

## GSoC Interview: Alexey Burshtein.

Contributed by DaaT  
Sunday, 31 May 2009

Today we here at ICO bring you one more interview in the Google Summer of Code series, with both accepted and non-accepted students. Today's interview is with Alexey Burshtein, Israeli with Russian origins.

So without further delay, click below and I hope you enjoy reading it.

ICO - To start things off, why don't you tell us about yourself and And how did you start with coding? What made you want to do it? [Ed. note: these were two separate questions but Alexey decided to answer them together as you can read below]

Alexey - These two questions are very closely related, so I answer them both at once. My name is Alexey Burshtein. I'm Russian in origins, (or should I say Soviet? :) ), born in 1978 in West Siberia, in the city named Irkutsk. By the way, I would like to use this opportunity to correct a mistake most people do when thinking of Siberia: usually it's thought to be a freezing cold, snowy, blizzardy place. And while it's usually so in winters, with typical temperatures going down to  $-30^{\circ}\text{C}$  ( $-22^{\circ}\text{F}$ ) and lower, the summer there is usually striking hot, with temperatures typically climbing up to as high as  $+40^{\circ}\text{C}$  ( $+104^{\circ}\text{F}$ ). No one said life is easy, especially in places like these :)

Anyway, when I was five years old, my family moved to another place of Russia, to a small city Nalchik located in North Caucasus, just about 200 km from Chechnya, (famous for its war against Russia). There I started my school studies, and there happened my first experience with computers. When I was seven, my father started working for a construction company, and they performed some calculations on ES Computers, which were basically the same old IBMs stolen by Soviets and manufactured under a different name. The technology used there can now be seen possibly only in a museum, and it would require a large exhibition hall: the computer as powerful as a nowadays pocket calculator took up a room the size of a basketball playground with controlled air temperature and humidity. And I'm standing by my words: I "worked" with these things. Even at that time &mdash; note I'm speaking of 1985 &mdash; western world was already enjoying Apple Macs, first version of Microsoft Windows (Lord keep us outta there! :) ), hard disks and other fruits of progress. At the very same time I was tapping my first programs (text-based games, "rogue"-style) in BASIC and stored them on punchcards, punched tapes, and rarely, on magnetic tapes &mdash; if I was allowed to waste precious storage resources and computer time for my coding attempts (which was usually not a problem since my father was in charge of the department :).

This was also the place where I saw the first PC-sized computer ("Iskra-226", a computer designed by russians, actually, a replica of Wang 2200). I got my hands on it in 1987. It featured 8-inch floppy disk drive, diskettes preformatted for 237 KB, graphic 2-color display with stunning 31 cm (12") diagonal at 512x256 pixels, and weighed just about 40 kg (90 pounds). Its enormous advantage &mdash; and I'm not kidding here &mdash; was BASIC being integrated into command prompt. Thus, the operator could just start typing in the program (10 INPUT X, "Enter", 20 PRINT X etc.) at the command prompt, run it, debug it &mdash; and do all of the above without opening a specialized development environment. I personally think this advantage was underestimated, and a command prompt which is also an IDE is a nice thing.

This computer also featured an external hard disk &mdash; at the size of grandpa's vinyl LP player and with capacity of 5 MB. I really could see the reading handle moving through the translucent top cover, and it was an amazing feeling writing a program which controlled the head's movement. I was not allowed to write to or read from the hard disk, but I did program the handle's movement so it would "dance" to my favorite songs. The sounds performed by the motors while they were moving the head, rotating the disk and speeding it up and down were all different, and I was trying to create music with these sounds. Sometimes I even succeeded, but usually not :)

Then IBM gave my school a set of IBM PS/2 computers as a gift, and when I arrived at 8th grade I started studying more advanced languages like Pascal. Usually I was among the best in class. I even designed an opening, a headpiece for my class assignments: a graphic intro with flashing stars, flying rockets, waving banners (in interstellar vacuum &mdash; realism was not something I was concerned with ;) ) etc. You should see the teacher checking the class assignments and running into my intro, with all these bells and whistles introducing something like finding a maximum out of three numbers or bubble-sort implementation. Besides the school classes, I participated in an after-school extracurricular group which studied algorithms and problem-solving at a more advanced level. There it all started... I continued to program after I moved to Israel in 1994, and haven't stopped since then.

