Midterm Assignment

Name: Danyal Ahmad

Class: BSSE

Course: Software Verification and

Validation

ID # 13838

Module: Semester-VI

Submitted To: Zain Shaukat

BUGZILLA

WHAT IS BUGZILLA?

- Bugzilla is a "Defect Tracking System" or
 - "Bug-Tracking System".
- Defect Tracking Systems allow individual or groups of developers to keep track of outstanding bugs in their product effectively.
- → Bugzilla is a Open Source Tool.

 It is written in Perl and Uses MySQL database.

Bugzilla has two forms of search:

- → It has Google-like bug search which is simple to use, and it also searches the full text of a bug.
- → It provides you a very advanced search system where you can create any type of search that you want such as time-based searches (For example, you want to see the list of bugs whose priority has been changed since last two days).

FUNCTIONALITY:

Bugzilla boasts many advanced features. These include:

- → Powerful searching
- → User-configurable email notifications of bug changes
- → Full change history
- Inter-bug dependency tracking and graphing
- → Excellent attachment management

- → INTEGRATED, PRODUCT-BASED, GRANULAR SECURITY SCHEMA
- → FULLY SECURITY-AUDITED, AND RUNS UNDER PERL'S TAINT MODE
- → A ROBUST, STABLE RDBMS BACK-END
- → WEB, XML, EMAIL AND CONSOLE INTERFACES
- → COMPLETELY CUSTOMIZABLE AND/OR LOCALIZABLE WEB USER INTERFACE

→ EXTENSIVE CONFIGURABILITY

→ SMOOTH UPGRADE PATHWAY BETWEEN VERSIONS

ADVANTAGES OF BUGZILLA:

- → It is an open source widely used bug tracker;
- → It is easy in usage and its user interface is understandable for people without technical knowledge;
- → It easily integrates with test management instruments;
- → It integrates with an e-mailing system;
- → It automates documentation.

DISADVANTAGE:

Not all testers like to work with Bugzilla. Some of them find that its interface is too sophisticated, complain that it is hard to manage the logged in errors and that there are many defects in code of this instrument.

SUPPORTING LANGUAGES:

→ Perl

→ Java

→ PERL6

→ PHP 5 or 6

→ PYTHON

→ D Programming Language

→ RUBY

→ C#

Perl:

pros

Lots of modules available in CPAN. They are very well tested due to distributed test system. Anybody can participate in testing. Relatively fast.

mod_perl is very mature and extremely fast.

Would make us not have to port, avoiding possible Second System Effect and making any transitions (such as to a web framework) easier.

Cons

See The Problems of Perl.

Private Methods aren't well supported require additional scaffolding.

Multiple Inheritance can be problematic in extreme edge cases.

Certain elements of syntax can be confusing for new users, by long experience in training new Perl users:

The difference between () and [].

The fact that %var is a () (which is also the array notation) but {} is \$var.

PERL 6

Pros:

Implements many features we'd like to have that Perl5 doesn't have.

Would be the easiest language to port our current code to, since it's so similar to Perl5.

Cons:

Essentially vaporware. There is an interpreter written for it, but it's in Haskell and it's not very popular yet.

Perl 6 is still very punctuation-heavy and very influenced by Perl 5.

PYTHON

Pros:

Quite popular.

Stable.

Actively developed.

Quite a lot of modules (but fewer than CPAN)

Cons:

Not having curly-braces on "if" statements and other blocks makes it hard to figure out where you are in the block structure without a special editor to help (like Komodo).

Poor Unicode handling--strings are ASCII by default, and are Unicode only if you prepend them with u, like u"string".

Python has no equivalent to Perl's "taint" mode.

RUBY:

Pros:

Extremely modern language, lots of great features built-in to the language.

RubyGems, a CPAN-like method of installing modules.

Becoming more and more popular.

Cons:

Not yet as well-known as Perl and other languages (lacks the large userbase of Perl, Python, or PHP) Good Unicode handling isn't coming out until 1.9, but that's the next release.

JAVA:

Pros:

Excellent "design by contract" features (compared to the other languages in this listing).

Very stable.

Fast.

Popular, although more in Enterprise apps and less in Open Source than scripting languages.

