Introducing Project Timelord

Through intense self-reflection, it has come to the attention of several Kubuntu developers that due to major architectural changes in the software stack, the usage of certain Ubuntu technologies and a small pool of human resources that Kubuntu is not currently reaching its full potential. We have realized that deep changes must occur, and in the spirit of the Timelord known as the Doctor, we have founded a project dedicated to his heroism and pure awesomeness. We have done this in the hopes that, with the help of the community, we can make these changes happen. Kubuntu will look into the Heart of the TARDIS and be reborn.

Project Timelord aims to establish a solid base for Kubuntu to build upon that will last the lifetime of a Timelord. (Including regenerations.) To accomplish this, we have compiled a list of improvements and changes which we feel need to be made to Kubuntu, as well as a roadmap for implementing them. The roadmap outlines both immediate and ongoing technical remedies to improve developer efficiency and the overall quality of Kubuntu, as well as plans to promote the growth of both the developer and user support communities in the long term.

Identified Issues

Translations

The goal of the Launchpad Translations project (also known as Rosetta) is a noble one. In a world of many third-party software projects, not all take the same care to ensure proper localization. To provide a consistent level of localization, Launchpad admirably tries to create a single online interface where translations for any application can be

improved upon and entire applications can be translated into a larger variety of languages than the application developers themselves may support. Launchpad Translations also aims to give Ubuntu a unified interface to translate Ubuntu-specific modifications in applications.

Through the valiant combined efforts of several Kubuntu and Launchpad Translation developers, most of the bugs where Launchpad Translations breaks upstream translations have been fixed; as far as we know. Unfortunately things are still not perfect, and until they are upstream has a very good reason to be at the least annoyed at us for mucking with their translations, and we will quite validly retain the reputation of a distribution with sub-par translations. Given that one of the goals of Kubuntu is to great translations so that Kubuntu can be used by anyone, this is a very important issue.

Furthermore, it has been found that the current levels of Kubuntu developer and translator resources are insufficient to reap the intended benefits of the system, much less fix what goes wrong. Kubuntu modifications to applications do not get translated in most of our supported languages due to a virtually nonexistent Kubuntu translation team. This lack of resources also leaves us unable to perform quality assurance on any languages other than the one or two major languages translations-minded Kubuntu developers can keep an eye on.

A great part of the reason we do not have Kubuntu translators is that in the past translations have been abysmal, depsite the hard work of the Kubuntu translators. We lost entire languages-worth due to this. The other part of the reason is the Launchpad Translations interface itself. Through chats with KDE translators on the KDE translators mailing list and elsewhere, we have found that the use of Rosetta is a major dealbreaker for getting them to help with our translations. One thing is clear; to get a translations community we must make it as convenient as possible for existing translators to contribute, and currently translators have expressed their disinterest in the Launchpad Translations interface.

Using Launchpad Translations also presents a burden to packagers. The current setup needs lots of attention to keep from breaking, and also presents an unnecessary difference from Debian KDE packages. The setup also makes things harder even for Universe packages that aren't translated in Launchpad, as the current setup can cause builds to fail if the package does not provide a Messages.sh script.

Marketing

Kubuntu basically has no marketing; or at least none compared to Ubuntu or even any other major distribution. Our current branding policy is unclear, and when we do give out information every release it is in a non-human-readable (geek) lump of technical information that the average user could care less about. Currently, LoCos are the only venue of promotional work we have.

A mission statement and vision are two other things that we currently lack.

Proper marketing for Project Timelord is an integral part for improving our image as a distribution.

Software

There are several closely related areas that need attention in the realm of Kubuntu's software offering. These flaws include how we handle bug reports, actual flaws in the packaging of upstream software and the integration of Kubuntu-specific applications with the surrounding system.

Handling bug reports

Kubuntu, like any software project, receives many bug reports each day. Most of the bugs reported are actually bugs with KDE itself. Currently, all crashes during the development cycle get reported to the bug tracker at Launchpad. As of Kubuntu 9.10, the “Report Bug” feature of every KDE application will launch a bug report assistant that will file a bug in Launchpad. In essence, all bug reports must come through us. Due to the limited amount of people to sort and triage these reports, we run into several problems. The limited amount of triagers must be curt with their responses if they are to get around to sorting all of the bugs. This leads to an unfortunate deficiency in politeness, helpfulness and speed with which bugs are sorted.

