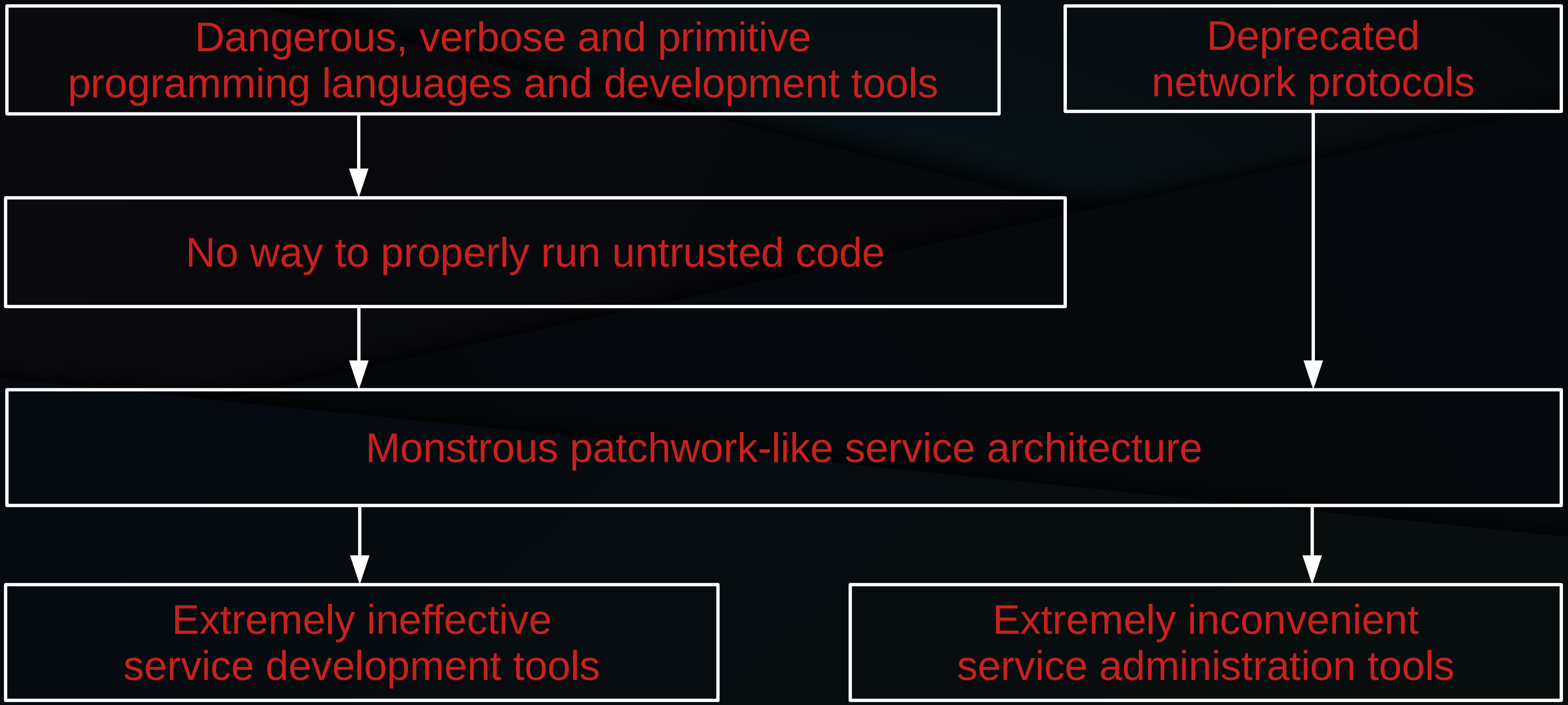
Grigoriy Okopnik
Evgeniy Okopnik

HardCode

general purpose programming language
service development and administration platform
IT content distribution platform

