FIU School of Computing & Information Sciences

Senior Project, 2013 Fall

Dynamic Presenter – Version Control

Student: Carlos Fernandez, Florida International University
Mentor: *Juan Caraballo, IBM*
Instructor: Masoud Sadjadi, Florida International University

Problem

A company may show presentations to an audience for a multitude of reasons. Making a presentation is time consuming work, so multiple collaborators may want to work on the same presentation to speed up development. However, in collaborative projects it is difficult for managers to effectively control the changes made by other contributors and to keep track of multiple versions of the same presentation.

Such a system requires the need to restore previous versions of slides to undo unwanted changes. For that reason the Version Control Subsystem was developed.

Current Systems

While this project is not based on a project from a previous semester, systems that solve similar problems exist.

Both Google Docs and Office 365 are existing online document editors that allow multiple people to collaborate on the same project in real time. However, they do not focus on preventing the introduction of unwanted changes.

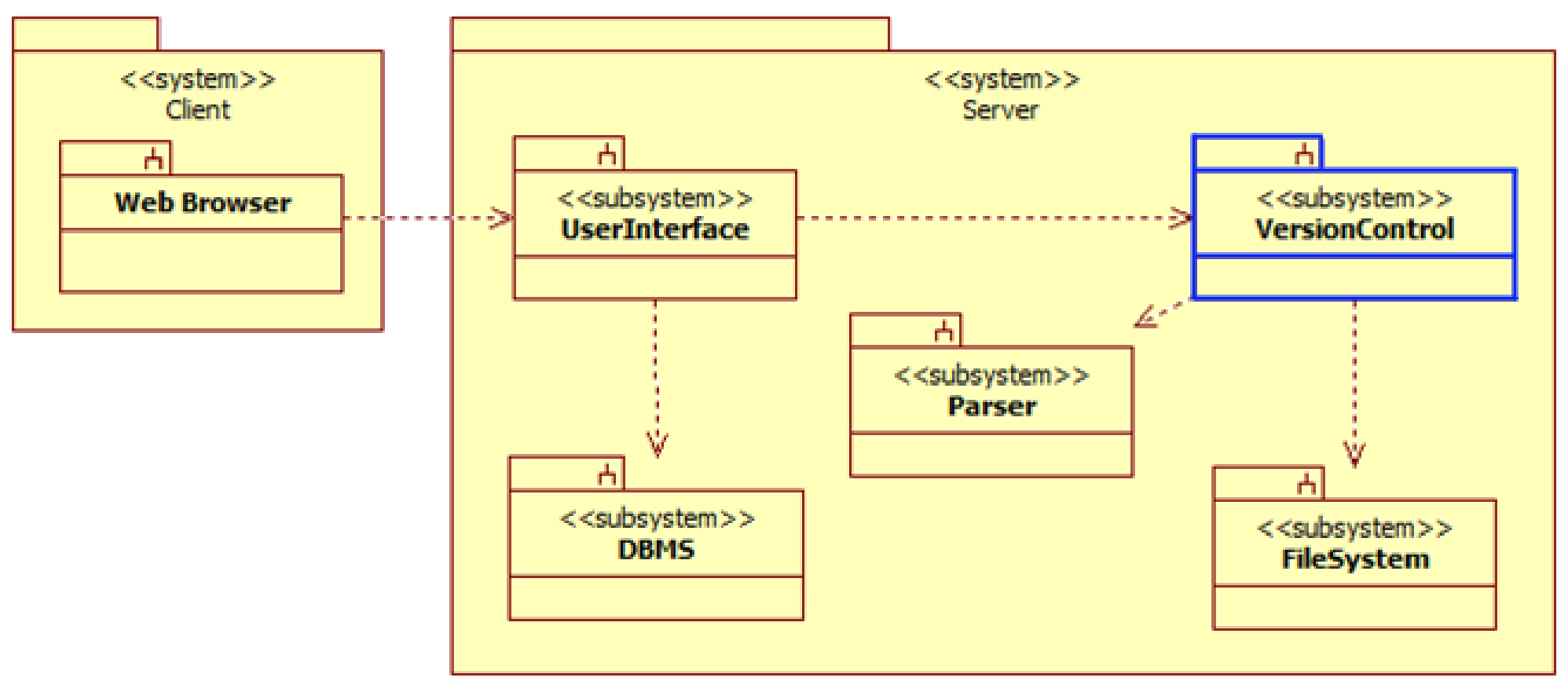
Requirements

Version Control Subsystem Requirements:

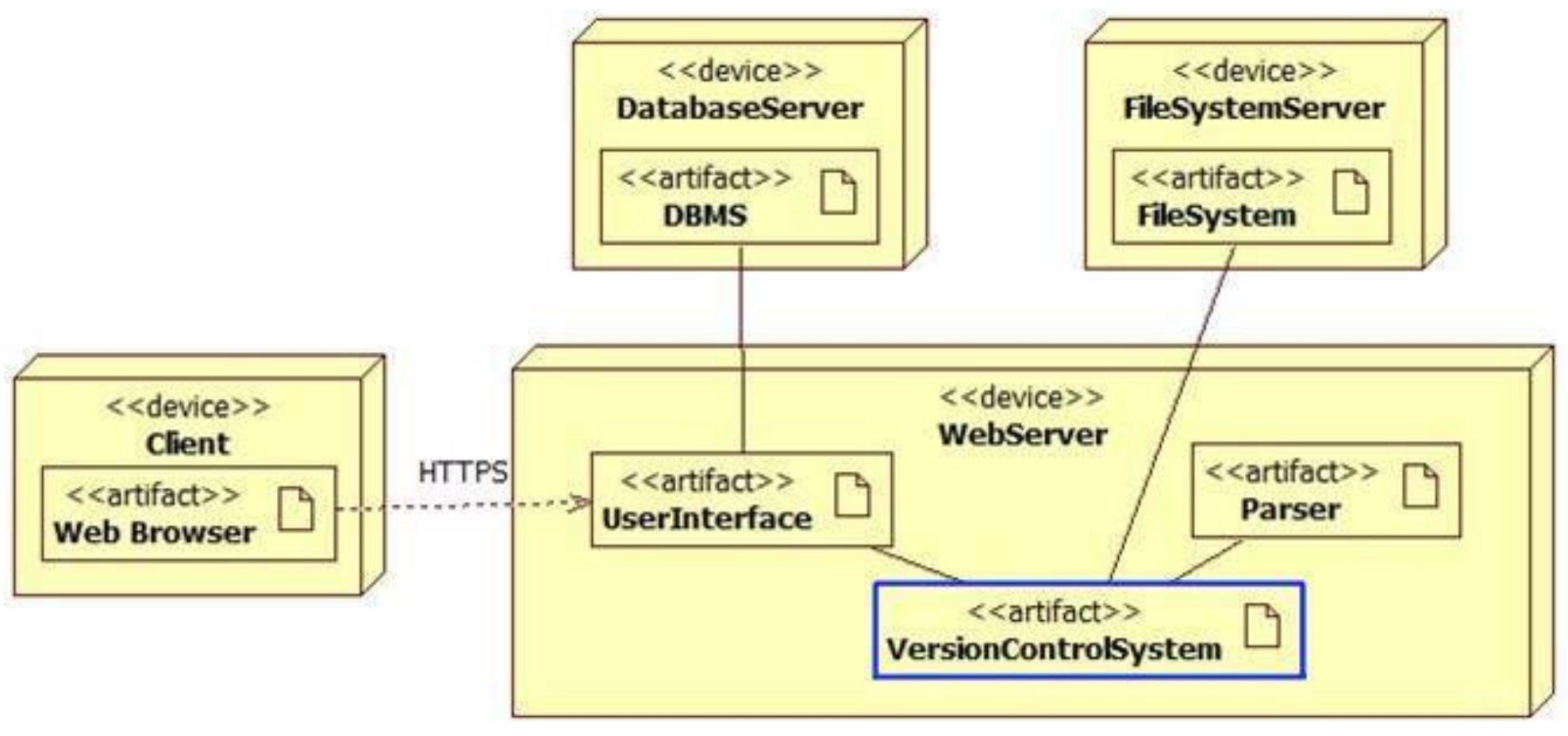
- Add slides to the current presentation
- Update slides and keep the previous version. Previous slides should be retrievable.
- Save the current presentation. Saved presentations can be retrieved later.
- Enforce check in / check out
- Expose parser functionality to the user interface

System Design

Architecture Diagram:

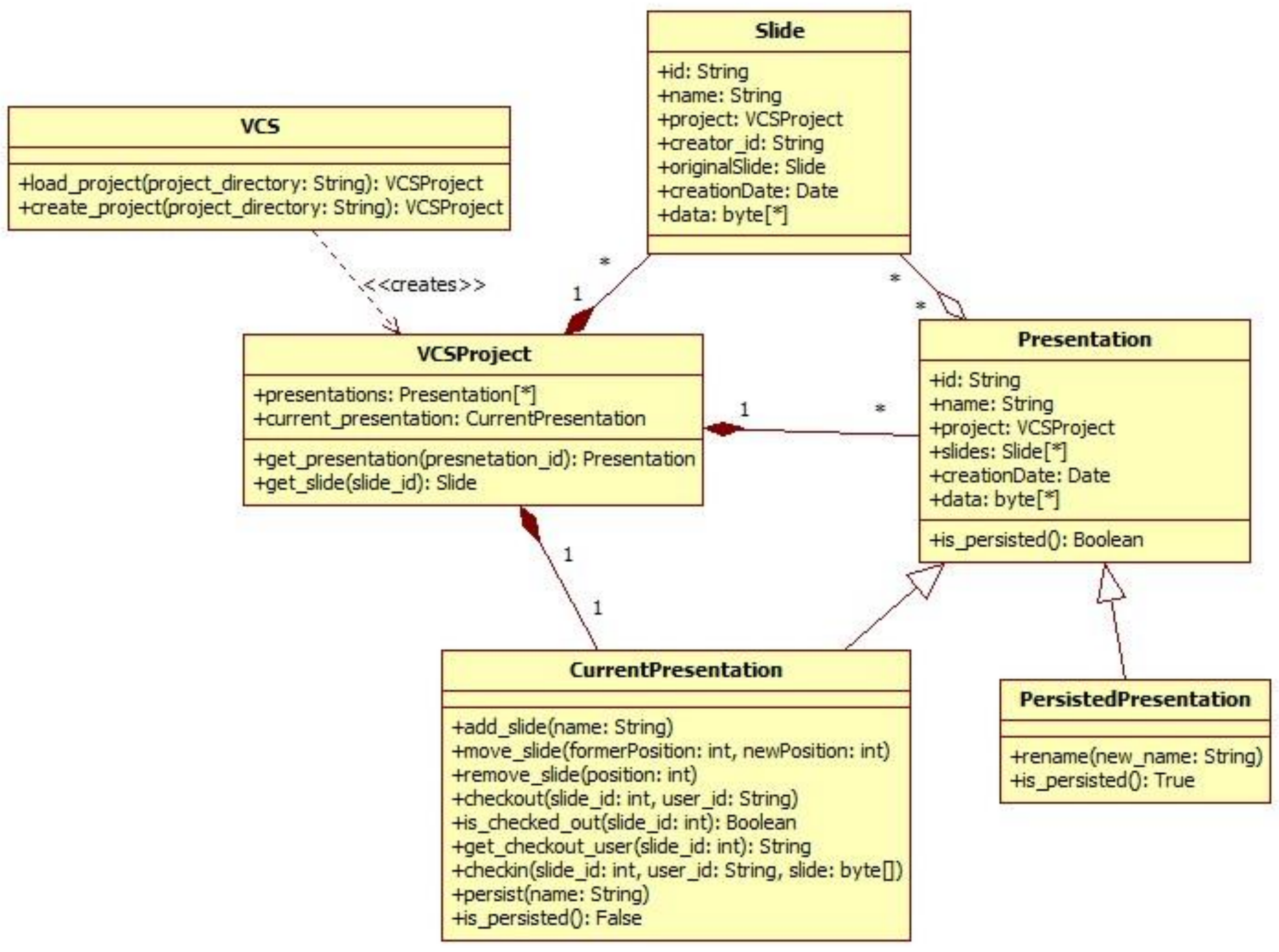


Deployment Diagram



Object Design

Class Diagram:


Slide
+id: String
+name: String
+project: VCSProject
+creator_id: String
+originalSlide: Slide
+creationDate: Date
+data: byte[*]
VCS
+load_project(project_directory: String): VCSProject
+create_project(project_directory: String): VCSProject
<<creates>>
*
1
*
*
VCSProject
+presentations: Presentation[*]
+current_presentation: CurrentPresentation
+get_presentation(presnetation_id): Presentation
+get_slide(slide_id): Slide
1
*
Presentation
+id: String
+name: String
+project: VCSProject
+slides: Slide[*]
+creationDate: Date
+data: byte[*]
+is_persisted(): Boolean
1
1
CurrentPresentation
+add_slide(name: String)
+move_slide(formerPosition: int, newPosition: int)
+remove_slide(position: int)
+checkout(slide_id: int, user_id: String)
+is_checked_out(slide_id: int): Boolean
+get_checkout_user(slide_id: int): String
+checkin(slide_id: int, user_id: String, slide: byte[])
+persist(name: String)
+is_persisted(): False
PersistedPresentation
+rename(new_name: String)
+is_persisted(): True

