AA VV

KDE Camp 2010

Camp KDE 2010
University of California, San Diego
January 15th-22nd, 2010.
http://dot.kde.org

Calomelano Editrice Virtuale
ebook numero 15

http://calomelano.it/ebooks
I edizione gennaio 2010

Sommario

Intro

By: Troy Unrau

2010 25 Jan

Last week, the KDE Community had their yearly Americas event, this year in sunny San Diego. Despite California not living up to its sunny reputation, the attendees certainly had a good time. The first three days featured talks about a variety of topics (day 1, day 2 and day 3), there were CMake and Qt development courses and of course several small meetings and work to be done. However there was more than sitting in the conference room at UCSD. We had a great time at Banana Bungalows on the beach, went out for a variety of food, had a few dragons and babies visit us and risked our lives getting to and from the university. Read on for some general impressions on the event, and for some motivation to attend Camp KDE 2011 next January, at a location still to be determined.

Accommodations

Enjoying the view from the Banana Bungalow deck

KDE events usually choose a hostel for primary accommodations as it provides close social interaction at relatively small expense. The hostel chosen this year was Banana Bungalow, on the beach of the Pacific Coast, with a few people staying at the nearby Best Western. The location was good for watching the waves, and reasonably cheap. The hostel was a decent place until it started to rain.

The place was run by longer term residents of the hostel, and always had people walking around and having fun, mingling with the KDE crowd. We filled the big picnic table downstairs with laptops and saturated the hostel's wireless, something that typically happens anywhere that KDE people congregate. Of course, the stay was made more interesting by some local beverages.

All was fine at the hostel until the heavy rain started on Monday. It rained so much power went down, the streets were flooded to two metres, and there was a real river flowing inside the hostel. Despite the dire circumstances, spirits were still high, and once again showed how the morale within the KDE community is something that cannot be destroyed! Fortunately, unlike the other visitors at the hostel, we were able to sneak away to the university during the day to enjoy warmth, wifi and food.

Conference Location

Till Adam trains without shoes

The main event was held at the Computer Science and Engineering building at University of California at San Diego (UCSD), courtesy of their Computer Science department. The talks were relatively well attended, and an increase over the previous year to around 70 people. During the first three days of talks, there were a number of students from UCSD that managed to make it to the presentations, despite it being a long weekend in the U.S. Many of the talks ran slightly behind schedule, with an average of one hour delay each day. Next time Camp KDE runs, we will arrange for someone to remind the speakers of remaining time using flashcards or similar.

UCSD was a good location for the event, providing ample electricity and wireless access for the attendees. The event, however, did highlight the not-quite-there-yet nature of wireless support for many chipsets in Linux, as a few people were relegated to wired access. When the rain got really bad, many of the attendees stayed at UCSD late into the night to take advantage of the additional shelter provided over the hostel. The main disadvantage of the location was the distance between it and the accommodations - having the accommodations at the event location were considered one of the main advantages to the Camp KDE 2009 event in Jamaica.

Food

This pizza did not have long to live

San Diego certainly had plenty of food available for general consumption, much of which was very good. For the most part, the attendees arranged for their own food, although snacks were provided by the organizing team for consumption at UCSD. Generally speaking, people gravitated towards Mexican food, being only a few miles from Mexico. Bean burritos and fish tacos were consumed in mass quantities, but the food court at UCSD provided opportunity for some additional variety like Chinese, Greek, Italian and Thai food.

In the evenings, the attendees would often migrate to a restaurant or bar near the hostel to find some local fare. Once bored of Mexican food, people began to find pizza, Thai, and more to satisfy their need for energy. Along with the food, a few local beers and other liquor were sampled, occasionally resulting in bouts of karaoke upon return to the hostel. There are a number of recordings of KDE contributors crooning that may or may not be used as future blackmail.

On the final day of the event, those still remaining went for a more formal dinner at the Mexican restaurant El Torito, near the UCSD campus. This was the last significant gathering of the event, and resulted in a number of very lively discussions. Among the topics discussed were what to do for Camp KDE 2011, and how things can be improved for following years. In general, the event was considered a productive, successful event by those who attended.

Social Event

What is powering their brewery?

On Tuesday we had the scheduled social event. The weather forced some small changes in plans. We ended up going on a tour of a local brewery, the Stone Brewery, a medium size craft beer maker. We learned a lot about making beer, the origins of Ale and Lager and of course sampled a few different beers. The brewery was fairly technical in their details, showing us their touch screen user interface they used to monitor the brewing process. The consensus was that their interface would benefit from a few Plasma animation classes.

After the tour, we stayed at the attached restaurant for some delicious food, and watched the rain. The original plan for the afternoon had been to go to an outdoor location, such as the renown San Diego Zoo, but this was canceled due to the rain. We ended up splitting into several smaller groups, with one group heading to Tijuana, another going back to UCSD, and a third making a trip to Fry's. For the people from outside the U.S., the visit to Fry's was rather entertaining.

At the end of the day, KDE hacking had taken over, with most people congregating at the conference room at UCSD. Tables were thrown together, and the urge to hack on things could not be denied. Many remained at UCSD until midnight, to avoid the rain.

Dragons Galore!

Konqi and Katie return!

During the KDE 4.0 Release Event in Mountainview two years ago, local KDE enthusiasts decided that they would put together professionally designed costumes for KDE's mascots, Konqi and Katie. This appearance was quite well received, and since KDE was returning to California for Camp KDE, it afforded Aaron Reichmann, one of the two original people beneath the costumes, an opportunity to dust off the costumes once again.