Faced with a large amount of unsorted bugs, developers have a hard time getting a reading on what needs to be fixed. When bugs do pop up with our packaging, they get lost in a sea of upstream bugs that are being tracked at Launchpad. This leaves issues open for longer, sometimes remaining untouched until somebody complains hard enough in venues other than the bug tracker, that things finally get done.

Packaging flaws

While our KDE packages are good overall, (Translations aside) there are a number of ways that our KDE packages are not perfect. There is a general lack of Quality Assurance on uploads to both development releases and on backports of KDE packages to stable releases. File overwrite errors are too common in both cases. As a potential issue, we also have a collection of patches for most of our core KDE patches. These patches may or may not be of good quality, depending on the patch.

Another issue that seems to pop up is that some KDE modules are not compiled with all the functionality they are capable of. This is usually due to the fact that this functionality may require packages that have not been reviewed and approved by the Ubuntu security team. These tend to be forgotten about too easily.

Kubuntu application integration

Kubuntu introduces several Kubuntu-specific tools that work to bring a more usable desktop experience to the user. These applications lack polish, however, and usually are not integrated well at all with the surrounding system. Many of these applications seem to mostly be ports of Ubuntu tools, which tend to be dialogs launched from the GNOME settings menu. These tools are launched as separate windows from the host application, causing needless clutter. Other applications are written in languages which are not not suited for the purposes the application serves. Examples include the use of memory-hungry python applications for non-frequent notification purposes.

User-developer Interaction

It is well-known that the gap between the user and developer is not small. As it stands, there are many places to file bug or wishlist tickets, including the Launchpad bug tracker, KDE's bug tracker, or Ubuntu's brainstorm tracker.

Launchpad's interface is also somewhat complex. (This is somewhat alleviated by apport)

Developers, as a whole, usually do not communicate in the same venues that the user does. (For example, forums such as the Ubuntu and Kubuntu user forums) This leads to a general ignorance of what the user currently thinks about Kubuntu, or ideas the user might have to help improve it.

Recruiting

Sufficient resources are something that Kubuntu currently does not have all across the board, from development to user support. This may be in part to the lack of a solid volunteer recruitment program.

Solutions

Translations

For the reasons mentioned above, the Project Timelord team has deemed that in Kubuntu's current state, using the Launchpad Translations system is counter-productive to achieving the goal of a localized, human KDE experience. During the Kubuntu 9.10 development cycle, a concentrated, cooperative effort between Kubuntu and Launchpad developers occurred to try to improve KDE translations in Kubuntu. Great improvements were made to this point, and while we appreciate the time and effort the Ubuntu Translations team has taken to improve the current situation for Kubuntu, we must face the facts that given current developer and translator resources that Kubuntu is unable to use this translation architecture in a productive manner.

Fortunately, KDE offers localization of excellent quality for many languages. We advocate the use of KDE's translations, abstaining from modifying the text of KDE applications until we can recruit and build a Kubuntu translations community that can translate these changes properly.

To recruit a new translations team, we will announce the indefinite postponement of the use of Launchpad and translate Kubuntu applications in their bzr repositories, generating PO templates for translators to use with their favorite translation applications. Once a translation community begins to form and grow, we can consider re-allowing a conservative modification of strings in the core KDE modules, with most string changes being discussed and pushed upstream rather than changed early in Kubuntu. We will then generate a single PO template for these strings, which will then be added to the Kubuntu-specific PO template as required.

If in the future it is the consensus of the translation community that their job would be better served and made easier by using the Launchpad Translations interface for translation work, we will re-evaluate the technical implications of once again using Launchpad Translations. Since the translators are doing most of the work, we want to retain them at all costs. The ultimate decision will be left to the translation community. Forcing a system that nobody wants will be more detrimental than not using the system at all, despite the technical benefits it may bring.

We do realize that switching translation systems is no small task. Therefore we will experiment with the changes during Kubuntu 10.04, and will chose the path that will yield the greatest benefit for Kubuntu 10.04 in the short run. Even if we end up not being able to feasibly switch from Launchpad Translations for Kubuntu 10.04, the option will still be kept on the table for the future.

Marketing

The development of a promotion program is required here, as none currently exists. Aside from this, how we currently present information to our users on our website should also be reviewed.

Branding

We need to develop and implement a clear policy on how much branding we want to apply to Kubuntu, and take the necessary steps to ensure we have the artist resources to make this happen. We should discuss modifications of upstream artwork (not replacements) to the upstream art team before implementing them in Kubuntu, no matter what policy we implement.

Promotion

Announcements on our website need to be examined to ensure that they are human-readable. We should consider making a set of news templates that we can fill with information, as well as reconsider how we distribute information and instructions about our PPAs.