Current state



Dangerous, verbose and primitive programming languages and development tools

Deprecated network protocols

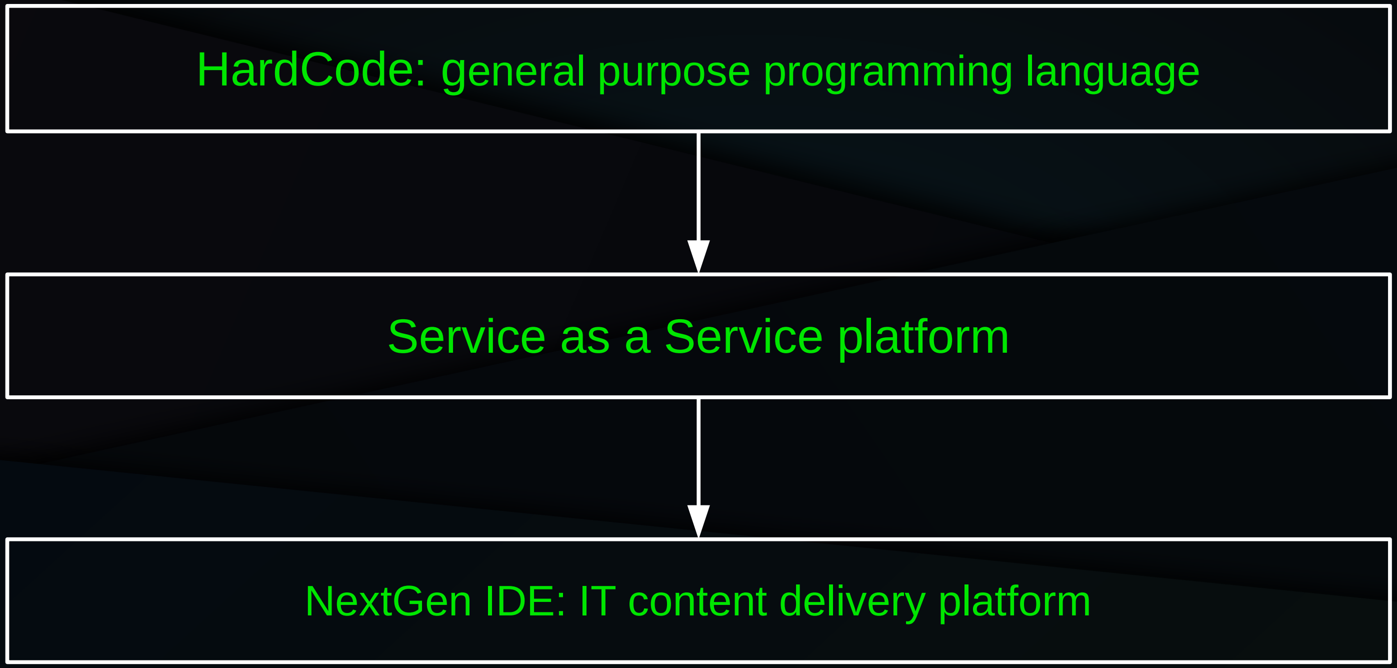
No way to properly run untrusted code

Monstrous patchwork-like service architecture

Extremely ineffective service development tools

Extremely inconvenient service administration tools

Project structure

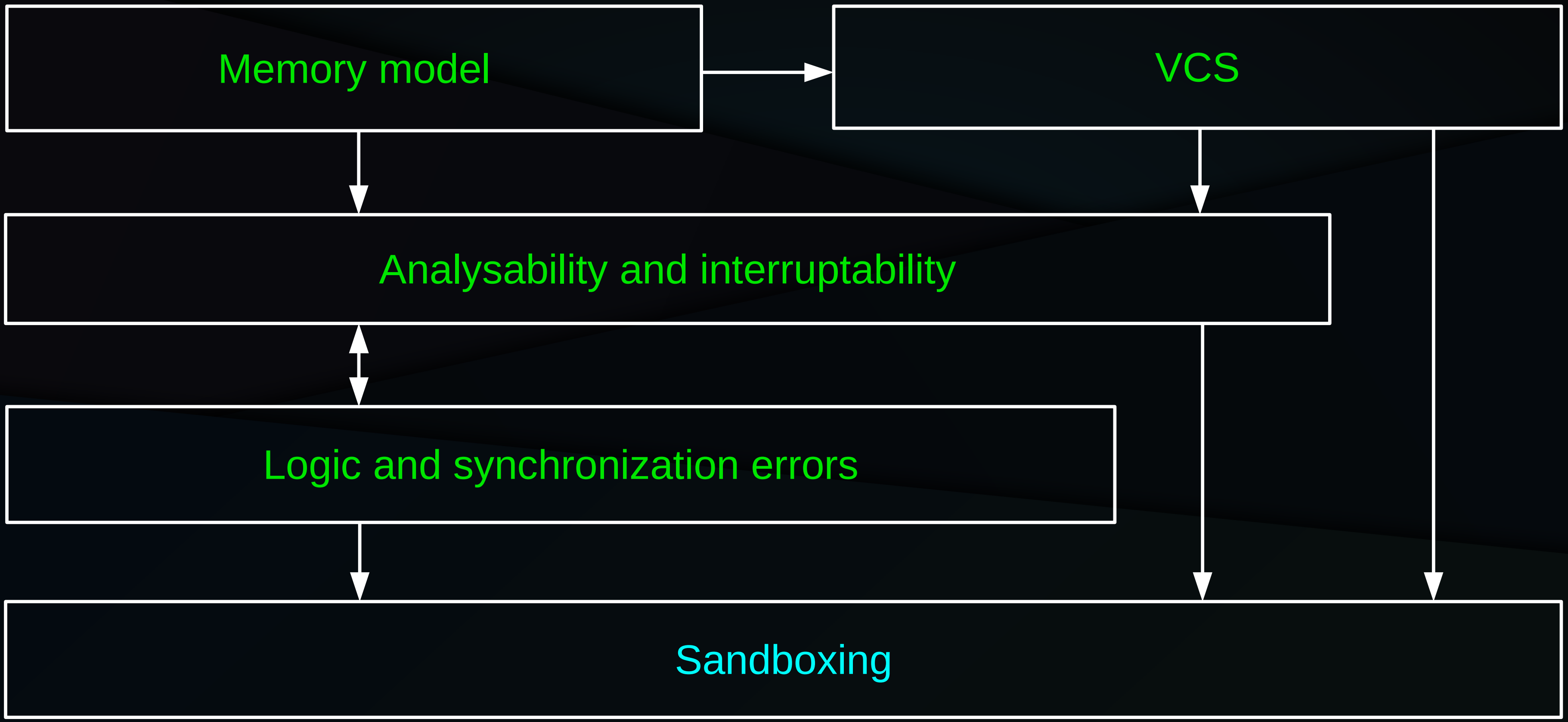


HardCode: general purpose programming language

Service as a Service platform

NextGen IDE: IT content delivery platform

HardCode programming language concept


Memory model
VCS
Analysability and interruptability
Logic and synchronization errors
Sandboxing

HardCode programming language concept

Memory model

- Unique hybrid memory model combining RAII with GC
- Unique non-blocking multithreaded Garbage Collector

↓

- Ultimate level of convenience security and isolation
- Performance comparable to performance of C
- Strict testability

HardCode programming language concept

Versioned code storage

- Source code is stored in a versioned DB specifically developed for HardCode
- Best VCS practices are integrated as first-class citizens
- Automated tests and documentation stored together with source code in a DB
- Test execution and profiling results are stored in a DB

⬇

- Version-managed development with Hot Reload – no program restart needed
- Consistent source management: refactoring, review, analysis, etc.
- Subject-based developer communication

HardCode programming language concept

Analysability

- «Out-of-the-box» analyzable complexity of whole program and it's fractures
- Interruptability and interruption latency is analyzable «out-of-the-box»
- Analytic profiling is available

⬇

- Very high source code readability
- Reach set of optimization features

HardCode programming language concept

Logic and synchronization errors

- No fatal run-time errors
- Most logical errors don't exist
- The rest of logical errors are fairly hard to code
- Expressive and reliable error handling
- No synchronization errors