This year, the original Katie was not available, so Plasma hacker Chani Armitage donned the Katie costume. Between her and Aaron, the dragons made two appearances to lighten the mood of the technical talks. Who knows where they will be seen again?

A Family Affair

Our youngest attendees

Perhaps as a sign of the maturing KDE community, more contributors are spawning new child processes. At this event, we had two infants in attendance, and several more attendees that left their children at home. Wade Olson (Camp KDE 2009 organizer) made a surprise appearance, with baby in tow, to supplant Kitware hacker Marcus Hanwell's child as the youngest conference attendee. According to Wade, spawning children has become a contagious event within the KDE community.

This is not the first time that children have attended a KDE event, and this is beginning to become a regular occurrence, however having two very young babies was interesting, especially when they would randomly cheer during a presentation. Overall, they were well behaved, and part of the positive atmosphere at the event.

Productivity

Marcus distracts us from the rum

Many of the benefits of a KDE event are of the social nature. Bringing KDE people together helps brainstorming and interaction in ways that email or IRC simply cannot match, so while much of the time spent during such a meeting is of the social variety, the practical benefits to the KDE community extend well beyond the conclusion of the event. But it isn't just meetings within the KDE community that occur. These events present the opportunity to meet many local Open Source personalities, such as the keynote speaker Philip Bourne or many of the UCSD students. A number of KDE people managed to arrange a meeting with a major company in the area to talk about KDE and embedded platforms. These meetings are far more useful in person than over email.

Additionally, having so many experienced hackers in the same room is good for code-related productivity. Most evenings, even if involving food or drink, also involved a significant amount of hacking. The Qt training provided by Till Adam of KDAB was invaluable to all attendees, rookies and experts alike. The Qt/Embedded training opened up a lot of eyes about the nature of embedded platforms, and how to potentially unlock the power of Qt/Embedded for KDE programmers. And Marcus's CMake training was useful to anyone that has ever tried to use the KDE build system.

While any of these training sessions were going on, a number of people were furiously coding on a number of projects. Amarok and Plasma, in particular, had a strong showing, and the immediate benefit of having these people in the same room was obvious to any outside observer.

James Cain won the Camp KDE Be Free contest, and was flown out to the event to participate. Coming from a mostly outside perspective, his experiences with the KDE community at the event have already begun to turn him into a significant contributor (his first dot story was posted already!). If Camp KDE has this affect on people, then perhaps it should be marketed as a way to get involved with the community, for future events.

Concluding Remarks

The streets were flooded

Camp KDE 2010, the second installment of this annual event, was a huge success. It showed growth over the previous year, and solidified the event as KDE's second major annual event, next to Akademy. While attendance will certainly grow in the future, the nature of the conference allows for significant productivity, and personal networking within the KDE community, as well as externally. The rain and other difficulties faced by the team were met with cheerful optimism, and really showed the true strength of the morale within the community.

A special thank you to Jeff Mitchell and the organizing team, and the many sponsors that made this event possible.

Day One

By: Jos Poortvliet

2010 17 Jan

The first day of Camp KDE is behind us, with the first of the many presentations being well attended. A full summary of the talks lives behind the break, and videos will be online shortly. If you are in the San Diego region, feel free to join us for Camp KDE for talks and training.

Saturday, the first day of Camp KDE 2010 in San Diego, started with a short introduction by Jeff Mitchell. Jeff, who was the principal organizer of the conference, introduced us to a bit of history about Camp KDE and then went into some statistics about the KDE community. The conclusion was that if we maintain our current rate of growth we'll have about 6 billion developers by 2050. Continuing with this theme, he spoke about current topics in KDE such as the migration to Git and the upcoming release of KDE SC 4.4. Jeff then introduced the talks to follow, including the work on KDE-on-Windows, KOffice and KDE technology on mobile devices.

Preparing presentations at the Hostel

The next year should be an exciting time and will see many improvements to user-visible parts of what we do. Seeing the publicity from several major news outlets lately, like IT News Today, CNET and Linux Journal, it's clear our vision is now being picked up more widely. People are starting to realize where we're going. This will bring in more interested developers who want to be on the cutting edge of innovative technology - and the KDE community is that spot.

At the end, a big thanks went to the sponsors: Collabora, froglogic, Google, KDAB, Nokia Qt Development Frameworks, UCSD, KDE e.V. Also thanked were the other people who helped Jeff organize the event like web developer and camera magician Leo Franchi, artist-ninja Eugene Trounev and UCSD liasion and swiss armyknife Andrew Huyng who functioned as the local team.

Philip Bourne on Open Data

Professor Philip Bourne followed with the opening keynote about open data. Philip Bourne is a computational biologist, the 2009 Benjamin Franklin Award winner and a leading advocate of open access to data. He's been around IT for over 13 years, when he became a teacher and researcher in computational biology. Currently his team distributes an amount of scientific data equivalent to a quarter of the Library of Congress each month.

Philip Bourne is a big advocate of openness, both for source code and data. Especially when it comes to data gathered and code written with public money it is simply wrong to let it become proprietary. He believes openness has big advantages for any type of content as it promotes learning and growth and does not discriminate. Because of these ideas he got involved with the Public Library of Science, a project to free scientific data and papers.

