Thanassis Tsiodras

✉ *ttsiodras@gmail.com*
www.thanassis.space

Education

Sept. 1990 – June 1995 **Engineering Diploma**, *National Technical University of Athens*, *grade: 8.62/10.*
Software Engineering *(I scored the 9th best score in all Greece in the entrance examinations, and received a scholarship for being in the top ten)*

Sept. 1995 – June 1999 **PhD**, *National Technical University of Athens*.
Telecom Engineering *(ATM, IP over ATM, RSVP, QoS guarantees)*

Experience

Professional coding

Jan. 2016 – today **Real-time Embedded Software Engineer**, *European Space Agency, ESTEC, TEC-SWE*.

I undertook the responsibilities of a real-time embedded software engineer, and in the 6 weeks since, I...

- identified two bugs in the RTEMS4.11/Sparc configuration
- solved both bugs, thus providing my section with an in-house built cross compiler (we can finally descend into RTEMS code when debugging on our boards)
- created the basic skeleton of a TASTE exporter to Rock (for use in the SARGON project, and eventual use by the ESA Robotics section at ESA/ESTEC)
- assisted DLR wih debugging a stack-related crash, and optimizing (by 26x!) the calculations for re-entry in an experiment they are doing (to be presented soon)
- Setup *Hudson* to automatically execute the TASTE regression suite, and a nightly build of the tip of the RTEMS4.11 branch
- Many more to come... So far, ESA has been an awesome place to work in.

Mar. 2015 – Dec. 2015 **ZenDoc**, *EverWorks*.

I undertook the responsibilities of the backend:

- created much faster implementations for our Pyramid-based endpoints - speaking to PostgreSQL over SQLAlchemy, and utilizing various service APIs during the process (MixPanel, Intercom, Segment, Recurly, Amplitude, etc). When the wrapper libraries around the HTTP APIs were not sufficient, debugging their insides was quite "interesting" (*for various definitions of interesting :-)*)
- integrated my new search-related webservices with Elastic Search
- created a standalone RabbitMQ-based queue processor that consumes async events sent from my backend (i.e. work that can be deferred so that it doesn't stall the HTTP responses, is queued in RabbitMQ and processed asynchronously. For example, this is how event notifications are sent to Segment / MixPanel / etc)
- automated many tasks using Python, cron etc - from bulk migration of users' spreadsheets to our DB, to syncing Recurly invoices and account states, etc
- migrated our servers from Rackspace to Amazon AWS. I used *Ansible* to automate large parts of the process, and afterwards I setup *monit* to monitor our services and react to downtime (the webapp is also hooked with Rollbar and NewRelic alerts).
- setup optimal (deduplicated and encrypted) hourly backups via Attic - including DB exports, rootfs, etc. I sync the results automatically to Amazon S3 buckets
- Overall: **I did both *Dev* and *DevOps***.

Sept. 2012 – Mar. 2015 **GAIA**, *NeuroPublic*.

I applied my coding and administration skills as a full-stack engineer, in the development of NeuroPublic's GAIA platform:

- Wrote a fully-functional AngularJS component for handling map data, utilizing OpenLayers 3 and automatically serializing user-provided geometry information (polygons and points) in the DB. *(Technologies involved: AngularJS, TypeScript, OpenLayers3)*
- Used *Python/requests* to write a scenario recording proxy. Placed in between browser and JBoss, the proxy was used to record complete execution scenarios from our apps, involving many steps in many tables. The recorded scenarios were then processed by a code generator I wrote, that created stress-testing Python scripts. While doing so, it modified the POST and GET requests appropriately, so that the generated scripts could simulate hundreds of concurrent users, logging-in and running the scenario *(Technologies involved: Python/requests, HTTP)*

.

- In the backend, a number of clustered JBoss instances were hosted inside CentOS VMs. I setup the Subversion repository and automated builds through Hudson - and also coded custom web gateways (*Python/Flask* backend, *AngularJS* frontend) for our developers and sysadmins: one inspecting and reporting the current version of all the J2EE apps hosted in all our development, testing and production servers (being in control of the Hudson build scripts, I was able to inject the build revision in the backing bean *.xhtml* - and I could therefore query it from the deployed apps at runtime). Knowing what version is deployed and what version is available (from Hudson) I also provided ops personnel with a "single button" to deploy the latest version of any application they wanted (the version built by Hudson) to any of the three environments (dev/test/production), using in the backend the command line interface of JBoss (jboss-cli). My gateways also generated a number of *.csv* reports, gathering data from our PostgreSQL (via Python/psycopg2) and Oracle (via Python/cx_Oracle) databases, and providing them to our ops and sales teams. Output from some of these reports was monitored via cronjobs, and appropriate alerts were raised when error conditions were detected *(Technologies involved: Python, Flask, gunicorn, AngularJS, JBoss, cron)*
- Automated the generation of JasperReports *.jrxml* files, via a custom - 10x times less verbose - description. A related article where you can see an example of the language is here *(Technologies involved: Python/jinja2/lxml, JasperReports)*

Sept. 2012 – Sept. 2013 **Optimizing the StrayLight algorithm**, *Semantix*.

