Bogusław Miśta

Świlcza 460
36-072 Świlcza
Poland

email: bodziomista@gmail.com
twitter: @ iaintshine
github: iaintshine
mobile: +48 509947122

About

Hey guys, I'm a self - managing generalist programmer with passion to learn. I constantly go beyond current superficial knowledge in order to seek newer, faster, better, scalier ways to solve problems. These days I'm absorbed by building and maintaining high - performance, fault - tolerant and scalable distributed services designed for gaming and players in mind. And yeah, I love working with open source technology.

Areas of Interests

- Social online gaming platforms
- Multi - threaded programming
- Object - oriented and functional programming
- Large - scale distributed systems
- (Micro) Service - Oriented Architecture
- Message Oriented Middlewares
- Beautiful REST API design
- NoSQL databastores
- Configuration management

Technical Highlights

I write native code in	C++, Objective-C, Go
Script in	Ruby, Coffeescript, Javascript, Thrift IDL
From time to time I hack in	Scala, Erlang, Python and R
I used to write mobile apps in	Objective - C for iOS
Know some programming paradigms	Object - Oriented and Functional
I try to parralelize my code with	Schedulers, Actors, Communicating Sequential Processes, MapReduce
I develop software using	Test - Driven Development, Behavior - Driven Development
Platforms I used to work with	Windows, Mac OS X, Linux, Microsoft Xbox 360, Sony Playstation 3/Vita, iOS
The webframeworks I wrote in	Node.js, Restify, Rails, Grape, Sinatra
I've designed some web APIs using	REST, Hypermedia, Thrift API
The protocols I've used	HTTP, OAuth 2.0, OpenID, JWT, JSON Schema, GeoJSON
To version my code I use	Git, SVN
The text editors I work with	Sublime, Vim, Visual Studio, XCode
SQL and NoSQL stores I worked with	PostgreSQL, MongoDB, Redis, Elasticsearch, Memcache
I try to tam the Zoo with	Zookeeper, Hadoop
Taming services is a different strory	Sentry, Statsd, Logstash, Nagios
To provision my servers I use	Ansible

Experience

Platform Architect and Engineer, Reality Pump

September 2012 - present

As a platform architect I was responsible for the architecting and building backend solutions used in gaming. One of my duty was to provide technical guidance and mentoring to awesome team of two other engineers building user - facing application as well as some of the backend services. Writing platform design documents, REST API design, technology research and setting the technical direction were on a daily basis. As a software engineer responsible for writing language agnostic extensible framework used across services, implementing client and server APIs, writing, running and automating unit and acceptance tests. Load testing and operational automation were my duty as well. Continually looking for ways to improve existing monitoring, deployment solutions and devops practices.

Projects:

- **Topware's Platform** - distributed, micro service oriented, highly scalable gaming platform with more than 10 loosely coupled services and still growing. Each of the service exposes Thrift API to support easy and efficient internal communication. Thanks to Thrift we didn't have to constraint ourselves to any particular language or platform. Json REST API is used to communicate with user - facing applications. Web API standards were created and used across backends. Services can post to and listen for events on global message bus. Advanced routing capabilities were a need hence RabbitMQ was an obvious choose. Each of the event is audited by the Auditing Service. After the collection the evaluator can compute aggregation statistics post hoc. All the events are stored in MongoDB NoSQL datastore. UAA (User Authentication and Authorization) Service is the server issuing access tokens to client after successfully authenticating the resource owner. To access protected resources we are using OAuth 2.0 protocol, where the OAuth access tokens are JSON Web Token encoded Bearer Tokens. There are multiple of different services of which description could take multiple pages hence if one want to know don't hesitate and ask me directly. We have implemented internal framework called Vineyard for both Ruby and Javascript (Coffeescript) that we are building internal services around, Go support is coming. The framework supports multiple run modes: web server, thrift server, command line, background worker, command line thrift client, and cron task. REST API load tests are performed using Gatling and Scala DSL whereas thrift services are tested using Locust tool and Python code. To automate software provisioning and service deployment we use Ansible.
- **Two Worlds II Online Backend** - one of the requirements for the game except for social gaming features was to bring micro - transactions so users could buy virtual currencies, unlock items and customize their characters. Logical continuation to G2Liveμ. Two additional services were introduced into the backend - G2Payment (written in Ruby and used as a proxy service between the game and payment gateways) and modified Prestashop (written in PHP and used as a user - facing store, modification to the shop were introduced by other company). All processes were communicating using internal json REST API.
- **G2Liveμ** - simple mostly monolithic server backend with social gaming features, written in Rails using Ruby language, sessions and scores are stored in Redis, achievements, achievements description, leaderboard descriptions and user data are stored in MySQL database, statsd and graphite are used to track number of concurrent user sessions, number of requests etc. Service is made of two distinct processes. The first one exposes json REST API to communicate with, admin and user dashboards and the second one is used for background processing - sending emails.