Philip strongly underlined something he considers crucial for sustainable free and open development (be it code or data): a good business model. With that he is not talking about a monetary model per se, but at least something which rewards people and gives them a reason to contribute to keep things going. Without such a model, or rather a variety of models surrounding a community working on something, it will not stay the same forever. There is a huge risk of commercial interests taking everything over at some point - as has happened with most scientific communities. Fifty years ago code was freely shared, as was data. Magazines were cheap, and everyone had access to the data used in scientific research. Philip presented us with a graph showing the costs of science journals for universities over the last 50 years, going from almost zero to many hundreds of millions per year. It doesn't make sense if you think about it - the researchers who wrote the articles are paid by taxpayers to write them, and review each others' articles for free as well. Why do we pay so much for this?

Welcoming Dragon

The professor is now working with a variety of organizations and people to improve the way science is disseminated and comprehended. Philip believes that through openness we can increase the number of people interested in and involved with science. This way we can not just increase the amount of available data but also increase our knowledge development. As Phillips put it: "I believe open access, if broadly accepted, could change the way education works."

An important key to openness is licensing. Pubmed uses creative commons, requiring all data to be available freely online, unrestricted and in a variety of formats. Copyright will, however, remain with the author and if you use the data attribution has to be given to the original authors. An important goal behind this is the ability to do mashups. Following the popularity of Youtube, Scivee.tv, a scientific web 2.0 video site has been developed. It provides a mashup between video, papers, and other content. The author talks about his paper, and while he talks the site will show the content being discussed. And it works the other way around - if something in the paper is unclear, you can click it and the video will move to that point.

A more recent topic is the use of semantic tagging and linking, where a paper can become an interface to a huge mountain of related knowledge. Currently Philip is looking for ways of integrating semantic tagging in the process of scientific writing, as well as developing proper incentives to motivate the authors to do this.

Finalizing his talk, he reminded us of his three take-home points. Openness is not dependent on content type, it needs a sustainable business model, and it promotes new ways of learning and comprehension.

Distributing Free Software

The usual public

After the keynote, Helio Chissini de Castro and Will Stephenson took the stage to talk about the work distributions are doing.

The first talk was by Helio and covered the work distributions are doing. Lately co-operation is the big thing in the distro world - each faces the same or similar issues, and talking about that with each other simply solves them faster. A win-win for all involved parties. Distributions have always had a hard time working with upstream, and Helio simply points out that the only way to do it successfully is to be persistent, maybe even annoying at times, and keep the pressure on.

Helio then moved on to his thoughts about the future for distros. Desktops are not the only devices anymore. KDE is very well prepared for this - we've built an infrastructure to quickly build an unique interface - exactly what device manufacturers want. They don't all want the same interface but they can build something unique themselves. KDE gives them the tools to do this effectively, which provides a great opportunity for distributions.

Will - packager and developer at openSUSE - then took over, explaining why free software will be very successful and why we're in the right spot to be part of it.

He started with what distributions do - selecting software, deciding on versions, splitting up, integrating things, preconfiguring software, maintaining and stabilizing applications and finally packaging it all and adding their own infrastructure to manage it including hardware configuration and an installation tool. They triage and fix bugs for upstream. They also find and fix security issues. Distributions gather input from users and communicate it back to upstream, and work on legal issues and licenses. Last but not least, distributions provide services like multi-year support and maintenance and help with corporate deployments to their customers.

After an exhausting list of challenges you'll bump into while developing distributions, Will began painting on the whiteboard and going through the history of distributions and their work. He stressed that we have to work together with distributions, take each other's situation into account and work within the social web we belong to. The problem is that we can only do so much with limited resources, and our success up to this point has driven end user expectations higher and higher to the point where we will not be able to keep pace.

There are three potential solutions to this:

- 1. Reduce expectations
- 2. Stop working on this and surrender to the web
- 3. Turn more users into contributors

Of course we should focus on the third option - get more resources in and simply become better. The demands and skills are there, but we'll have to work on it and do more with them. This conclusion quickly turned into a discussion with the whole room on how we can bring in new contributors.

Relaxed presentation skills demonstrated by Will

Coding with patterns

Usability specialist Celeste Lyn Paul wanted to help the attendees to learn a bit about designing interfaces. She started to explain how we, as humans, follow and easily recognize patterns and those can be used by developers to make their applications easier to learn and use. A design pattern is a recurring set of widgets, functionality or interactions which are found throughout software and across the environment. Celeste showed a few examples of how we're doing this right, for example with search dialogs in Okular, Konqueror and Dolphin. They are not exactly the same but follow the same pattern, doing what it is supposed to do - helping the user comprehend what is going on and what the possibilities are. Patterns create consistency and we need more of that in our KDE software. Bringing certain patterns, pieces of functionality, in libraries and reusing it all over our software is what Celeste heavily advocates.

Celeste then moved on to giving a few examples of patterns she has been working on with developers. The first example was the Get Hot New Stuff interface, our easy interface to add online content to applications. She showed how the button for Get Hot New Stuff is different among many applications. Most show the GHNS icon and phrase the text in a similar way: "Get New xxxxx". But several omit the icon, and use different wording like "download more files" and things like that.

Celeste has therefore designed a short specification of how this item is supposed to look. That way, application developers can take advantage of the pattern to keep KDE applications consistent.