In the context of the next Proba launch, the European Space Agency (ESA) will be processing radiance images on the ground segment with a custom algorithm, called *Straylight*. In task 2.2 of the "TASTE Maintenance and Evolutions" ESA project, I ported the prototype Straylight implementation from IDL to C++, and optimized the code using multithreading, SSE, cache-utilization optimization and CUDA. I achieved a 35x speedup. A further example of my ability to optimize algorithms and low-level program massively parallel machines can be found in my real-time CUDA raytracer, showcased in NVIDIA's CUDA Zone. *(Technologies involved: C++, IDL, Python, Valgrind, Cachegrind, OpenMP, SSE, make)*

2006 – 2012 **Lead Engineer in ESA's ASSERT/TASTE project series**, *Semantix*.

When developing the ASN.1-driven code generators for the Roaming Products (see below) I gained unique experience in auto-generating complex C/C++ code, based on custom-made domain specific languages. After presenting this technology to the European Space Agency, Semantix was immediately invited to join the ASSERT consortium - and in a series of eight (8) ESA-funded projects that followed, I created over 15 custom code generators (using Python and ANTLR). These covered many needs; from automatically creating device drivers that speak to FPGAs, to generating Graphical User Interfaces for TM/TCs, to auto-grafting of interworking C "glue" code for multi-modeling-tool based development (based on a hybrid ASN.1/AADL model). Large amounts of complex - and very error-prone, if manually written - code are automatically generated, translating C data structures at runtime, between C code generated from an ASN.1 compiler (developed in-house in Semantix), and C code generated from modeling tools (SCADE/ Simulink/ ObjectGeode/ PragmaDevRTDS/ etc). *(Technologies involved: Linux, RTEMS, Leon/SPARC, Python, ANTLR, an ever increasing list of modeling tools and their code generators, SystemC, VHDL, Perl, make)*

2005 – 2006 **Lead Engineer in Semantix Roaming products**, *Semantix*.

Lead engineer in the design and development of Semantix Roaming Products. Based on the company's extensive background on processing of roaming data (TAP/BER files), a new product suite was developed that covers all the roaming-data processing of Telcos. I was responsible for the design and development of the central modules in the architecture (e.g. implementing TAP3.x validations, conversions amongst TAP3.x versions, automatic code generation, runtime meta-type system, etc) as well as the porting across non-Microsoft architectures. The custom code generators that I created, parse a small, domain-specific language that I designed, specifically for TAP3 validations; and generate hundreds of thousands of error-free C++ code (we never received a bug report, in spite of the complexity of the 1000s of validation rules and the number of clients using the programs) *(Technologies involved: Solaris, Linux, Windows, portable C++, code generation, gcc, gdb, Microsoft Visual Studio 8, Python, Perl, make).*

2003 – 2004 **Software Engineer, Telecom projects**, *Semantix*.

Both Vodafone Greece and Telestet Greece (TIM) migrated their legacy rating and billing systems to Infranet/Integrate systems. I was actively involved in the development of a series of turn-key bridging applications: from a massively multithreaded C++ application that migrated customer data from the old system (Jupiter) to Infranet, all the way down to scripting automated procedures and real-time filtering of datasets. Optimizing and speed were key issues. *(Technologies involved: Solaris, C++, gcc, gdb, cron, Perl).*

2002 – 2003 **Software Engineer/Device driver developer**, *Semantix*.

In the context of a signal monitoring station, I led the design and development of a distributed CORBA-based architecture of device controllers (coded in C++ for Windows 2000), which controlled a number of recording and monitoring devices. The platform monitored the devices and performed innovative "resurrection" techniques to sustain their logic regardless of software crashes and bus-specific idiosyncrasies (RS232/TCP/IP/GPIB). The devices I programmed spanned over a wide range of signal capture and processing: real-time spectrum analysers, GPS and NMEA (compasses), analog and digital recorders (tape, video, real-time grabbers implemented as PCI cards), antenna rotators and more. *(Technologies involved: Windows, C++, CORBA, communicating with HW devices over custom APIs and buses)*

2001 – 2002 **Device driver developer**, *4Plus S.A.*.