⬇

- Programs don't crash (except due to system resource outage)
- Programs have far less logical errors
- Programs are much shorter, more expressive and readable
- Far less debug cases while debug itself is far more handy where needed

HardCode programming language concept

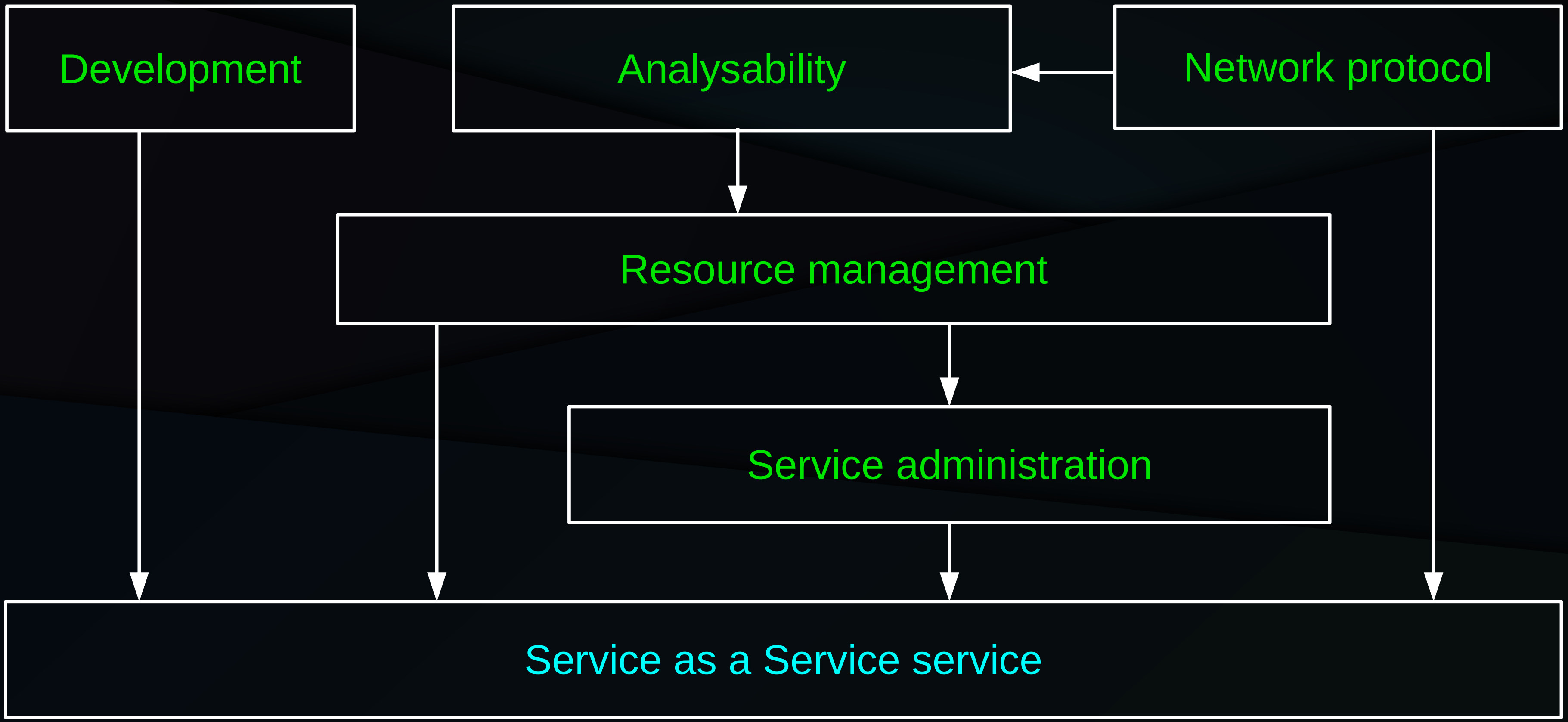
Sandboxing

- Actually working sandboxing (like in Lua but works)
  in statically typed language
  which is made possible by isolation, analysability and analyzable interruptability

