+7 909 589 93 65 (mobile)
Russia, St.Petersburg, 194214 (time zone GMT+4)
kirr@navytux.spb.ru

latest is at http://navytux.spb.ru/~kirr/kirr.pdf

Kirill Smelkov

4 January 1979, St.Petersburg, Russia

Education

St.Petersburg State University, Physical Faculty,

present	Ph.D student	[Dept. of Quantum Mechanics] (on hold)
2003	MA of Physics	[Dept. of Quantum Mechanics, *thesis*] (A mark)
2000	BA of Physics	[Dept. of Mathematical Physics, *thesis*]

Publications

- Morskoy Vestnik, Vol 1(21) 2007 (in Russian).
  "Principles of data bus construction for Integrated Bridge Systems."
- International Journal of Quantum Chemistry, Vol 100, Issue 4, 2004.
  "Chemical bond modeling with correlation effects included."
- International Journal of Quantum Chemistry, Vol 96, Issue 3, 2004.
  "Adiabatic potential analysis for some carbon-containing molecules."

Awards

- Medal for Labour Merit

Free Software projects

- *RAWV – low-latency lossless video streaming over 1Gbps Ethernet.*
  http://repo.or.cz/w/rawv.git
- *Navymail – plumbing to store and synchronize mail in Git.*
  http://repo.or.cz/w/navymail.git (work in progress)

Employment

2007 – present

Marine Bridge & Navigation Systems Ltd., St.Petersburg, Russia
Senior Software Engineer

Continue working on Ship's Integrated Bridge Systems in a new role after first successful project. Designed and implemented several new game-changing technologies. Modernized target and development infrastructure and tools. Developed and maintained every aspect of the project ranging from low-level bits and custom Linux kernel to high-level application logic, networking, ui, target system, protocols and documentation, archives & integrating everything together. Managing project intranet infrastructure. Performance bottleneck analysis and tuning for target system. Debugging hardware issues. Doing releases & builds. Taking part in and guiding deployments, test-lab, remote & on-field testing, with gaining lots of integration experience for system to work stable as a whole. Part of the codebase ported to Win32. Various derivative projects are also spawned and done along the way. I designed and implemented the following:

- Network/hardware/software architecture upgrade for next-generation systems.
- Optimizations to key system parts to be an order-of-magnitude faster/smaller, more robust.
- Low-latency lossless video streaming over 1Gbps Ethernet.
- Remote deployments access over GSM (software and infrastructure).
- Next generation build system: fast, correct whole-project incremental rebuild, run from in-tree, cross-compile to Win32 via mingw + incremental porting from Win32 to Linux via winegcc/winelib; faster and incremental flash build.
- New bridge subsystems – Technical facilities, Damage control.
- Radar infrastructure rework to support sources/sinks, codecs, record/replay/stream over network; 10× compression codec. Linux driver update for new boards.
- MIL-STD-1553B testing utilities (monitor, BC replay, universal RT stub), I/O agent upgrade to also work in Bus Controller mode; maintaining vendor driver.
- Docutils/TEX extensions and styles for ЕСПД (Russian unified software documentation system = GOST 19.*); Wrote and typeset several key documents and protocols.
- Unified Linux/Windows Git-based workflow for code/documentation/data/media & builds.
- On-target testing infrastructure.

Project: Radar training synthesizer. Designed and prototyped software system architecture for S-57 charts → radar signal synthesis engine. The proto works on both Linux & Win32 (via cross-compile).

Project: Ship's Automatic Weather Observing System. Took part in designing I/O subsystem architecture & implemented most of it. Running-in and debugging hardware issues. Guiding the project through testing stages and first deployment.

Project: ECDIS Communication Unit. Continue maintenance and developing for-service features. Doing releases. Taking part in and guiding deployments. Debugging issues remotely & on-board.

Used technologies and tools: KISS approach, Debian GNU/Linux, C/C++/Python, Git, TopGit, msysGit, git-annex, GNU Make, Docutils/RST, TEX, VLAN, perf, grub2, tdb, cython, swig, qemu, py.test, GSM modems, HTTP, SSH, plymouth, gcc, mingw, tinycc, ctags, S-57, GDAL/OGR, cairo, VNC, wine, Qt, SDL, SVG, fuse, dokan, wiki, bugzilla, mailman, etc...