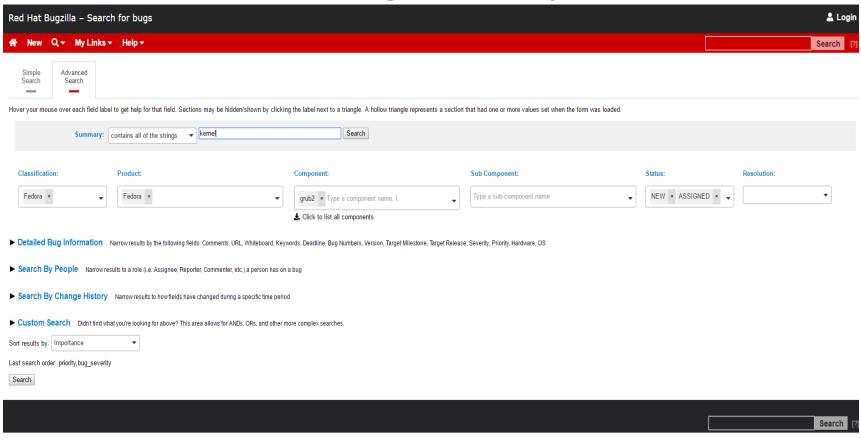
Cons:

Generally slower to write in than scripting languages.

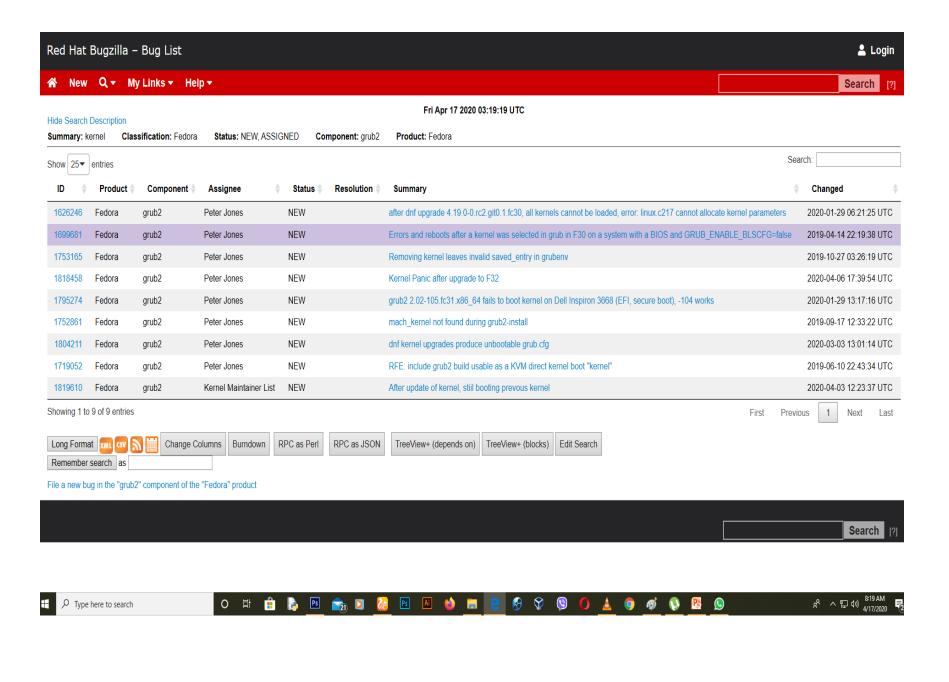
Nothing like CPAN's client-side module installer until Java 7 (see JSR 277).

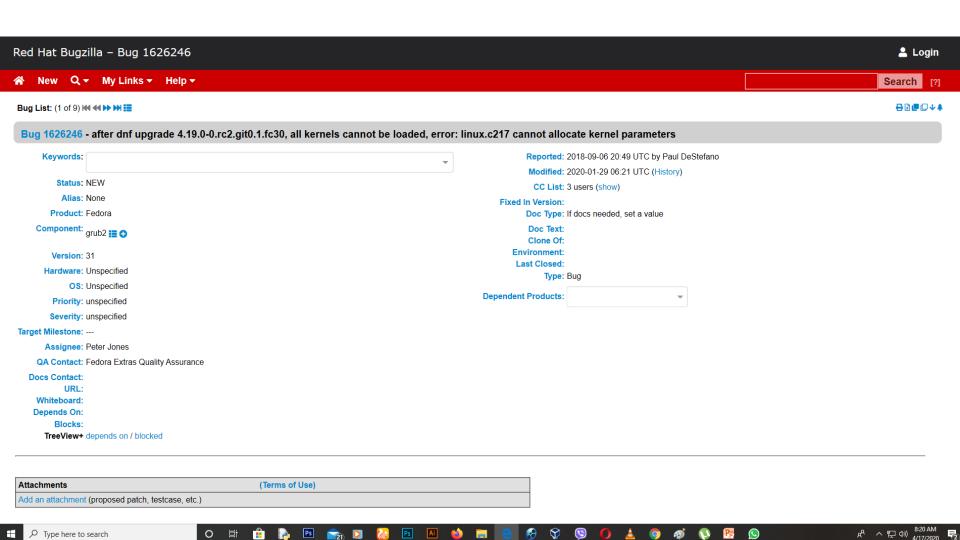
The standard Java Classpath is not FOSS, but Sun's OpenJDK is.

Working Example:









Thank You