A second example was with rich information lists. She showed a number of lists found in various applications, went through the items on there and pointed out the differences and inconsistencies. Some have actions (which all look and work differently), font sizes differ, icon and descriptions exist in some cases but not others. Again she provided recommendations to promote consistency, yet allow for all the different requirements for all uses of a rich information list.

She finished her presentation with a call for volunteers, both to help work on identifying other scenarios where this happens in KDE software and also to work on developing reusable classes so that each new application which uses a similar design pattern (lists, GHNS, etc) does so in a consistent way.

KDE's youngest contributor

Free Software Jobs

After Celeste's talk Alexandra Leisse and Till Adam (barefoot!) took the stage. Their subject was discussing how participating in open source provides a unique opportunity to try new things and maybe even totally change directions in your career. They went through their own history, explaining the kind of strange places they started out (theater for Alexandra, rock music for Till) - ending up in Free Software. Both lives used to work out - but around 25-30 they decided they wanted something else and ended up at software consultancy (Till) and community management (Alexandra). They admit they never learned what they are doing right now - basically, as Till said, we just fake it... You need certain skills, of course. Like being able to deal with Divas, how to improvise, be disciplined and deliver under pressure. Be willing, eager actually, to learn. Not to be afraid of making mistakes. Be willing to wait and continue working on something until it finally pays off. Interesting thing (also true in this room): there are a lot of musicians in Free Software, even though these skills can be found in many other professions. All these things and more make Free Software contributors stand out of the crowd and be different. What is it that the drones lack? We as FOSS people can bring something to organizations. Things like a strong attention to detail - does this fit, is everyone on the same page here? We're not just willing to deal with but can fanatically embrace uncertainty - be flexible to an incredible extent. We also have a feeling for opportunities, we spot where we can go and what is possible. And finally, we often have pretty good people skills, especially for engineers.

Basically, Till argued that in Free Software you can be whatever you want. No matter how little or how much you know, if you put the effort in, you can learn and become something. It is a great place to grow and become more. When you enter and become a part of the Free Software community, you build up connections with a large number of people which presents an incredible number of opportunities. The whole community can not only help you learn and become what you want, it can help you get in contact with the right people, and provide a platform to present yourself and build up a reputation. You can present your skills and experience - and you should.

Till one-ups Will, and takes off his shoes

An important principle to keep in mind is the Halfling-Dragon principle: "If you find yourself in the company of a halfling and an ill-tempered dragon, remember that you do not have to outrun the dragon; you simply have to outrun the halfling."

Oh, and musicians are welcome at KDAB and Nokia!

At 15:30 it was Snack Time. Jeff had organized a variety of snacks, from potato chips and sodas to some nuts and dried fruit. Everyone slowly moved to the tables with food and drinks, while others went for the much needed nicotine.

Contributing as a non-developer

Justin Kirby then spoke about contributing if you're not a developer. He's been using KDE for many years but never really figured out how to help, as the typical path to contribution is writing code, which was not something he was interested in. At Akademy in July 2009 he finally became involved in marketing. He gave a quick overview of what areas you can contribute to without coding, and how it helps - from taking care of things a developer would otherwise have to do to simply adding something which would never have been done if you didn't do it.

Justin then went through six different types of jobs that anyone can do for KDE without knowing a single thing about writing code, and then provided specific places to look and people to contact in order to help out.

He started out with documentation, explaining that anyone who uses KDE software is capable of helping out with adding or updating content in the KDE Forums that cover the basics. He then talked about how you can help developers out by filing bugs when you find them and by getting involved with bug triage. The easiest way to get started with this is to participate in a current Promo tasks to be done on their wiki page. All of these things are something that any KDE user could easily help with in their spare time.

Justin espouses contributors, contributors, contributors

He then talked about how people with expertise in areas like graphic design or artwork could become a KDE Artist. This team helps out regularly with designing logos, icons, themes, and things that help shape the identity of our software. For those who have experience in usability or interface design there is also the KDE Usability Project which aims to provide feedback to developers on how to make their applications more usable.

Last but not least, he talked about how all users should help out with supporting their fellow community members. He mentioned that this is often somewhat distribution-specific but that everyone can help out with answering questions on forums, or share tips on using KDE where possible (blogs, microblogging, etc). He stressed that helping users fix problems is what ensures they continue to be a user of KDE and hopefully a contributor at some point as well.

Marketing in the KDE community

Your author presented the KDE marketing efforts, explained what they do and how the team currently works. He proceeded to talk about the many challenges the team faces and the opportunities there are to do more and better. The talk finished off with some discussion and the usual call for getting involved. Many of the plans and ideas can be found on the wiki - a good starting place for those interested in joining the team.

Plasmalicious user experience

Artur Duque de Souza ended the day with his talk about users and what they want from software. The answer to that, as he described it, is that people just want to get things done and they don't care how it happens. They just want it to be simple to get the job done. For this reason, developing software is all about user experience - but this experience has to be designed differently when creating KDE Software for netbooks and mobile devices.

Artur in action

The work by the Plasma team on getting the KDE Desktop ready for a variety of user interfaces plays a big role here. Working on the netbook interface the plasma team learned you have to create different UIs for different devices in order to continue to make life easy for the user. Another thing they learned is that some applications simply do not work on a small screen - they have to cram too much information on the screen. If KDE software moves to an even wider range of devices, work has to be put into different UIs for different devices. However, the approach of the team, starting with design and then moving on to coding has been paying off, and Plasma is quickly becoming a key player in the workspace market.