It would be nice to get more coverage in magazines. To accomplish this, we need to create a unified source to publish information that can be used in magazines or other publications, printed or not. This way information isn't scattered between kubuntu.org or various blogs.

On the front of LoCos, we should encourage that promotion be done through them. Kubuntu LoCos should be nutured as a goal to reach this end. To make their jobs easier, we should provide them with stock graphics, et al to make promotion work as easy as simply interacting with people.

Product Management

While we may have a general idea of what we want to be, we should draft a concise mission statement and vision statement. Developers can use it as a general guideline of what Kubuntu should be developed to be, and users can use it to make sure we fulfill the misson statement, should we begin to stray.

As part of Project Timelord, we need to publicize an easy-to-read roadmap outlining Project Timelord. Promoting Project Timelord (followed by properly implementing it) are vital to improving the current image of Kubuntu. Improving Kubuntu's image is the key to increasing volunteer resources in all areas of Kubuntu. A proper roadmap will also help us pace ourselves, as to not burn ourselves out by trying to fix too much at once.

Software

Handling bug reports

The conclusion that Project Timelord has come to is that currently Kubuntu does not have the resources to handle upstream bug reports and wishlist items. Until a proper bug triage team can be recruited, we should revert the "Report Bug" feature to open bugs.kde.org in the user's web browser rather than Launchpad. Packaging issues and severe upstream issues that need a resolution quicker than the next KDE release may still be reported via the ubuntu-bug application. In addition, we should disable automatic crash reporting during the development cycle, instead using the wonderful Dr. Konqi bug crash reporting wizard KDE uses. This will make all crash reports go to KDE rather than to Launchpad.

We must also take steps to clean up the upstream reports currently in Launchpad. Project Timelord advocates a mass closing of all low priority upstream bugs, politely asking the reporters of bugs that haven't been linked with reports at KDE to file a report at KDE's bug tracker, or upstreaming them ourselves when appropriate. Low priority, easy-fix bugs may still be reported as papercuts to the One Hundred Papercuts under this system, continuing the Ubuntu tradition of providing a useful desktop experience.

This will increase the efficiency of the currently-limited bug triaging team. If recruitment is successful and results in the eventual buildup of a first-class bug triage team, we may consider tracking upstream bugs in Launchpad.

If it is found that we constantly find ourselves in need of pre-made responses for certain types of bugs, we may consider composing a nice set of boilerplate responses. These responses will be polite, yet informal enough to not come off as extremely stiff and automated. They should be tailored specifically towards the common issue at hand. Possible implementations may include a random response generator so that all responses are somewhat unique, but this is not a requirement as long as the set of responses is polite and down-to-earth.

Packaging flaws

To solve the problem of all-to-common overwrite errors, a concerted effort must be made to have multiple people (rather than one or two people) test upgrades of KDE packages, even if it means delaying the release of the packages for a bit. Scripting an automated upgrade tester should be looked in to as a way to test for such errors with more efficiency.

To make sure that KDE software comes with its full potential enabled by default, a review of all optional dependencies should be done on all core KDE modules. The results

will be tracked on a wiki page, where we can make a list of software that needs approval by the Ubuntu security team. Developers will have an easy list to remind them of what needs approval reports. It would not hurt to do this review every major KDE update, some time between the initial packaging of the betas and Ubuntu feature freeze.

We must also review **all** of our patches in **all** of the core KDE modules, checking each for their necessity, sanity and quality. In addition, we should check to see if any are upstreamable, and upstream the patches accordingly. Patches that fail to meet sanity/ quality/necessity checks will be eliminated.

Kubuntu application integration

Improving the integration of Kubuntu applications with the surrounding system will likely be the part of Project Timelord that will take the longest. In many cases this will involve departing from the base of applications hastily ported or created after the switch to KDE 4, replacing them with well-designed replacements that will be easy to maintain as well as future-proof. We will also try to depreciate Kubuntu-specific applications that duplicate functionality implemented in upstream applications we can use, lessening the number of applications we have to maintain ourselves. All default applications will be re-evaluated to see if they are still the most appropriate choice for fulfilling the goals of Kubuntu.

