|  |  |
| --- | --- |
| **J. Stephen Riley** | linkedin.com/in/jstephenriley  Redmond, WA 98053  (206) 683-2560  stephenriley@google.com |
| ***Leader, Builder, Innovator*** |  |

|  |
| --- |
| **Technical addendum** |

While working in management positions, I always strive to stay technical to better support my teams. I love to roll up my sleeves and do code reviews, help with tough QA repros, lead architecture and design discussions, and conduct classes to develop my people. And I *love* to code. This document dives into my technical contributions in my various roles.

(Ask me about items tagged with … there’s a good story there.)

|  |
| --- |
| **Technical Narrative** |

|  |
| --- |
| **Director of Engineering, Data Engineering** |
| Ovation.io **** Cambridge, MA 2020 to 2022 |
| Technologies: C# (.NET Core), BigQuery, Postgres, Ruby on Rails, Python, Perl, React.js |
| ***Key Technical Contributions:***   * Designed the internal Ovation PHI tokenization scheme, vetted by HIPAA legal consultants. * Designed and developed a HIPAA “firewall” system to provide compliant PHI tokenization, allowing Ovation to develop and deploy both PHI-handling products and tokenized data products. * System architect for Ovation’s genomic database containing thousands of human genome samples, optimized for rapid sequence matching in order to match demographics and ICD-10 comorbidities. * Designed the migration from MySQL to Postgres for the flagship lab management system (LMS). Implemented an automated refactoring system to automatically convert Rails code from MySQL to Postgres. * Coached the team in migration of the JavaScript frontend code to TypeScript. |
| **Sr. Director, Engineering – Shared Services** |
| Apptio, Inc. **** Bellevue, WA 2017 to 2019 |
| Technologies: Java, C# (.NET Core), MySQL, PostgreSQL, Angular 1.8, React.js |
| ***Key Technical Contributions:***   * Pushed the TypeScript agenda by writing a code generator (perl) to produce the .d.ts files for our Carbon-based UI library * Prototyped the localization of our SSO service to demonstrate how L10N works to my teams. This was an Angular 1.8 site backed by a DropWizard Java service. * Taught classes on Unicode and its various encodings as part of the L10N effort. * Coached the legacy product team using POSA to guide their refactorings. * Built many CLI and automation tools (.NET Core) to make dev work more efficient. When in doubt, automate! |



|  |
| --- |
| **Co-founder + CTO, Board member** |
| Stabilitas.io **** Seattle, WA 2014 to 2017 |
| Technologies: Perl, .NET Core, Elixir, PostgreSQL, RabbitMQ, Twilio, ElasticSearch |
| ***Key Achievements:***   * Built a global emergency notification service for phone, SMS, mobile push, and email. Mix of Elixir, Perl, and C# services running against RabbitMQ queuing system. * Built a horizontally scalable GIS tracking system for people and materiel. Monolithic REST API service (Mojolicious framework) scaled out as necessary; extensions and async features were services connected via RabbitMQ queues. * Won SBIR grants for work on free text classification system and a free text to GIS data service. These were implemented in a mix of Python, C#, Elixir, ElasticSearch, and Google ML API. |



|  |
| --- |
| **Head of Next-Gen Technologies** |
| Avalara **** Seattle, WA 2012 to 2014 |
| Technologies: .NET 4.5, AWS, Cassandra, LocalDB, ANTLR4, SOA, DSLs |
| ***Key Achievements:***   * Worked with tax lawyers, accountants and auditors for 14 months to “solve taxes”. Our team came up with a “grand unified theory” of sales taxation that covered all US, EU, and BRIC taxation models. * With a purely data-driven design, TNG was only 7.5% the size of the legacy system, while supporting any industry or geography. * Applied compiler techniques (AST simplification) to reduce the know tax formulae from 690+ to 10. * Implemented an Excel-formula-like DSL to allow CPAs to define complicated tax logic and enable a lightweight, data-driven architecture. |



|  |
| --- |
| **Vice President of Engineering** |
| PlayNetwork, Inc. **** Redmond, WA 2009 to 2012 |
| Technologies: C++, Python, ffmpeg, OpenGL, SQLite, ASP.NET, SQLServer, Windows Server, Linux |
| ***Key Achievements:***   * Started the migration of decade-old, buggy, multi-threaded C++ code by splitting out functional subsystems into Python services running concurrently on the media appliances. * Planned architecture roadmap to migrate away from a dead-end 4D database platform to a global, distributed, L10N-ready web application. * Developed video compositing effects for low-end video hardware with OpenGL, ARBfp1.0, and VDPAU. |



|  |
| --- |
| **Other selected technical contributions** |
| Various 1997 to 2009 |
| Technologies: C++, Python, Elisp, C#, VB, ANTLR3, IIS, … |
| ***Key Achievements:***   * Used Emacs (Elisp) to process SEC filings formatted as HTML on a 90-day timeline * Built the first web API for Wall Street * Accidentally built an EC2 clone with VMWare ESX * Built a “drive-by exploit” recorder and replay system that could detect, watch, playback, and isolate the source of complex browser attacks—all automatically * Built a video appliance that cost 1/3 of competitors’ systems, and whose backend systems reduced video preprocessing requirements to O(1) (from O(n)) relative to customer base. |