

Installation Notes for SilkRoute II

Chris Ré
University of Washington
Department of Computer Science

May 26, 2005

1 Overview

SilkRoute is a tool for publishing relational data as XML. It does this by providing a virtual XML view of the data, which we term a *Public View*. SilkRoute offers many additional features including an optimizer, a java api and a supporting tool chain. Additionally it comes with data and a skeleton webservice. SilkRoute is intended to be used in an information integration setting.

Please keep in mind that SilkRoute is a research prototype. While I'd like it to be as painfree as possible - odds are there will be more than a few warts. This document is not intended to be full of details of the innerworkings of SilkRoute. There are related technical reports which cover these subjects, linked off the website <http://silkroute.cs.washington.edu>.

2 Installation Notes

In order to use SilkRoute, you need to have a database system installed. SilkRoute has been tested over MySQL, PostGRES and SQL Server. It uses an ODBC driver to connect to these sources and should work with any ODBC compliant relational source.

2.1 Needed Software

The following is a list of common components of a SilkRoute installation.

1. A Relational Engine
2. An ODBC Driver for the database
For example: myodbc or psqlodbc
3. An ODBC Driver Manager
For Example: UnixODBC *preferred* iODBC

4. OCaml. A French ml variant.
<http://caml.inria.fr>
5. Galax. An XQuery implementation. Source Version.
<http://www.galax.org>
6. (Only for SQLServer) FreeTDS.
<http://www.freetds.org>

2.2 Installation Procedure

Database ODBC Install Install your database engine and its associated ODBC engine. This process is engine specific but is generally not terribly painful. Testing that this installation works with their provided test suites is *highly* recommended.

ODBC Manager Install Install a driver manager. SilkRoute can be made to work with iODBC but UnixODBC is preferred. After installation verify that you can connect to your sources with the provided isql executable. Testing to make sure this component is working will make your life much much easier if something should go wrong.

Install OCaml Install the latest version of OCaml (3.08 at the time of this writing). OCaml is a wonderful language to use if you're looking for a new one to learn.

Install Galax Galax (0.5) is an implementation of XQuery that has very clean processing phases for dealing with XQuery compilation. SilkRoute uses Galax to compile to Core XQuery. Again a nicely written piece of software¹. This may take some doing as the tradeoff for being highly modular and extensible is that there are many factored out dependencies. Follow these directions carefully. They should be very clear.

Install the Silkroute Source Open the file `config/Makefile.constants`. In this file set the home directory, the base of your ocaml installation, the location of galax's configuration file, the base of your java installation² and also your unixodbc directories. Then you should simply need to type `make` in the root source directory and silkroute and its libraries will be built.

¹...at least the parts I had little to do with.

²This is only necessary if you anticipate using SilkRoute from Java. An example is the provided skeleton webservice

3 Usage

Silkroute provides a standard unix help function from the commandline (`--help`). There are many options here. Please read the below to understand how the different files of SilkRoute function. First a simple step-by-step example post-installation. Suppose you have a postgres database installed.

1. Go to the `sample/data` directory. There is a file named `db_dump.txt`. Import this file into a fresh postgres database instance.
2. Edit the `.cv` file of the sample `Student.cv` as described below in the section on Canonical View.
3. Execute Silkroute.

A sample commandline for executing silkroute would be:

```
$ ./silkroute -postgres Student1.uq
```

3.1 Details

Provided Data The `sample` directory provides a sample database instance and some user queries. User queries are denoted with the `.uq` suffix. Public Views are denoted with the `.pv` suffix. Canonical views with the `.cv` syntax. There are dumps for the example student database.

Canonical Views The `.cv` files contain connection information as its first line. The syntax of this line looks like:

```
CONNECTION <odbc source> <user> <password>
```

You can override any of these parameters with the `-db`, `-user` or `-pass` flags from the command line. These files also contain a schema of your data³. This type limited type information is necessary to deal with the complex semantics of XQuery.

Public Views The public views represent the virtual XML document. A user will query over these views as is shown in the provided user queries. These are essentially XQuery programs augmented with two special functions, `CanonicalView` and `PublicView`. These functions allow you to retrieve both the canonical view⁴ of a database and, since XQuery is composable, other public views. These views allow users to build up arbitrarily nested - but not recursive - views of their relational data.

³I know it looks painful but there is a tool that takes a schema in postgres or mysql format and creates this file.

⁴There is a restriction that you can talk about at most one canonical view.

Database Options Each of the three supported engines has a flag to deal with the subtle difference of the connections. For example, TDS requires that you not reuse connection handles or have aliases over a certain length. These small details are taken care of by setting the appropriate database flag (eg: `-postgres`, `-sqlserver` or `-mysql`). These flags also allow SilkRoute to use cost estimates from the engine which can result in significantly better optimization choices.

4 Tool Chain

5 Acknowledgements

Should we put in grant information and the people who have helped?