SEM

Add Additional XPath Functionality to Client

Narrative

In the old client, performing an XPath query results in a srcML archive containing the results of the query. While this can be convenient to view just the chucks of code that match the query, for the purposes of stacking multiple XPath queries or displaying query results in the context of the document body, it would be beneficial if users had access to functionality in the cli that allowed for XPath queries to be marked up in the context of the entire docuemnt.

To support this functionality the new client should provide options that allow the user to provide either additional element or attribute markup to denote an Xpath query result. Example 1 illustrates the search result markup with an element and Example 2 shows the use of an attribute.

Example 1:: Element to identify xpath search result. Element contains the namespace of “test” and element name of “search”. CLI command options follow.

```
<test:search><while>while<condition>( <expr><name>a</name>
<operator>&lt;</operator> <literal type="number">20</literal>
</expr> )</condition>
   <block>{
       <expr_stmt><expr><name>a</name><operator>++</operator>
</expr>;</expr_stmt>
   }</block></while></test:search>
```

Element CLI Command:

```
srcml [srcML input file] –xpath=//src:name –xmlns:test=http://test.cc
```

Example 2:: Attribute to identify xpath search result. Attribute contains the namespace of “test”, attribute name of “search”, and a value of “Attr”. CLI command options follow.

```
<while test:search="Attr">while<condition>( <expr><name>a</name>
<operator>&lt;</operator> <literal type="number">20</literal>
</expr> )</condition>
   <block>{
          <expr_stmt><expr><name>a</name><operator>++</operator>
</expr>;</expr_stmt>
   }</block></while>
```

Attribute CLI Command:

```
srcml [srcML input file] –xpath=//src:name –xmlns:test=http://test.cc
```

One specific applicaiton of this feature would be in assisting the srcMX GUI client for srcML to provide a visualization of the XPath queries within

the scope of an entire file instead of simply showing the results on their own.

Change Plan

- API support provided by libsrcml is assumed
    - Coordination with maintainer of libsrcml for feature support is step one.
- Add fields to srcml_cli.hpp to store necessary input data for functionality
- Add cli options and functionality to collect the data from CLI input
- Extract Method Refactoring on transform_srcml.cpp
- Add element logic to method
- Add attribute logic to method

```
INSERT DIAGRAM AND ANALYSIS EXPLANATION HERE
```

Add Git Input Source to Client

Narrative

The srcml client currently support a number of input sources from both local and remote files including HTTP(S), FTP(S), SSH, etc. While the remote protocol support is quite comprehensive, one noticible omission is native support for a source repository using git version control. This feature addition could allow users to enter a git repository url and even a

revision sha as input to retrieve the source of a project as input to srcml. This would prevent a user from having to first clone a repository externally to their local machine before running srcml and would reduce the number of steps required in this circumstance to one. This feature can also prove useful for a webservice based version of the srcml client in which access to the source code on a physical device (such as tablet or phone) might not be feasible. Example1 shows the the CLi option interface extension.

Example1: Prefix of “git://” to identify a git resource and –revision to aquire a specific version of the sourcr repository.

```
srcml git://[local or remourl for repo] --revision=[SHA]
```

Change Plan

- Add new files src_input_git.cpp and src_input_git.hpp
- Update CMake to build with libgit2 (static) or load library dynamically
- Create function src_input_git (ParseQueue& queue, srcml_archive* srcml_arch, const srcml_request_t& srcml_request, const std::string& input_file) in src_input_git.* files.
- Use libgit2 for repository aquisition
- Take aquired data for each retrieved file and create and queue a ParseRequest
- Import src_input_git.hpp in create_srcml.cpp
- Add conditional to srcml_handler_dispatch function in create_srcml.cpp for “git”

- Add function src_input_git (ParseQueue& queue, srcml_archive* srcml_arch, const srcml_request_t& srcml_request, const std::string& input_file) to the conditional.

```
INSERT DIAGRAM AND ANALYSIS EXPLANATION HERE
```

Object Oriented Input Sources

Narrative

In it's present state, the new srcml client's design is purely functional in most respects. One specific area of functionality that could benefit from the use of an object oriented design would be the input source handling. With a more object oriented approach, an inheritance hierarchy combined with one or more polymorphic methods could be utilized to simplify the way in which the input sources are deterimined and replace much of the contitional logic in create_srcml.cpp and consolidate those input checks and object creation in srcml_input_src.hpp. The result of this change would make collecting data from an input source in create_srcml.cpp a single method call and make adding additional input sources more straight forward as developers would only need to inherit from an abstract base class and fill in the necessary functionality.

Change Plan

- Create Abstract Class "input_src"

- Include data members and functions from “srcml_input_src”
- Include public virtual function to generate a parse request to be implemented by all functions that implment the abstract class
- Starting with “src_input_file”, convert to subclass of “input_src”
- Substitute object creation in “create_srcml” and usage in place of function call to ensure behavior is preserved
- Move object creation into “srcml_input_src”
- Change function in “creat_srcml” to pass an “input_src” object
- Remove object creation from “create_srcml” and ensure behavior is preserved
- Repeat for all other input sources
- Replace includes to srcml_input_src.hpp to input_src.hpp
- Remove srcml_input_src.*

```
INSERT DIAGRAM AND ANALYSIS EXPLANATION HERE
```