Google Summer of Code 2008 Proposal: Nepomuk Collection-Plugin for Amarok

Abstract

With Nepomuk, KDE (beginning with 4.1) will get a central place to store and access all kind of meta data. It gets more useful the more programs use it. With this project Amarok could become one of the first programs that makes full use of it and so a reference for other KDE applications which want to integrate Nepomuk. My goal is it to write a collection plugin for the upcoming Amarok 2 that reads the meta data of the songs from Nepomuk and write them back to it. Doing so, all the data would be easily available throughout KDE and new possibilities would be there for Amarok. As an example for one of these I will add custom labels support (Nepomuk calls them tags) back to Amarok.

Project

The Nepomuk-KDE Semantic Desktop project (nepomuk.kde.org) aims to become the central storage for meta data, the one from the files itself (e.g. size, author, id3-tags) collected by Strigi, the file indexer, user entered (ratings, tags, comments) and most interesting automatically generated data provided by the KDE applications (e.g. file source, relations between files and contacts or play counts). Nepomuk provides access to all these data to all applications to help them show helpful context information and it will allow to search through the data. Nepomuk itself stores this data using RDF with Soprano.

A lot of code rewriting and refactoring has been done for the next big Amarok release 2.0. A new clean framework allows it to write custom collection plugins as collection source. The first goal of this SoC project is it to write such an plugin which uses the data about the audio files on the system stored in Nepomuk to provide the collection and feed all possible meta data back into it. With the use of Nepomuks features then as goal two the user should be able to add arbitrary labels to songs, artists and albums.

Amarok and KDE as a whole would benefit form that in different ways (some of this will be long term possibilities which need support by other applications or are else wise not direct part of this project):

- As Strigi (file indexer and desktop search tool used in KDE 4) already scannes all files and extract its meta data one would be able to just start Amarok and the whole collection would be already there without any further collection scanning and setup.

- The central KDE wide storage of all the Amarok data will allow other applications to access and use it. So one could for example show and change comments, ratings and labels in dolphin or a audio tagging application or cd burner could make use of the information. Or even the user can use a different audio player (well it needs to use Nepomuk as well) from time to time with all data (playcount, ratings, scores and so on) from Amarok.

- All the data would be searchable with the a search Nepomuk search application (not yet available). So one can for example enter a part of a lyric and it will find the song with the help of the lyric Amarok stored in Nepomuk. Or one could e.g. search all songs bought with Amarok (well for that to work Amarok needs to store this information in Nepomuk, integrate that would probably be behound the scope of the SoC but it could be possible in the long term) .

- Amarok will itself be able to make use of other foreign meta information provided by Nepomuk. For example that could be used in smart playlists. (build a playlist with all songs which the user got from a selected contact (with Kopete or E-Mail e.g.)) (sure this would need Kopete and K-Mail to feed that Info into Nepomuk).

- Nepomuk will track movement and renaming of the files and Strigi automatically index new files. So Amarok does not need to do this and it can completely relay on Nepomuk for that. (I have to mention that at the moment, as Nepomuk is still in development, it is not able to do this, but it will most likely be able to do it in the nearer future.)

- As Nepomuk is a new technology it new to all KDE developers. Amarok could become a reference for other developers who want to integrate Nepomuk features in their applications. For that I will cleanly document my work on Amarok.

- For the Nepomuk project it is helpful to have a bigger and quite data heavy program like Amarok use it form the start. So they could make first experiences with it in real usage and further optimize it. On the other side Amarok could influence the development of Nepomuk especially because the main Nepomuk developer will probably mentor this project.

As Nepomuk is not used in any bigger project at the moment and as it is a very new technology there are some things which should be considered:

- In this project even more than on others there is a lot research and experimenting needed so it is less time available for actual implementation. That also means that is harder to estimate the time needed for different tasks.