2003 – 2006

Marine Bridge & Navigation Systems Ltd., St.Petersburg, Russia
Software Engineer

Taking part in Ship's Integrated Bridge System development, then leading the project. From scratch layed ground for several important components; took part in system architecture design. Developed and maintained every aspect of the project ranging from low-level bits to high-level application logic, network, ui & integrating everything together. Managed project intranet infrastructure. Performance bottleneck analysis and tuning for target system. Debugging hardware issues. Experimental and on-field testing; taking part in first deployments. I designed and implemented the following components:

– (in part) system architecture.
– distributed publish/subscribe communication middleware.
– I/O agents – Serial, MIL-STD-1553B/RT, Proprietary/Ethernet.
– Protocol libraries – NMEA-0183, MIL-STD-1553B family, Proprietary (lots).
– Unified publish/subscribe data namespace: I/O is done by exchanging {name → value} pairs.
– Linux driver for RADAR capture board.
– Radar rendering engine, Qt+SDL overlay, radar and ARPA-tablet GUI.
– Bridge subsystems – Audio, Power, Time, Video (in part), Targets...
– Linux driver for several CAN boards; in-house CAN/CANopen-based protocol.
– NUT driver for Eltek AL175 UPS alarm module.
– Build system for code & target flash.
– Tools to perform live host/target run via network.

Also took over ECDIS Communication Unit development and continued its legacy QNX4 codebase for 7+ months. Later switched to Linux & IBS codebase to unify efforts. Like with Bridge, guided the project through testing stages and first deployments.

Used technologies and tools: OO approach, C++, STL, C, Python, Debian GNU/Linux, X11, SDL, Qt, darcs, cvs, gcc, g++, gdb, oprofile, valgrind, ctags, wiki, TEX, docutils/rst, doxygen, bugzilla, mailman, wine, freetype, x86 assembler, MMX, etc...

2001 – 2003

Night Bird Software Ltd., St.Petersburg, Russia
Software Engineer

Taking part in Home Automation System project. This first work allowed me to learn GNU/Linux development a lot. I designed and implemented the following components:

– Video: *record, playback, streaming to net*; also DC10+ video-capture board hardware & its driver modifications to support subimage feature.
– Audio: *record, playback, streaming to net*; plus Java applet for GSM playback on client side.
– Modem: *voice, fax, ppp.*
– Speech: *text-to-speech synthesis for Russian and English* using Speaking Mouse (Win32 DLL via Winelib) and IBM ViaVoice for Linux engines.
– Custom distribution based on RedHat Linux. I integrated crypto-fs support into the system and wrote additional parts to the installer (UPS setup, raid tuning, custom installation profile).

Used technologies and tools: OO approach, C++, C, Python, Java, Linux, speech engines, wine, anaconda, cvs, gcc, g++, gdb, valgrind, ctags, etc...

Additional experience and skills

– Good math and learning abilities.
– Can concentrate on high-level design as well as on low-level bits when needed.
– Read and navigate through other's code easily.
– Threading & concurrency, distributed systems, real-time, networks, protocols.
– Once took part in cognitive psychology study. Research done with R+Graphviz.

Hobby/study and other projects developed while being at school/university:

– Tools & optimized libraries for quantum-chemistry research (Fortran, C).
– AON (Russian analog of Caller ID) software decoder. Hooked it to vgetty.
– Firmware for Z80-based modem, debugged low-level tx/rx protocol. Later developed this modem software emulation for Linux (via sound card).
– C compiler for Z80 (started with a friend, later put on hold).
– Custom accounting system (MS Excel, VBA, PC Anywhere).
– Binary patched Laser Squad to work via modem (enjoyed playing it with friends).
– Simple multitasking kernel for Z80 (dreamed since first heard about UNIX).
– Visual image recognition based on 2D density series (high school diploma).
– At school I used to program on ZX Spectrum (Basic, Z80 assembler) and in the first years I enjoyed reading books and pen+paper programming (had no computer at home).

Languages	Russian (native), English (technical)
References	Available upon request
Hobbies	Horses, studying things

```
                ~,  ^^              |
                /|    ^^          \ _ /
               / |\            -=  ( )  =-
~^~ ^ ^~^~ ~^~ ~=====^~^~-~^~~^~^-=~=~=-~^~^~^~
```