Until 2001, I didn't thought about becoming a professional programming. On the contrary, I dreamed of becoming a physicist, and did everything to achieve this dream. The first doubts in my decision arose after I realized that advanced nuclear physics (which was the subject I was particularly interested in) is almost 95% advanced math, and my grades in

"Introduction to programming" are much higher than in any course on physics.

Since 2002 I was working for IBM in IBM Haifa Labs, in its Systems & Storage department. My responsibilities were supporting and developing debugging tools, developing of the advanced mirroring (continuous copy) services for DS8000 and DS6000 high-end enterprise-level storage servers, code maintenance etc. I left IBM this April to continue studies for my Masters degree.

I should note that the thing that attracted me to computers were computer games. Good old titles like "Prehistorik", "Nightmare on Elm Street", "King's Bounty" (the precursor of "Heroes of Might and Magic" series), "Starcon", "Wing commander" and, of course "Prince of Persia", among others, were almost perfect &mdash; but not totally perfect. Therefore, I felt urge to "correct" them, to make them behave more in tune to what I would like to see. But when I started working on a game, it turned out it was easier to create a new game from scratch than to disassemble and "correct" an existing game (Pascal and BASIC were all I knew then). As far as I can remember, the only thing I completed and "released" among my friends was a FPS spaceship simulator. In my opinion, games are very important for the development of a programmer, they are at least a good point to start from.

ICO - When did you learn about Haiku? Had you any previous experience with BeOS or Zeta?

Alexey - In 1999, I was working at Israel's largest ISP, "NetVision Ltd.", in the department of private customers' technical support. Remember those guys you call when you can't connect to the Internet? I was one of them. Hell, I was one of them for almost 4 years, from Jan 1999 to late 2002, while the average person escapes from this department after 8 months. Since then I've been thinking of writing a script for an IT horror movie, based on the real stories, but I'm afraid this movie will be banned for extreme violence :)

Anyway, on one of the cold December evenings, on a long and lonely shift, I caught one of my colleagues running through the department with a CD in his hand. The CD carried an inscription "BeOS 4.5".

Two weeks before that day I bought a video adapter, nVideo Riva TNT2 M64, and when I was wandering the nVidia's web site in search for the drivers, I noticed there were also drivers for BeOS. Its logo was so attractive, blue-white-and-red... It jumped to my memory at the moment I saw that inscription. Surely, I grabbed that CD out of my colleague's hand and copied it (see, at that time "paying for software" was for me something that happens to bad folks in Hollywood movies).

I installed the OS onto my computer, and was amazed by the speed, responsiveness, quality and usability of it. One of the first things I did was connect it to the University's LAN (I was living in the dormitories at that time). It took mere 20 seconds to accomplish this target. I remember myself sitting astonished in front of the keyboard, claiming it just can't be that simple. In Windows 98 (which was the only other OS on my computer at that time), I had to reboot several times to achieve the same network connection. The blue color of the desktop's background was exactly of my favorite's tint. Along with shiny yellow tabs, the OS immediately became my best friend. In fact, it was the major operating system on my computer until last autumn, when it stopped booting because of some hardware changes.

I rarely refer to pieces of software as animated, living creatures, but this was one of the cases: I immediately started thinking of BeOS as a pretty, athletic, sporty girl, who wears an attractive dress, who sometimes behaves silly and immature, who sometimes is capricious or jealous (especially when she doesn't feel enough attention - i. e., you don't boot it for a long time), but usually she's full of forgiveness and ready to help you, to support you in every way she can. It turns out that I'm not alone - and here, for example, in the discussion of this picture one person immediately yelled "Hm... Haiku!"; it seems, he also associates Haiku with a girl. My point is: BeOS and her derivatives are living beings, just like the ships are. By the way, Windows is associated for me with an ugly, slow, heavyweight working bullock, nothing more.