In the end, the big thing we learned is that Qt Everywhere pretty much means KDE Everywhere. But we need to focus on the design differences that arise on all these mediums and create a great user experience on all devices!

This talk concluded the day. Everyone then went off to either get some sleep (it was a tiring day) or have food and drink and talk a bit more. Details of the following days will be posted shortly.

Day Two

By: Jos Poortvliet

2010 21 Jan

The second day of Camp KDE was filled with many more interesting talks. This day's talks were of the more technical nature versus the first day, and the KDE team took notes. As usual, the talks were recorded and videos will be available soon. A detailed rundown of the second day of talks are behind the link.

The second day started with a keynote by Frank Karlitschek about KDE vs. The Cloud(TM), looking ahead a couple of years into the future. Frank pointed to the trend to use cloud based applications instead of desktop applications. The question this raised was: can KDE could provide a superior experience if we properly combine the advantages of cloud based technology with the freedom of open source and the power of desktop applications?

He began by defining "The Cloud". Chrome OS and Google's huge variety of cloud based services in general are the most pervasive examples, though it has also become more common in multimedia thanks to services like Flickr, Pandora or Last.fm. Frank went on to present the results of a survey he did on the advantages of the cloud. The most important advantages of cloud apps are the social features, the easy access to apps and the ability to access your data from everywhere and share stuff with others.

Is the growing reliance on web applications and online services a good trend from the FOSS perspective? And where will KDE software be in 10 years if it continues like this?

Frank points to the things Google has been doing with and on the web, and the dangers such a successful monoculture presents. Google is definitely behaving itself right now, but things might be very different in 5 or 10 years.

And we had a visit from another dragon!

Frank then presented the results of a survey on disadvantages on the cloud. The biggest problems for users are possible privacy and data protection problems and the dependency on third party infrastructure.

The big question of course is: What can be done about it? How can we take back the web? As the KDE community is a technical bunch, we are naturally focused on technology. We cannot and will not fight Google by sending lawyers at them or starting a political movement - we must create something cooler and better. So how can we get the best of both worlds and create a compelling alternative? He outlined the following requirements to compete:

- provide a rich desktop with rich applications instead of web apps
- be social
- ease deployments
- ensure data you own accessible everywhere
- facilitate mashups

So for this we need 3 building block:

- social services
- online data sharing
- easy deployment of applications.

Frank now introduces the first part of a solution, the Social Desktop and libattica. Libattica is an easy to use Qt-only library which provides social and transparent data access for any application with build-in authentication, error handling and support for different data providers. Libattica is currently used in the KDE SC 4.4 desktop by a variety of applications like the social desktop widgets. This makes adding of social features to apps really easy.

Frank also released a reference server implementation so that other online services can easily integrate with the KDE Social Desktop. The goal for this is to not only support the OpenDesktop provider as social network but also other unrelated networks like Facebook. The social desktop can then combine all the networks in a single experience, and in time even help in getting our data back.

A second technology is GetHotNewStuff which has recently been heavily reworked and updated. In time, the GHNS framework can also help in the deployment - both for data but, with the integration with the openSUSE Build Service which is in the works, also for applications. The future even holds integration with Qt Creator, where a developer writes an application, does a few mouse clicks to upload it with GHNS where the openSUSE BuildService (or some local install of the Build Service software) takes over and creates packages which then anyone can easily install.

The third needed building block is a technology to make storage, access, revision control and sharing of documents really easy. Frank announced the ownCloud project which is a personal cloud storage solution to manage all you personal data. ownCloud will use the AGPL license so everybody can install his/her own cloud storage on a server, desktop and anywhere else. Everybody has control over his/her own data but is still able to access it from all devices, have revision control, automated backups, sharing with others and data encryption.

So let's make this happen - there is a wiki page set up and a gitorious repository if you want to help. They are still in an embryonic stage but the future of the social desktop integration into the KDE projects is bright.

Ian talking about Telepathy

Once Frank was done, Kris Moore took over to talk about KDE on PC-BSD, a desktop distribution of FreeBSD. Kris goes through some recent developments in PC-BSD, including the development of a software management tool and a variety of small tools building upon KDE technology. He explains the PBI package management, which works based on self-contained binary installation packages which still integrate with KDE.

He explains why PC-BSD makes use of KDE applications and technology. The short answer is they love it - the long answer would involve the professional look and feel, the great technology and the continuous innovation coming from the KDE community. Of course there are issues the developers bump into - something Kris does not skip on. The many new dependencies, in part introduced due to the cross-desktop cooperation, make the work of packagers harder. So do the 'Linux-isms' developers introduce while working on KDE software.

The questions revolved around the different and unique way the PC-BSD team works, including: how they keep applications in their own folder at all times; and how each package can be so self-contained that pretty much the whole OS can be upgraded, yet the apps keep on working as they were. This has side effects such as increased RAM and harddrive storage usage. The team is looking for solutions for this and are really creative in their efforts to find a solution. An interesting point which came up is that due to the similarity between Mac and BSD, the people working on Mac benefit from the PC-BSD work and vice versa. All in all, the PC-BSD team and their work is something to keep an eye upon.

Kris was followed up by Ian, who talked about Telepathy. He explained why Telepathy (an Open Source real time communication framework) is needed, what it aims to do and how it works. Based on D-BUS it can easily be used from within KDE, and Ian gave a quite detailed description of how Telepathy works.