Backend Technology Solution Engineer, Better Reality

May 2014 - June 2014

As a solution engineer I was working closely with the client by providing necessary documentation and expertise during the evaluation process of the backend platform for their needs. Custom solution was designed from the ground up exactly for their needs with scalability in mind. The backend system has a distributed nature and employs the microservice architecture where services can communicate using RPC (thrift) calls or can post events to global message bus with advance routing capabilities (rabbitmq). Services are discovered using separate distributed service discovery orchestrator and all events in the system are audited by the central auditing service. User facing features:

- Email and social registration, anonymous accounts support (full profile transfer possible)
- Standard and incremental achievements with social comparison charts
- Arbitrary data Geo - leaderboards with hierarchical data drilling (world, zone, country, state, local area, friends/social circle)
- Advance privacy options (location, division restriction)
- Location pinning - user might choose for which country one wants to play for

Engine Architect and Developer, Reality Pump

January 2012 - September 2012

Working closely with Lead Engine Programmer, I take responsibility for high level decisions regarding, low – level technologies, to be used and researched in a modern engine crafted internally at Reality Pump studios. Research topics:

- R&D of a new solutions to n – way threading of game technologies (lock – less subsystem communications, task – stealing job scheduling, property based and observer based distribution changes using distribution controller).
- Responsible for design and implementation of G2Live system - service abstracting platform specific online social and multiplayer gaming services with backends for Valve's Steamworks, Sony Playstation Network, Xbox Live, Apple's Game Center and internal lightweight dedicated solution called G2Liveμ. Each of the backend is fully asynchronous and allows to play without internet connection because of the support for local data caching with embedded SQLite.
- Design and development of new virtual file system allowing great performance, flexibility and rapid prototyping (custom compression algorithm – dictionary based deflate, to allow seeking in compressed file packages, hot – asset reloading, file system queries, flat and recursive directory iterators, URI based path resolver, virtual mount points e.g. network mount point, DVD mount point, etc).

Engine Architect and Developer, Mobile Division, Reality Pump

March 2011 - January 2012

Responsible for managing a full life cycle of a new lightweight cross - platform game engine from designing through implementing to testing. Engine was designed from ground up for ease of use and optimized for mobile platforms. Features highlight:

- R&D 3D – ET technology implemented and SIMD optimized for all supported platforms. http://www.3d-et.com/en/index.html
- To prevent thread oversubscription task requesting job scheduler was incorporated and used across subsystems
- For fast assets iteration remote (on device) hot – asset reloading client subsystem was written with native dedicated server for Mac OS X and Windows platforms

PS3 Programmer, Reality Pump

October 2010 - March 2011

Hired as a PlayStation 3 platform specialist at the end of the Two Worlds II project lifetime to be support for a Lead Graphics Programmer. Mainly responsible for low – level SPU kernels programming. Some of the tasks include:

- Re - implementing HLSL GPU Navier – Stokes kernels to SPU
- Porting of internally used SIMD - friendly XNAMath library to PPU and SPU Units

Senior Engine Programmer and Architect

March 2009 – October 2012

As an engine programmer I took responsibility for general rendering technology and low – level optimization and system programming. I analyzed game runs to find bottlenecks in both CPU and GPU sides using GPAD, PIX, RAZOR, VTUNE, XCODE INSTRUMENTS and other profiling and tuning software. CPU side features were optimized using pipelining and vector intrinsic. GPU features were optimized for all platforms utilizing hardware to maximum. As an engine architect responsible for designing easy to use, maintainable and readable APIs. Incorporating design patterns, high – level programming methodologies, designing classes and interactions using UML modeling language, were common practice.

Game and Engine Developer, Gamelion Sp. z o.o

July 2007 - December 2007

Responsible for managing full life cycle of programming a Global Cricket project from designing through implementing to testing. The game was developed by Gamelion Studios, published by Nokia and embedded in every Nokia E51 as the game showing rendering capabilities of the device. As a programmer responsible for writing a game engine that could be reused in future similar sport titles.

Education

Jagielonian University in Cracow, European Games Academy, Postgraduate Studies

Completed postgraduate studies at 3D Games Programming.

Rzeszow University of Technology, Master of Science Degree, Computer Engineering

Completed studies at Computer Engineering majoring in Information System with a very good result.

Diploma thesis title: “Cross – platform system abstraction layer and 3d game engine for developing advanced 3d games for mobile platforms”.*

Teaching

Periodically through studies being speaker at ‘KOD’ scientific group. I gave lecture at University of Technology in Rzeszów about mobile development and 3D games programming.

Human Languages

- Polish - Native speaker
- English - Professional working proficiency (FCE level)

Open Source

- @ Github account
- ansible-prestashop - ansible playbook for deploying Prestashop on a Virtual Machine using Vagrant
- presta_shop gem - a library for Ruby to interact with PrestaShop's Web Service API
- ccurl - command line utility over standard curl command with fancy output
- resume - my resume written in markdown format

Affilations

- DataKrk (formerly Cracow Hadoop User Group)
- Software Craftsmanship Cracow
- Cracow Ruby User Group (KRUG)
- Golang KRK

Personal Interests

Gymopsing, hardware hacking.