My obsession with BeOS can be judged by these facts: one of the most poetic compliments I ever said to a woman was "You're even better than BeOS". If this does not convince you, then consider this: I have three cats, they're named Bes, Haiku and Zet - guess in honor of what. :)

Since early 2000 I'm a fellow participant in the Russian's BeOS community (located now at <http://www.qube.ru>), known as Alex Hitech. I've written several programs for BeOS, but usually I don't release a program until I'm assured of its quality, therefore my list of releases is very small.

[Ed. note: Alexey here adds in a follow-up e-mail] By the way, I forgot to mention that one of the things I made for BeOS was writing a short book "Immigrant's guide to BeOS" (previously named "BeOS Bubble", in honor to the BeOS Bible). It covered roughly chapters 1-5 of the BeOS Bible, i.e. description of BeOS, its advantages and disadvantages, installation, detail cover of all programs included in BeOS and guide for connection to the Internet. The book was written in Russian, it was published online until very recently, and was included on the "BeOS / Zeta expedition CD.

ICO - How did you find out about GSoC and when were you first interested in participating?

Alexey - Since I was watching the BeOS community from about 2000, I was well informed about the Google Summer of Code sessions during the last few years - in fact, since Haiku started participating in GSoC. I thought of participating, but that would require an approval from my managers, which would not be an easy thing to get. By the way, developing a freeware but closed-source program usually does not require any approvals; the only condition is that the program should not directly compete with employer's product.

The problem is as follows: if someone works on an Open-Source project in their spare time, they could accidentally learn from there a solution which they would also use later in their closed-source work. Bringing a GPL code into storage software would be catastrophic for my employees, because due to "infective" behavior of GPL they would have to release the whole software of storage server under GPL license. Small companies may even go bankrupt because of this. Therefore every software company that develops proprietary software takes measures to prevent even accidental contacts between the employees and the open-source code which was not studied by the company's lawyers. It goes to extreme: in some companies, employees who develop an open-source project, sit in another building and are not allowed to contact by any means employees who work on almost same, but proprietary project.

And that, in turn, means, that if someone wants to work on an open-source project, they absolutely must verify it with their manager, the manager then checks the project's description with the lawyer, and it, like a snowball, can go up to the highest management prior to be approved. Businesses do take their licensing seriously, oh, they do!

If we return to me and GSoC, I got a principal approval to develop my own applications for BeOS (under condition I don't develop a storage server competitive to DS8000 :) ), but my engagement in GSoC activity would require a much tougher approval process, thus I didn't even try to submit an application.

This year, however, I knew I was going to leave IBM, therefore I decided to finally take a chance at GSoC. I've submitted 3 applications, all three for different Haiku projects.

ICO - What convinced you that Haiku is a project worth working on?

Alexey - As you may understand from above, I'm a huge fan of BeOS and its derivatives. Haiku is worth working on by default. :)

Seriously, the strenghts of BeOS multiplied by the free and open-source license are, in my opinion, overwhelming. I don't mean that the whole world suddenly stop using Windows and install Haiku instead, (though it will be the perfect future :-P ), but, seriously, Haiku is elegant, well-designed and fast. And it's easy for both users and programmers to play with.

Two facts that will support my claim:

Firstly, my mother is a very special computer user. If Wikipedia will ever have an article about "primitive computer users" (and I mean the opposite of "advanced computer users"), this article will have a picture of my mother as the illustration. She still uses a computer with 80486 processor and Windows 3.11, because it supplies all her needs, (read "runs "Solitaire"). But nevertheless, she succeeded to work just fine with BeOS when she had to use my computer - launching programs, games, getting on the Net, playing music and movies wasn't difficult for her in BeOS. To make things harder, BeOS wasn't localized, while my mother doesn't know English at all, - yet she mastered her way all right. For me, this is exactly what's meant by "intuitive user interface": even a user whose experience with computers in almost nonexistent succeeded in accomplishing tasks she wanted to. A strong point of BeOS is the interface and since it's Haiku's interface too, it's one of Haiku's strong point as well.