↓

- Actually effective FaaS at back-end
- Actually adequate program-browsing design at front-end

Service as a Service platform



Development

Analysability

Network protocol

Resource management

Service administration

Service as a Service service

Service as a Service platform

Service development

- Today development looks like docker-compose:
  configuring whole service graph may take few weeks
- Service as a Service platforms allows to boot up entire service graph in 2 clicks
- The difference between monolith and microservices is opaque for a developer
- All the source code and resources are stored at the server-side,
  the only client-side application needed for development is IDE

⬇

- Effective development, flexible optimization
- Fast onboarding, flexibility and mobility for developer

Service as a Service platform

Network protocol

- Cluster architecture driven network protocol combining UDP + gRPC + TLS + dTLS advantages
- API schema is stored at versioned DB which strictly enforces compatibility

⬇

- Optimal latency
- Adequate encryption
- Convenient cluster-based data model at network interaction schema
- No SLA downgrade
- No explicit or hidden errors caused by broken compatibility

Service as a Service platform

Analysability

- Request serving complexity is analyzable
- Client-side program execution complexity is analyzable
- Network protocol makes whole request graph analyzable
- Whole big network-distributed program is analyzable
- Error stack is stored and addressed by unique identifier

⬇

- Network service integration is analyzable as whole
- Whole request execution is analyzable and traceable
- Full debug info available at error occurrence

Service as a Service platform

Resource management

- All system-level resource management tools are available (same as with K8S)
- Fine-grained resource isolation is implemented at programming language level
- Resource and code optimization is designed to be performed on top of built-in analysis and analytic profiling
- Splitting service into microservices is performed by service configuration

⬇

- Highest security standards
- No overheads for container-level isolation or container deployment
- Extremely effective code optimization procedure
- Convenient and flexible analysis-driven resource management

Service as a Service platform

Service administration

- Initial deployment at cloud service requires no configuration,
  same 2 clicks as forking on GitHub
- Extra configuration is minimalistic
  due to absence of intermediate unnecessary entities and concepts
- No notion of container deployment
- Metrics and logs are collected without losses

↓

- Simple and reliable service administration: no DevOps needed!!
- Very fast service deployment and rollback
- Verbose information concerning incidents

Service as a Service platform

Service as a Service service

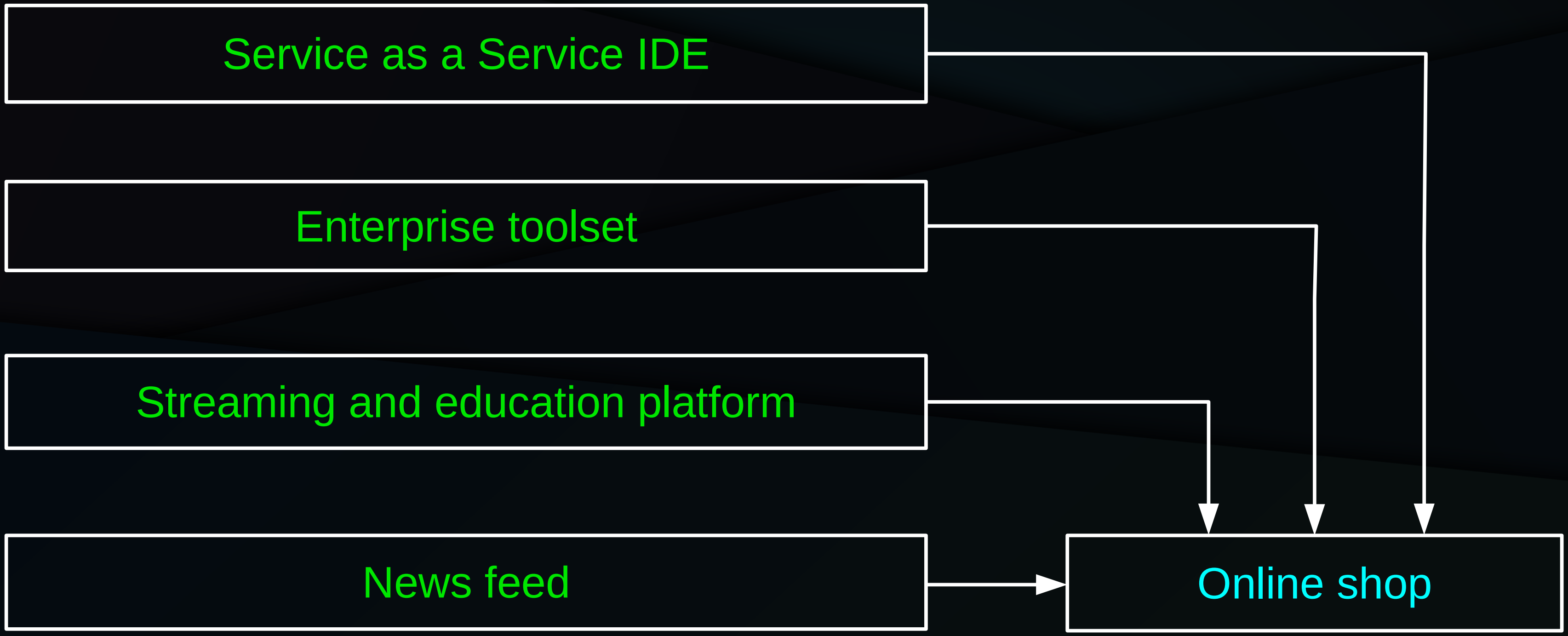
- Service is simply a tool for network schema design and callback implementation (similar to FaaS + Swagger, but with no band-aid design)
- Complex deployment schemes are available «out-of-the-box» for any service
- Deployment and request event history is collected and maintained by the platform
- Untrusted code is properly runnable «out-of-the-box»

