Mercurial SCM

Kyle Brooks, Jesse Hunt, Chris Sigmund

What is Mercurial?

Mercurial is revision control tool for both version control and ease-of-access for group software development, mainly based on Python architecture. It was developed after BitKeeper, the former repository for the Linux kernel project, announced there would no more free source code hosting.

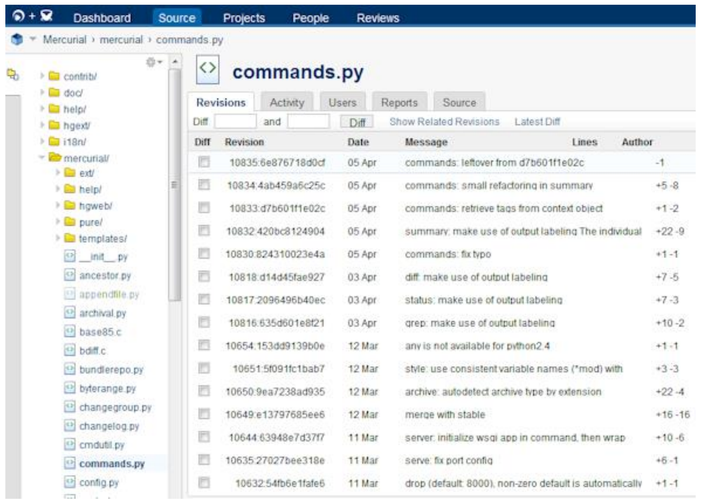

Figure 1. Mercurial implemented in Crucible software

Building Mercurial

The process to build Mercurial locally is relatively straight forward. Using a simple build command, the code is built from the /build directory, and placed both in local python directories on the system along with the built files in the /mercurial directory.

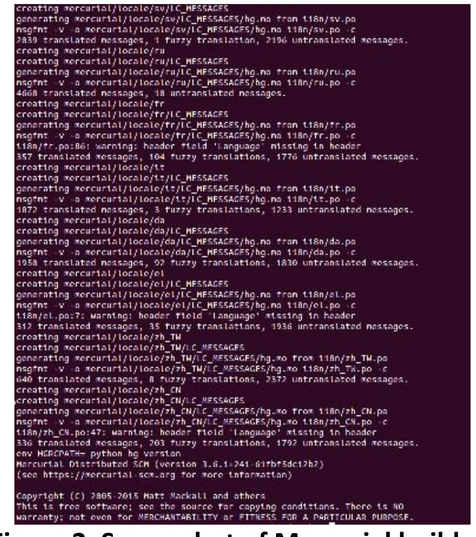

Figure 2. Screenshot of Mercurial build

Figure 3. Mercurial Logo

Making a Test Plan

The intention behind the test plan was to build an automated test system that could read a text file and interpret the results. The test plan focused on unit testing, so the units picked to test were methods that had clear inputs with concise and clear outputs, such as string manipulations and math methods. Python was chosen as the script of choice for ease of use with the native code.

Test Case	01
Requirement	Break a given integer into the largest increments of time it can be broken into.
Component	progress.py
Method	fmtremaining(int)
Test Input(s)	100
Expected Outcome	1m40s

Test Case	02
Requirement	Given the range of two points, calculate the point of overlap between them
Component	simplemerge.py
Method	intersect(ra, rb)
Test Input(s)	(0,100), (50,150)
Expected Outcome	(50,100)

Figure 4. Example Test Cases

Implementing a Test Plan

The test plan proved to be a challenge to implement. Because of the use of native python code, it was difficult to effectively use assertions to test dynamically created statements. Through extensive testing and research, the team was able to create a test script that quickly and effectively ran through our test cases.

05	namespaces.tolist()	Return elements in given array concatenated into one ele
06	progress.fmtremaining()	Break a given integer (representing time in seconds) into
07	progress.fmtremaining()	Break a given integer (representing time in seconds) into
08	progress.fmtremaining()	Break a given integer (representing time in seconds) into
09	progress.fmtremaining()	Break a given integer (representing time in seconds) into
10	progress.fmtremaining()	Break a given integer (representing time in seconds) into
11	simplemerge.intersect()	Given the range of two points, calculate and return the p
12	simplemerge.intersect()	Given the range of two points, calculate and return the p
13	simplemerge.intersect()	Given the range of two points, calculate and return the p
14	simplemerge.intersect()	Given the range of two points, calculate and return the p
15	simplemerge.intersect()	Given the range of two points, calculate and return the p
16	templatefilters.count()	Return the length of the given string
17	templatefilters.count()	Return the length of the given string
18	templatefilters.count()	Return the length of the given string
19	templatefilters.count()	Return the length of the given string
20	templatefilters.count()	Return the length of the given string
21	namespaces.tolist()	Return elements in given array concatenated into one ele
22	namespaces.tolist()	Return elements in given array concatenated into one ele
23	namespaces.tolist()	Return elements in given array concatenated into one ele
24	namespaces.tolist()	Return elements in given array concatenated into one ele
25	namespaces.tolist()	Return elements in given array concatenated into one ele

Number of Tests: 25
Passed: 24
Failed: 1

Figure 5. HTML Output of the script

Breaking Mercurial

The final objective with Mercurial was to break the methods testing so they would show up as having failed. During the process of breaking the code, the largest difficulty faced was figuring out where the code was executing from and how to make sure the code we changed went there. This proved to be critically intertwined with the previously mentioned build process.

References

Mercurial. (2015, May 26). Retrieved November 29, 2015, from https://www.mercurial-scm.org/wiki/Mercurial