Romain on Windows

After lunch, Romain Pokrzywka told what he referred to as "the horror story of KDE on Windows." According to him it is alive and well - and yes, he runs Windows 7 with KDE Applications on it. He demoed a few of the applications, including a few crashes and error messages. The mixing of Windows and Plasma widgets was pretty impressive as most of the things do indeed work pretty well. After the demo Romain spoke about the history of the KDE-on-Windows efforts to give us some background on where it came from before talking about where it is going. We were introduced to the heroes of KDE-on-Windows and Romain explained how the development work currently goes. A few of the things Romain demoed are pretty new and impressive - Nepomuk and Akonadi, for example. The KDE PIM demonstration results in input from Till Adam who makes everyone anxious for KDE SC 4.5 - while 4.4 isn't even released.

After the second demonstration Romain moves to what the team is currently working on. Solid is heavily under development, and Strigi development still has to be started. Furthermore, a few things are not planned at all. Among those are KWin and the Plasma taskbar and systemtray. KWin isn't even a technical possible, and the taskbar and systemtray would be possible but are very difficult and are simply low priority. Windows has a desktop shell, and while Plasma is very nice, it's not crucial for a good experience for KDE applications on Windows. Romain himself does like to use Plasma, purely as a desktop replacement. Several of the widgets really offer interesting functionality, especially folderview.

Romain then lifted the lid on a few future plans like support for more platforms (64-bit and WinCE), more applications, better integration and per-application installation packages. On the integration front, it was concluded that applying a random style to each application when it starts would make KDE applications feel more at home in Windows, fitting in with the huge variety of styles on the platform. One day, the KDE-on-Windows team would hope to play a role in the world-domination plans of the wider KDE community.

Marcus Hanwell was next with a talk about CMake and improving the development process. After a short introduction of Kitware, the developers behind CMake and their other technologies, he showed lots of colorful and pretty pictures of the scientific work he has been involved in. After this Marcus went into the details of the development process Kitware has set up using CMake, CDash, CTest and the other tools they use. By using extensive automated testing, the development process can become faster and leaner, thus allowing developers to focus on writing features instead of fixing bugs. CMake makes it possible to work much easier with cross-platform C++. The adoption by KDE has made a huge difference to the spread of CMake, and these days it is really pervasively used. Marcus detailed some technical features of CMake and then moved on to giving tips, tricks and examples for how to use CMake. After the extensive CMake introduction Marcus continued with in-depth CTest information, revealing the brand-new and still experimental git support. Finally, there is a call for interested developers: Kitware is hiring, looking for Qt knowledge!

Leo Franchi then gave a talk about Amarok. He started by showing a variety of media players, showing how all the applications seem to focus on sexy and good looks, copying iTunes-like functionality, showing big covers etc. He points out how all these applications try to be intelligent but often fail at it. And of course the applications almost all try to push as much paid content to you as possible. While Web 2.0 can be can be cool, somewhere money has to be made. Having all your music streamed to you for free - that's not gonna happen in the long term. Integration of offline content and online is key here, as you can see with the business models of pretty much all other music services out there.

So what is the problem a good media player should solve? First of all, there is a lot of music in the world. So what does the user want to hear? There are services out there which have been working on this, like Last.fm, and there are a lot of ways to visualize and calculate similarity, and some of these smarts should be put into the apps. The dynamic playlists in Amarok are a way to let the user take care of most of their music tastes themselves, but the team is working to introduce a variety of 'intelligent' biases using data from Last.fm and other services. The information is out there - the challenge is to find and integrate it. Other desktop players fail at this and it is something the Amarok team wants to get right.

Leo introduces Playdar, a server tool which runs on the background and can get you the music you want - finding it from wherever it can. It will play the song from your collection if you have it; play it from a friend which has shared his or her files if they have it; or find it on the world wide web, say from the streaming service you're subscribed to. Work is being put in integrating this technology as well. Leo's topic solicited a lot of responses and discussion, which kept the team busy for a while longer.

After the snack break Caio Marcelo, a developer from INdT gave an introduction to QML, a new language to write user interfaces for Qt. QML is simply a way to describe trees of objects and properties which together describe the user interface. QML allows you to embed ECMAscript (javascript in Qt) as well to perform simple transformations. QML is heavily inspired by Qedje, the user interface language developed by the Enlightenment developers. QML allows you to use animations and effects and describe regions around objects which respond to mouse input. Caio demonstrated a variety of examples based on QML and shows how easy it is to create dynamic interfaces using this declarative language.

To explain the concept of the declarative UI, Caio gave a few analogies:

- EFL: "QML is a kind of Edje for Qt"
- Web: "QML is a kind of HTML+CSS"
- Qt: "QML is a kind of .ui file from QtDesigner"

You can use QML in canvas-based applications like Plasma and mobile interfaces. You can even write small applications like Plasma widgets entirely in QML. It would also be a great candidate for a new UI for KDM as it can give themes a lot of flexibility. The power of QML is to allow a sort of advanced themability - themes can change the interface entirely so you can have a single application with several QML 'themes' for different devices from large screens to mobile devices. Graphics, usability and interaction designers can work much more closely with developers, changing interfaces directly. Declarative UIs can bring a variety of new features like easier network transparency to UIs, and the whole range of possibilities is still unclear.