- Also it is not really clear if Nepomuks performance is great enough to operate directly on the Nepomuk data. This needs to get figured out. If it shows that it is to slow for Amaroks needs in fast browsing and querying the collection I will use a SQL database as an intermediate cache for the most used data (paths, artists list, album names and titles and eventually a few more). That way one could have the speed from the SQL database together with the new possibilities from Nepomuk.

- Another problem is that (at least for now) Strigis capabilities in reading tags from audiofiles are far behind those of the current Amarok collection scanner which uses Taglib to read tags. Here communication with the strigi developers is needed to figure out how there plans and roadmap to evolve that is. In the meantime a possible workaround would be to let the existing Amarok collection scanner do that work and let it write the data instead to the SQL database to Nepomuk if using this plugin as a collection provider. This needs to be discussed with the mentor and the Amarok Team as a whole.

For the second goal (re enable support for labels in Amarok using Nepomuk) some work on the Amarok GUI is needed. As part of SoC that will probably only become some first go on it to just do the basic things like show and add tags through an item in context menu of the collection browser. With an extension to the current meta data classes I will write it would late be easy to use and show the tags in different parts of Amarok. For example write a plasma applet to show a tag cloud or the tags of the current song in the context view.

Time line:

I commit to work at least 30 hours per week on average on the project. I have to take in account that I am from Germany with different schedules in universities than in the USA and so a part of the SoC time fall into the lecture period (in fact the time until July the 4th) . Because I only will attend a few classes in university and also be able use the remaining time starting from 5th of July full time for the project it will work out. My plan is the work about 25 hours a week in that first period ending with July the 4th and using the rest of the time fulltime with a around 40 hours a week working on the SoC project.

04/15 – 5/4	Get an overview and understanding of the Amarok code base, read Nepomuk documentation about the API, RDF, the SPARQL query language and get used to the Nepomuk API, make some tests of the performance of Nepomuk to decide together with the mentor and the Amarok developers if a SQL-Database (as explained before) should be used as cache
5/5 – 6/1	Coding on the Nepomuk collection plugin, while getting feedback from mentor and Amarok devs on which information should be stored and where exactly (if for example new Data Classes need to be defined in Nepomuk).
6/2 – 6/15	Investigate if we could relay on strigi for collection scanning or if I have have to extend the current Amarok collection scanner to write its data into Nepomuk storage. Code on the Collection Scanner if needed. If it turns out, that is is not needed I will gain about a week buffer to polish things at the end or as buffer time for unexpected things.
6/16 – 6/29	Planing on how to extend the Amarok Meta-Data Classes for support of labels. Doing this with heavy talking with the Amarok developers about the ideal way of doing that. Start coding on it.
6/30 – 7/6	Finishing work on supporting labels
7/7 – 7/20	Working on the GUI to make use of the labels features added before
21/7 – 8/10	Testing, bug fixing and buffer time. If there is time start begin on adding some small things which makes use of the new possibilities or polsing the GUI for label support. (for example integrate a way to select from a range of common used predefined labels)
8/11 – 18/8	Final testing and documentation.

About Me:

My Name is Daniel Winter and I am 26 years old. I am a 4th year student of Computer Science on the Fachhochschule (University of Applied Sciences) Dortmund in Germany. I am have experience with Java, C# and C++ (among others). The last 1,5 years I worked at a local company on a business software with C# and storage in a MySQL Database. I am quite new to QT(4). I read a book about it and played around a little with it, but it seems quite easy to get used to it. KDE libraries are also new for me. I got known to Linux about 9 year ago. I fully switched to Linux for 6 years now. I made first experience in FOSS development while porting the Bemused server to Linux (bemused.sf.net). It is a smaller software written in C to remote control XMMS with Nokia mobilephones over bluetooth. I also made two patches to Amarok a few years ago.

I would like to work on this project because I am really excited about the things Nepomuk will do for KDE in the long term and would really like to see it become a success. I first liked the idea as I heard about the now dead project tenor and I am happy that Nepomuk fills this gap. Also Amarok is one of my favorite applications I would like to see it further evolve. This project brings this to things together, which makes it a very interesting project for me.