I developed 7 Windows device drivers for high speed network boards, used to implement protocol testers. I developed in C (for kernel mode), and the corresponding APIs in C++ (user mode), addressing the needs of the 7 different FPGA designs. The final products were running under Windows NT/Windows 2000, and were used to stress-test the 3G/UMTS Siemens switches, simulating traffic from thousands of users in real-time. *(Technologies involved: C, C++, Windows DDK, Windows SDK)*

Open source coding

1997 – today **Artifacts of my free time programming**.

I write articles about coding and administration on my page on the Web.

- What's running inside my Bash?, an article at the top of Hacker News (C)
- Real-time 3D software rendering, used in Phoronix benchmarks (C++)
- Solving artificial intelligence problems, using multiple languages (OCaml/F#/C#/C++)
- Incremental backups via sparse-files, hard-links, Samba, etc (Linux/Windows administration)
- Shielding files against silent corruption with Reed-Solomon

...and many more - my articles have been Slashdotted, gained the top spot in Reddit/programming, discussed in Hacker News, etc.

Languages

English **Fluent** *Cambridge certificate, grade A*

Greek **Native speaker**

Computer skills

Programming C, C++, SQL
(Free time: OCaml/F#, Lisp)

Scripting Python, Perl, Bash
(on occasion, awk/sed one-liners)

Systems Linux, Solaris/BSDs, Windows

Web AngularJS, TypeScript, jQuery, OpenLayers3, basic CSS

Admin Linux/BSD, Amazon EC2/S3, Ansible, Attic, monit, cron, etc

Misc x86 ASM, SSE, OpenMP, CUDA

Interests

Puzzles/AI Some of my posts about solving puzzles with artificial intelligence

Soldering iron Tinkering, mostly. Here's breadboard proof, with an ATmega168.

Publications

[1] G.Mamais, T.Tsiodras, D.Lesens, and M.Perrotin. *An ASN.1 compiler for embedded/space systems*, Embedded Real Time Software and Systems, Toulouse, France. (February 2012).

[2] M.Perrotin, E.Conquet, J.Delange, T.Tsiodras, and A.Schiele. *A Real-Time Software Engineering Tool-Chain*, SDL Forum, Toulouse, France. (July 2011).

[3] M.Perrotin, E.Conquet, P.Dissaux, T.Tsiodras, and J.Hugues. *The TASTE Toolset: turning human designed heterogeneous systems into computer built homogeneous software*, ERTS, Toulouse, France. (May 2010).

[4] J.Hugues, M.Perrotin, and T.Tsiodras. *Using MDE for the Rapid Prototyping of Space Critical Systems*, The 19th IEEE/IFIP International Symposium on Rapid System Prototyping, Monterey, California, USA. (June 2008).

[5] M.Bordin, T.Tsiodras, and M.Perrotin. *Experience in the Integration of Heterogeneous Models in the Model-driven Engineering of High-Integrity Systems*, Proceedings of the 13th Ada-Europe international conference on Reliable Software Technologies, Venice, Italy. (June 2008).

[6] M.Markaki, T.Tsiodras, G.Mamais, and I.S.Venieris. *Performance Comparison of Alternative VCR Methods for Video-on-Demand*, The 7th IEEE Workshop on Computer-Aided Modeling, Analysis and Design of Communication Links and Networks, Sao Paolo, Brazil. (August 1998).

[7] S.L.Tombros, G.L.Lyberopoulos, G.Tselikis, T.Tsiodras, and ... *Fixed Access to UMTS Service Capabilities*, ACTS Mobile Communication Summit '97, Aalborg, Denmark. (Oct. 1997).

[8] N.E.Igoumenidis, G.L.Lyberopoulos, S.L.Tombros, T.Tsiodras, and ... *Towards Personal Mobile Multimedia Mobility in Broadband Networks Supporting Fixed and Wireless Access*, ACTS Mobile Communication Summit 98, Rhodes, Greece. (June 1998).

[9] N.Doulamis, N.Tsapatsoulis, A.Doulamis, T.Tsiodras, and S.Kollias. *Markovian models for the output of real world MPEG-1 encoders*, Proc. of the 2nd Erlangen Symposium 'Advances in Digital Communications'. (April 1997).

[10] N.Doulamis, T.Tsiodras, A.Doulamis, and S.Kollias. *Low Bit Rate Coding of Image Sequences Using Regions of Interest and Neural Networks*, Proc. of 3rd Intern.Workshop on Image and Signal Processing, Manchester, England. (November 1996).