Marco Martin proceeded with a talk about the Plasma Netbook interface. He listed 'the competition', explaining how they work and what advantages and disadvantages their approaches had. Introducing a number of insights, Marco showed how deeply the netbook team has thought about their user interface. The requirements basically boil down to creating an interface with as few elements as possible to get the task the user wants to do done as easily and quickly as possible. However the team had more in mind than just creating this interface - it was also important to make as few assumptions as possible about the form factor, creating a very flexible solution. The goal was to push as much functionality as possible in the underlying infrastructure, to assure future interfaces like for mobile or multimedia devices would be even easier and faster to develop.

Marco started to present the various parts of the netbook interface like the Search and Launch interface and the Newspaper Activity, describing how they work and what assumptions the are based upon. While going through these items he shows how they look and work.

Looking at the future, it is clear that what KDE SC 4.4 brings is just step one. The look and behavior can still change, and there are many new widgets needed to complete the experience we want to offer: email is coming but appointments and other web services are still needed. Furthermore it is a challenge to scale up and down for widgets, and this is something which still needs investigation. But more and more tiny devices are coming on the market, and the work on this will prepare the KDE technology and community to provide an innovative and creative alternative UI for them.

Chani on stage

Chani Armitage took the stage with the last presentation of the day, with the work of the Plasma team on Activities as main topic. She started out by demoing a few of the Activities she uses - which is a lot. So what is an Activity? An Activity is what you are actually doing. If you work on a math task for university, you have a bunch of windows like a PDF file with the assignment, a calculator and some other things. These things belong together but there is currently no organization: we have to manage them ourselves. Chani wants to organize this in a better way. We now have implemented the Desktop Activities, but she is trying to associate windows to a desktop as well. You should be able to save activities and open them on demand. And this not only to make them easy to manage but it also saves resources. More importantly, applications should be able to react to your activities - KMail should show email folders relevant to your math course, the file open dialogs should have relevant shortcuts etc.

This is going to be a challenge. The current infrastructure does not support these things - changes will be needed in many places, including the KDE Applications. For example saving sessions and opening them will be hard to get working in all situations - say an application is running in two Activities? Hopefully a new cross-desktop standard can emerge from this. It is not all grim, however. Nepomuk already has experimental functionality to tell applications what activity is currently running so they can customize their content.

Chani has already implemented what she refers to as 'a horribly ugly hack' which allows closing and re-opening activities in Plasma but this is just a very first step.

The third day will have more talks but those will be even more technical. All in all, the author is enjoying the conference. As usual it is good to meet old friends and catch up with them, meet new friends and learn what cool stuff those are involved with and to generally hang out and have good conversations.

Day Three

By: Troy Unrau

2010 21 Jan

The third day of talks at Camp KDE was somewhat shorter, due to the afternoon Cmake training provided by Marcus Hanwell of Kitware. However, in order to provide complete coverage of the talks for the readers of the dot, summaries of the third day's technical talks are provided within.

Stephen Kelly: Akonadi Models and Proxies: Tips Tricks and Hacks

First talk of the day was by KDE PIM developer Stephen Kelly. Akonadi is a generic interface to access your notes, email, and other PIM data. Stephen walked us through the model view concept, the features of the Akonadi model-view, some challenges the developers faced during development and how the rest of the KDE community could take advantage of Akonadi.

A simple way to share PIM information between applications is to simply code the apps to directly communicate with one another, as was done in KDE 3 times. For KDE SC 4, the Akonadi framework was introduced as a central, pluggable storage interface. It ensures that there are no conflicts (for example, two applications trying to save data simultaneously to the same file or online service), and allows access to the data without having to start the applications.

To present the data within the applications, Akonadi provides a QAbstractItemModel which can be directly linked to a view within an application such as KMail. Stephen demonstrated a number of models and views, including the EntityTreeModel, which represents all data provided by Akonadi. Applications like KMail will want to restrict the displayed information to provide a good user interface, rather than a dump of all the information available.

Qt training in progress

Stephen then walked us through a number of KDE classes which extend the functionality of Qt to provide an easy way to select and filter for the desired information. He gave examples from KMail and KOrganizer. He demonstrated the advantage of the model-view architecture by showing that email could be displayed using the category view from SystemSettings. Lastly, he demonstrated using Nepomuk and Akonadi in conjunction to sort PIM information by tags.

Finally, he walked through a number of the test applications and processes in place to ensure that everything functions as expected. There are hundreds of such tests, and there is also a dynamically scripted test application to observe the behavior of the proxy models.

Will Stephenson: Spreading KDE with the openSUSE Build Service

Will returned to the stage for a second talk this week to discuss the openSUSE Build Service. His key message was: the build service is for everyone. It is composed entirely of free software using GPL and similar licenses and supports a myriad of distributions. Since the openSUSE hosted version only allows free software to be built with the service, other companies have been downloading and installing a local instance of the software for their own build farms.

The build service is more than just a build farm – it is a software repository that builds for numerous architectures and distributions, complete with dependency management and package signing. It can publish the results in distribution repositories to make downloading and installing seamless to their users. It is used not just meant to build single packages, but can be used to build entire distributions. During the openSUSE release deadlines, it is rebuilding the entire distro nearly every twenty-four hours.

Will introduced us to a number of clients for using the build service. There are web clients, python based command line tools (similar to Subversion) called OSC (openSUSE Source Commander), and a mono-based user interface. Will gave some examples, showing how to build KDE SC 4.4 RC2 for Fedora 12.