⬇

- Effective service development
- Zero downtime «out-of-the-box» for standard operational procedure
- Actually operational Function as a Service implementation
- Actually operational Data as a Code implementation
- Actually operational Computation Outsourcing implementation

NextGen IDE

IT content delivery platform



NextGen IDE

Service as a Service IDE

- Service development IDE
- VCS integration
- Strict semantic analysis, testability, refactoring and debug
- UI for deployment and CI/CD configuration
- IDE is the only tool developer needs

↓

- High reliability and efficiency
  of development, debug, deployment and incident resolution

NextGen IDE

Enterprise toolset

- Tightly integrated issue tracker (akin: Jira)
- HR portal (enterprise solutions)/social network (akin: Linkedin)
- Messenger (akin: Slack)
- Notification system (akin: e-mail)

⬇

- «Out-of-the-box» enterprise toolset
- Software-driven workflow instead of manager-driven
- Unified time management
- Effective HR/career management

NextGen IDE

Streaming and education platform

- Video streaming coding and debugging
- Team-work online software development
- Built-in tutorial framework
- Gamified education
- Built-in Q&A service
  with guaranteed compilable and runnable snippets (akin: Stack Overflow)

↓

- Effective staff education and expertise sharing
- Convenient and controllable staff hiring

NextGen IDE

News feed

- Project suggestions
- IDE extension suggestions
- Educational material suggestions
- Vacancy suggestions
- HardCode and infrastructure publications
- IT-industry news

⬇

- Recent and relevant information for developer
- Wide terms for advertising

NextGen IDE

Online shop: core and partner solutions

- Service as a Service cloud platform services
- IDE extensions: debuggers, analyzers, editors, linters, etc.
- Enterprise toolset: multiple stuff
- Educational content, reviews, hackathones, steam subscriptions (akin: YouTube, Twitch)
- Libraries and services (akin: GitHub)
- Client applications (akin: Steam)
- Game engines (akin: Unreal)
- Games
- Browser platform