Secondly, I'm a programmer. I used to program at operating-system level in various operating systems, including AIX, QNX and OS/2 (I was lucky to work in same department with people who developed OS/2). And I have to admit, there was no OS in which applications' development were easier. The API is smooth, logical, and it makes sense - which is not the case, for example, of Win32 API.

Ease of use and ease of development - these are the strong points of Haiku, which, I believe, promise its happy future.

ICO - What did you apply to work on and why did that project specifically interest you? Were you divided between that project and any other on the list? [Ed. note: here Alexey brings two questions together once more]

Alexey - As I already said earlier, I've submitted 3 applications. I'll describe in detail each of them:

#### 1. Web cam driver

As more people will use Haiku, some of them will use this to-be-Multimedia system for things which are pure multimedia. Ability to use a camera is an essential thing in modern multimedia experience, and as more people participate in video chats, they will be upset if they are able to use cameras in Linux or Windows, but not in Haiku. Besides, I thought it will

be nice to have Haiku User Groups parties being broadcast on the Net in real-time :)

## 2. Improving WebKit and creating native Haiku WebKit-based browser

The more people will use Haiku in their everyday life, the more Haiku will benefit from it. Since most computer users use them to play games, edit documents, play multimedia (reading e-books is also playing multimedia in this case) and surf the net, creating a full-featured fast browser will solve a quarter of newbies' problems in Haiku. Or even half, if Google Documents are good enough for their purposes.

## 3. Creating a calendar/organizer application

This one is my favorite. I started working on this program long before submitting the application. Actually, the reason was - I had to find a way to store my timetable in a convenient way.

Trivial option would be to use a spreadsheet like MS Excel (God forbid!) or GoBe Productive, and build the timetable there; then make a hard copy, by automatic means (like in "print it") or manual (copy to a leaf of paper using conventional pen), and carry it with you at all times. But this is too, let's say, old-school and low-tech; it won't reflect changes which occur sometimes.

A more advanced option will be to use a kind of time-planning software, like MS Outlook (man, they're everywhere! :)), Palm Desktop or BePlan, and make the schedule for the whole semester using recurring events. Then every single occurrence may be adjusted, if needed. More than that; this timetable can be synchronized to handheld devices or mobile phones, making the timetable portable and updateable. But then it would be great to add more functionality to the program: for example, it can remind you a day before an exam that you should start preparing yourself to it; send an E-mail; run a script; play a music etc. None of the time scheduling programs that exist for BeOS / Haiku allows it, although AutoPilot and BeThere are not far from what I mean. But, again, they can't stick their calendar onto Desktop, which, I think, is a pity: if there is support for Replicants, it should be used wherever possible (and convenient).

After all, I prepared a rough design of this application and even started to implement it. Some more details are in the application page.

ICO - How was the application process? Was anything overly complicated or discouraging?

Alexey - I must admit that requirement to solve one of the "easy" bugs is not too encouraging, but I understand the reason for this requirement. Anyway, it looks like this will be a standard procedure for Haiku's GSoC from now on, so I'd better learn to agree with this requirement :)

ICO - Besides Haiku, did you apply to any of the other organizations involved with GSoC? If so and you don't mind sharing, which?

Alexey - No, I didn't. I thought about applying for the ReactOS, but, AFAIK, they didn't make to GSoC this year.

ICO - Would you be interested in a possible Haiku Code Drive, like the one which occurred last year? [Ed. note: This interview was conducted before the Haiku Code Drive 2009 was announced].

Alexey - Sure I would. I'd prefer to complete the Calendar/Organizer application, because it will benefit me most :), but any other option is also possible.

We here at ICO would like to thank Alexey for taking the time to answer our questions and doing it in such an involved way as you could read. I hope everyone enjoyed it.