These changes will be spread over as many development cycles as is sensible. Final implementation specifications can be worked out during the planning session for the cycles they are worked on. The list of proposed changes, in not particular order, is as follows:

- The install-package application and it's dependency Gdebi-KDE should be removed, with their functionality being replaced by KPackageKit. KPackageKit accomplishes most of what install-package does currently, and only needs a few tweaks to replace the batch installation functionality of install-package. Gdebi-KDE is pretty much unmaintained, duplicates work.
- Eventually we want to either write a custom System Settings module to replace software-properties-kde or patch missing functionality into KPackageKit. The current situation of it being launched as a separate window with kdesudo is undesirable. Will require work with Ubuntu to make software-properties itself PolicyKit-capable.
- Eventually, we will ditch language-selector-qt as a standalone application. We will either replace KDE's Locale & Language System Settings module with a custom module or patch language-selector's functionality into the existing one in a way that makes sense. If discontinue use of the Launchpad Translation system, language-selector will be too tied to the Launchpad Translations architecture to suit our needs anyway.
- Consider rewriting printer-applet-kde in C++, if feasible. We should port it away from KSystemTrayIcon to KNotificationItem at any rate.
- Rewrite update-notifier-kde as a C++ kded module, which would use persistent KNotifications to notify of reboots, upgrade hooks, and other update-related items. The project will be renamed to kubuntu-notification-helper, since it no longer actually presents package update notifications. Creating a python helper program for distribution upgrades which will hook in to the current update-manager-kde utility will be investigated.
- All Kubuntu applications not being replaced will be made compliant with the current draft of the KDE Human Interface Guidelines, and receive user interface tweaks to enhance usability when possible. This item will be ongoing for as many cycles as it takes to complete.

User-developer Interaction

To help users report bugs in the correct tracker, we must publicize the new policy in our Kubuntu user forums as well as to our IRC support team. We may also consider drafting a wishlist of usability improvements for Launchpad, as well as considering creating a weekly aggregation of user blogs to keep an ear on demands and issues.

Recruitment

To recruit bug triagers, we should regularly hold Kubuntu bug days. We will publicize these on blogs, and improve our current documentation for bug triage. We can also hold training or tutorial sessions to train potential triagers in preparation for these bug days.

As Kubuntu's image improves, (or perhaps a bit before) we will start to focus more on recruitment, ramping up recruitment in the areas of general development, artwork, user support and bug triage.

Implementation Roadmap

This is the timeline with which we have mapped out what we wish to accomplish between the upcoming Long Term Support (LTS) Kubuntu release and the next LTS in 2 years time.

Pre/early 10.04, lay foundations

- Announce Project Timelord to the world! his will include:
    - Announce on kubuntu.org, kdenews.org and blogs. This will be an important part of restoring our image, so we will make a big deal of it.
    - Establish a Timelord-upstream communication link. Ask what upstream expects Kubuntu to do and be(come), where they think Kubuntu provides a inferior KDE implementation and what they like about Kubuntu. In general, get an idea what upstream thinks of Kubuntu.
    - Work with upstream to work upon improving, based on upstream feedback.
- Establish intrinsic policy changes:
    - Establish new bug triage policy.
    - Make it a practice to always consult with upstream personally to determine whether or not a currently-pre-release piece of software will be ready for our release. To assume makes an ass out of u and me. ;-)
    - Re-evaluate support policies. We should look to see what level of support we can really provide for our "officially" supported releases. At the moment we really can't provide total support for 4 releases all at once. For example the non-security SRU rate for Intrepid is abysmal, with Jaunty only gaining fixes for the most critical of issues it was released with. Once we come up with something concrete, we should write it down.

10.04

- Review all patches.
- Fix translations, through whatever means necessary .
- Improve quality assurance.
- Re-evaluate **all** current default applications for **all** purposes, keeping in mind: installed size, functionality and quality of localization. Make changes as necessary.
- Do thorough testing of all current Kubuntu-specific applications, making a list of all bugs encountered and quashing as many as possible. (It's an LTS after all, and some of our apps have embarrasingly old bugs.)
- Depreciate gdebi, replacing it completely with KPackageKit.
- Finish/include kubuntu-notification-helper to replace update-notifier-kde.
- Port printer-applet-kde from KSystemTrayIcon to the new KNotificationItem class.
- Package and include kcm-touchpad for touchpad configuration in System Settings.
- Add KAuth integration to userconfig. (A bit of a stretch, may be deferred to a later release in reality.)

10.10 through 12.04, the next LTS

- Work through any issues we were unable to solve, scheduling them until they are fixed.
- Language-selector revamp.
- Integrate software-properties-kde directly into a System Settings module.
- Port printer-applet to C++. (If feasible.)
- Make Jockey a System Settings module.

Ongoing changes

- Improve marketing, promotion.
- Improve developer-user interaction.