Funding and collaboration

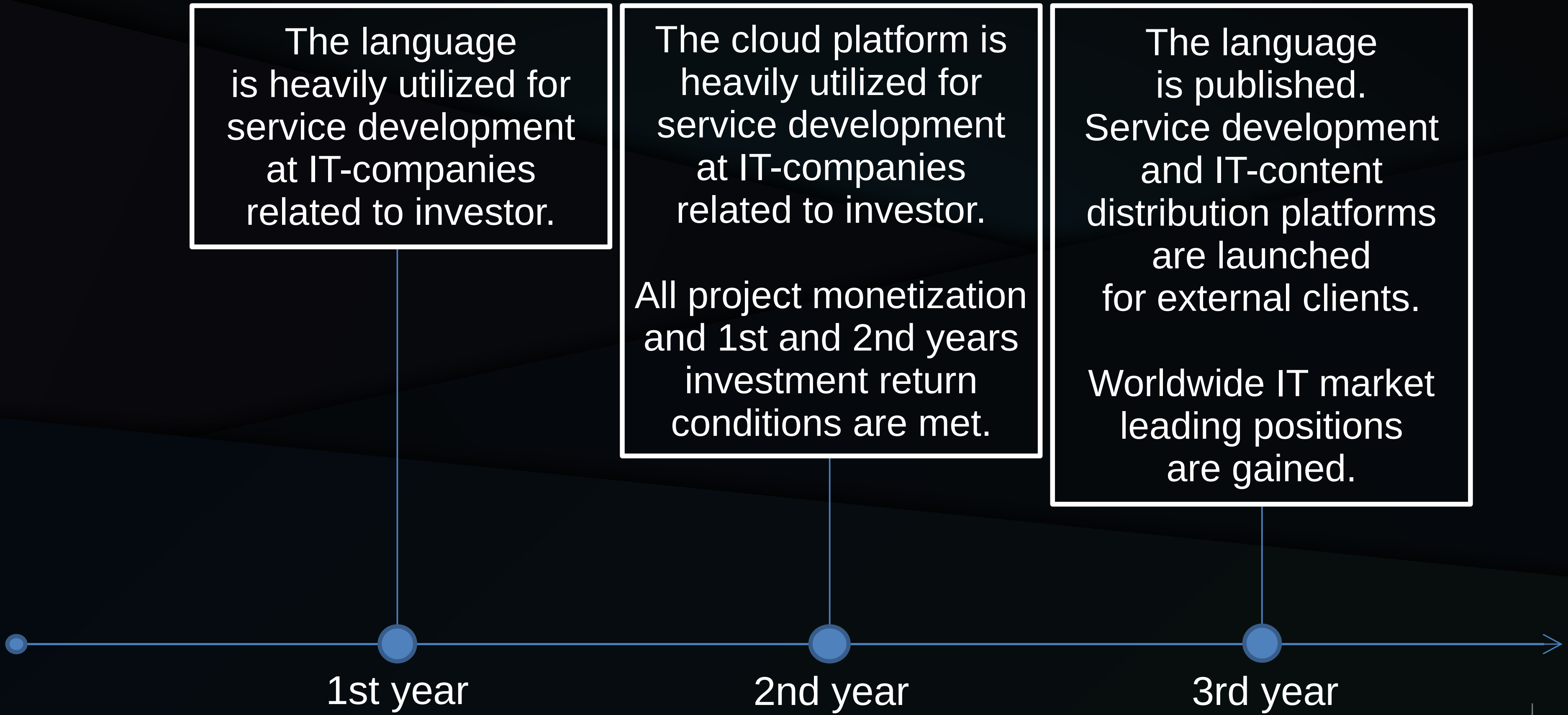
- Development.
  Core developer team funding. 10-20 developers.

- Initial use cases and feedback.
  Collaboration with IT company developing services for internal use.

- Advertisement and building up customer base.
  Funding cloud platform operation and advertisement.

- IT content delivery platform development.
  Funding platform development.
  Collaboration for enterprise software development and advertisement.

Project roadmap (annual results)



1st year

The language
is heavily utilized for
service development
at IT-companies
related to investor.

2nd year

The cloud platform is
heavily utilized for
service development
at IT-companies
related to investor.

All project monetization
and 1st and 2nd years
investment return
conditions are met.