Will then presented a prototype help tool called "kde-obs-generator". It takes a tarball as input, scans it, determines if it is a Plasma Applet, a KDE Application, a wallpaper, and so forth. The output includes the required RPM spec files, or Debian control files, and can be very quickly submitted to the build service. He demonstrated this for the application kvkbd.

Recently, connections have been created directly for kde-look.org and kde-apps.org. These packages are now being automatically created giving access to pre-built packages for users of several distributions.

Jeff Trull: Qt + Salesforce

Jeff Trull talked about Salesforce.com and his work on integrating their services into Qt and KDE. In principle, the site is a typical modern web 2.0 website, consisting of a bunch of tables, queries and generated displays. They provide SOAP and REST APIs to the outside world.

For using SOAP from free software, the common options are Apache Axis2, QtSoap, and gSoap. Salesforce examples were provided only in Java and Objective C. gSoap provided mixed results, but those were sufficient to get to the next step: providing Salesforce data as a Qt Table model. This included lazy data transfer, handling of untyped data, and provide a reasonable fit for the existing data.

QAbstractTableModel was the first consideration, but QSqlTableModel ended up being the best match. It understands tables and authentication, lazy fetching and varying data types. Fortunately, Salesforce data could be accessed through a SQL query. Finally, QSqlQueryModel was chosen, and Jeff created a new SQL driver based on the SQLLite driver.

Jeff showed an example of a table view that pulled contact information on the fly from Salesforce. The next step was to create an Akonadi resource so that the contacts could be directly available within KDE. He showed the contacts showing directly from KAddressBook.

In the end, the discussion centered around creating or fixing Akonadi resources for other websites and services.

Developer food

David Wonderly: Documentation

After breaking for lunch, the conference continued with a short talk about the current state of KDE documentation by David Wonderly. He had this comment about documentation: “It has no use if it doesn't answer the questions that people ask.” He uses this to show that the docs should be mirroring the questions being asked in email, IRC, or other KDE support channels.

He provides a number of tips. Excessive documentation is worse that bad documentation – keep it simple, avoid tl;dr. This helps to maintain the document over time as the software is changing. Documentation needs to be well outlined, and clearly flowing. Avoid processes that take more than ten steps, and if it becomes more involved, break it into smaller walkthroughs. And lastly, "avoid fluff." A long and complicated sentence is difficult to maintain and translate. He gave a few examples of sentences that can be rewritten for additional clarity and ease of translation.

Adenilson Cavalcanti: Plasma New Animation Classes

Adenilson of INdT provided an overview of the animation framework in place in plasma libraries, integrated into Software Compilation 4.4. This was made possible by the introduction of the Kinetic classes in Qt 4.6, which allow for more realistic animations.

In one month of work, proof of concepts were developed for the Kinetic scrolling list, drop shadows, stacked rotations, fades, and a pulse effect. Now we're four months, 180 commits and several refactorings by a group of five KDE Plasma hackers later.

After having been distracted by pretty photos of Brazil, Adenilson found his second set of slides, and continued with some live demonstrations of a number of animations. These included examples of the animation classes running on phones and netbooks, showing that these animations can easily run on devices low on resources.

He showed a number of code snippets and described how to add the new animation classes in a few lines. This was followed by examples on how to update current plasma widgets to take advantage of the new animation capabilities, with the dictionary widget being the featured example. He handed around a phone to show off the animation examples.

Adenilson Cavalcanti: KDE-PIM Google Sync

In an Adenilson double feature, he returned immediately to talk about implementation of KDE-PIM support for Google Calendar and Contacts using Akonadi in KDE 4.3. This is related to the previous talk given by Jeff and is one of the multitude of data providers being created for Akonadi.

He showed a video of contacts being added in KAddressBook just like a locally stored contact, then he logged into Google Mail to find his contacts there, including details and pictures. This was followed by another video showing a calender item being created which was seamlessly synchronized with Google Calendar. The total of this code base was over 700 commits, and 7000 lines of code. Before implementation, this was one of the most highly voted feature requests on the KDE bugtracker.

The backend is implemented using Google's REST + atom (HTTP + XML) APIs. Authentication requires HTTPS post to get an authorization token and session id. Examples of the protocol were demonstrated using screenshots from Wireshark, a network tool used to inspect traffic.

He then went through the design of the Akonadi plugins required to get Google synchronization, including issues with determining what has changed on the Google servers, and reverse engineering of the protocol.

This concluded the third day's technical talks at Camp KDE, although scheduled training sessions for CMake, Qt and Qt/Embedded happened later in the event. These trainings were of particular high quality according to the comments from many attendants, and will continue for the remainder of the week (with time for coding in between).

Licenza di questo ebook

Tu sei libero:

	Di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera.
	Di modificare quest'opera

Alle seguenti condizioni:

	Attribuzione — Devi attribuire la paternità dell'opera nei modi indicati dall'autore o da chi ti ha dato l'opera in licenza e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
	Non commerciale — Non puoi usare quest'opera per fini commerciali.
	Condividi allo stesso modo — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.

Reportage dal KDE Camp 2010, l'incontro americano annuale della community di sviluppatori di KDE (un un ambiente desktop grafico per postazioni di lavoro Unix usato in molte versioni di Linux).

Da questo tipo di assembramenti di GEEKS e NERDS prende forma il futuro di Linux.