3rd year

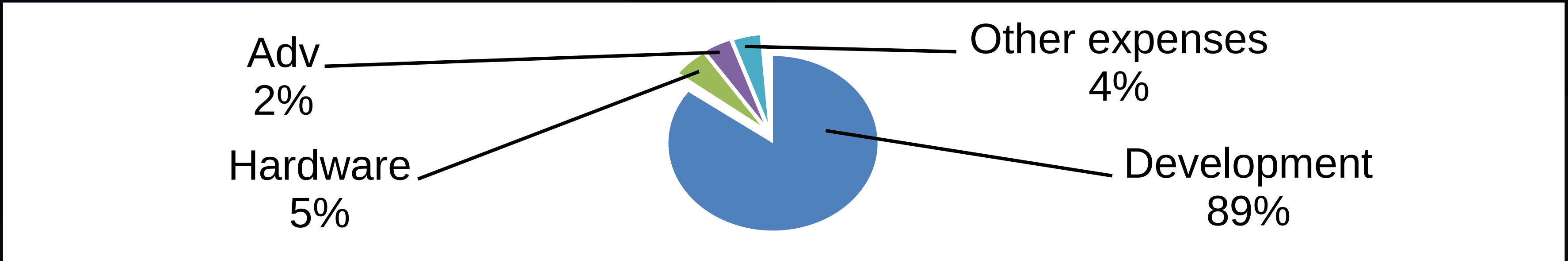
The language
is published.
Service development
and IT-content
distribution platforms
are launched
for external clients.

Worldwide IT market
leading positions
are gained.

Financial plan

	2022-2023	2023-2024	2024-2025	2025+
Expenses, $'000,000	3.0	5.0	8.0	20.0
Revenue, $'000,000	0	2.0	70.0	550.0

Approximate cost structure (1st year) – $3,000,000



Current state

- Detailed HardCode programming language concept has been written
- Technical description with source code snippets
  in both current languages and the HardCode language has been written
  (Detailed technical presentation takes about 2 hours excluding Q&A)
- Innovative multithreaded non-blocking Garbage Collector
  playing key role in memory management
  has been designed and implemented
- Language-integrated version control and hot development system has been implemented and verified in production
- Detailed Service as a Service platform concept has been described
- IT-related content distribution concept has been developed
- Development and integration financing scheme has been described
  as well as roadmap and hiring list
- General monetization principles of the product have been defined
  based on top IT solutions of the market
- All the technical aspects have been reviewed by experts

What's the interest of investors and software companies?

- New powerful software development technology.
  Worldwide business building capability on top of new fundamental designs (HardCode language and Service as a Service platform)
  → way more resource efficient service development✓
  → hugely reduced costs for service maintenance✓
- Reliable services, quick customer issue handling
  → customer loyalty✓
- Development, debug and testing optimization
  → fast product launch✓
  → effective product development and maintenance✓
- Cloud service platform as a product
  → low estimate ROI is at least 300-400%✓
  → new partnership relations (no fear of lock-in)✓
- HR-brand
  → highly skilled developers involvement✓
  → reduced staff turnover✓

Summary

Objectives

- HardCode: General purpose programming language
  - Setting a new level of code security, expressiveness and performance
    Programs don't crash and work faster while code is much shorter
  - Shifting the software programming, debug and optimization paradigm
    Getting rid of lame routine, boosting programming workflow extremely
  - Implementing sandboxing that actually works now
- Service development and deployment platform under Service as a Service paradigm
  - Dramatically change service development and administration procedure
    2 clicks to deploy
  - Throwing away lame occupations out of life
    No need for Admins and DevOps
  - Making exciting concepts actually viable
    Function as a Service, Data as a Code, Computation Outsourcing
  - Secure network interaction schema consistency, drop down network failure to potential minimum
    Delivering top SLA «out-of-the-box»

Summary

Objectives

- NextGen IDE: IT content delivery platform
  - Service development IDE
    Developing and administering services with uniform toolset
  - Version Control System
    Getting the whole customer base — developers
  - Extended enterprise toolset
    Maintaining enterprise software development workflow
  - Streaming and education platform
    Shaping developer culture for our ecosystem