Implementation

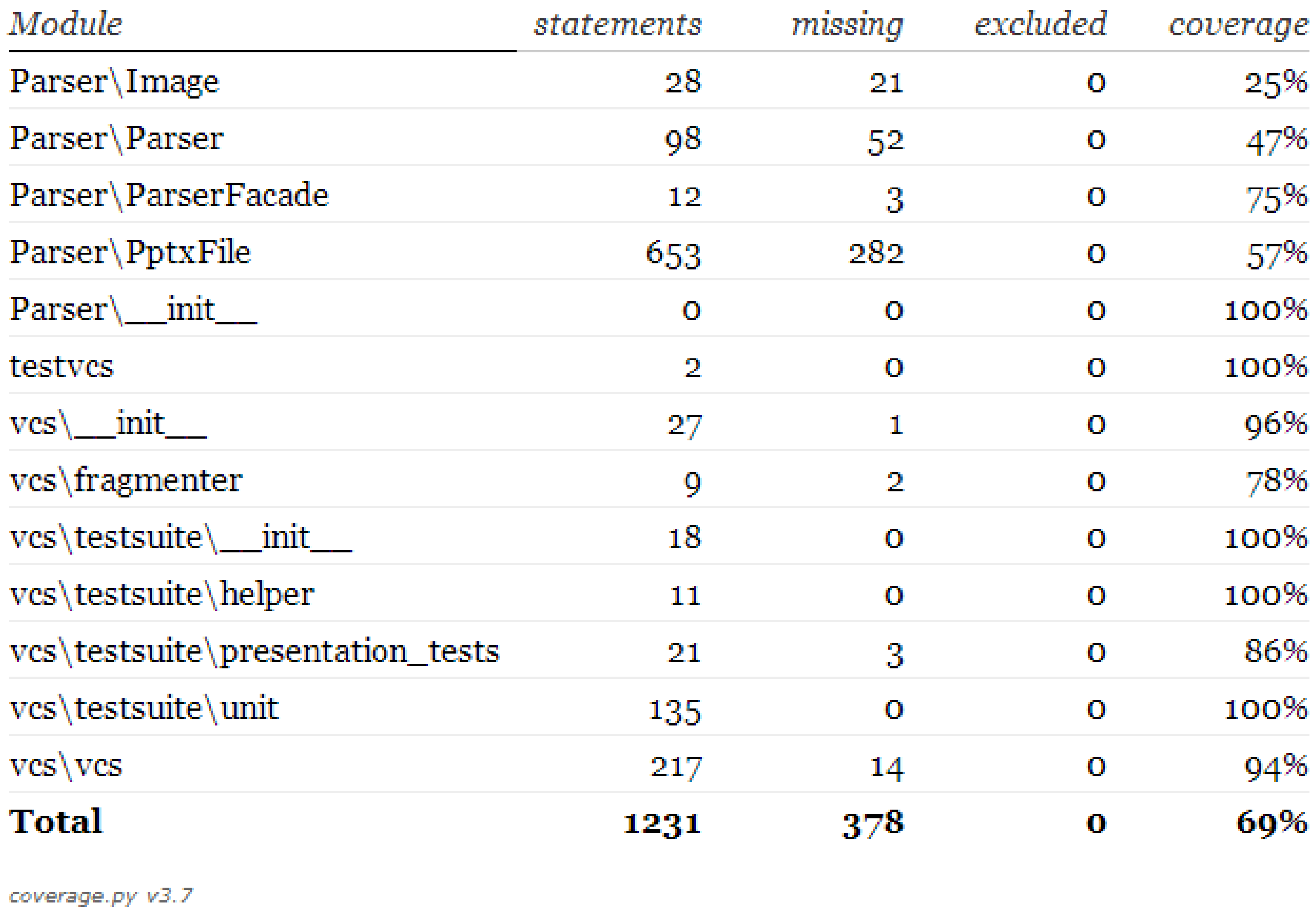
This subsystem was written in the python programming language, and uses SQLite for storing presentation and slide metadata. The actual slide data is stored as independent data files on the file system. The system uses the proxy pattern so that loading data is efficient.

Python was chosen as the system's programming language both because it is simple to work with and because it lends itself well to implementing the other subsystems.

Metadata used to be stored without SQLite, but concurrency become a necessity and the current mechanism of persisting data was developed.

Verification

A simple test suite was created using Python's unittest module. This test suit attempts to examine every single possible input and output of the Version Control Subsystem.



Module	*statements*	*missing*	*excluded*	*coverage*
Parser\Image	28	21	0	25%
Parser\Parser	98	52	0	47%
Parser\ParserFacade	12	3	0	75%
Parser\PptxFile	653	282	0	57%
Parser__init__	0	0	0	100%
testvcs	2	0	0	100%
vcs__init__	27	1	0	96%
vcs\fragmenter	9	2	0	78%
vcs\testsuite__init__	18	0	0	100%
vcs\testsuite\helper	11	0	0	100%
vcs\testsuite\presentation_tests	21	3	0	86%
vcs\testsuite\unit	135	0	0	100%
vcs\vcs	217	14	0	94%
Total	**1231**	**378**	**0**	**69%**

coverage.py v3.7

Summary

- A version control subsystem is required to maintain and restore presentation and slide history
- The version control subsystem uses the parser subsystem internally, and is used by the user interface.
- The subsystem is being developed using Python and SQLite.
- A test suite was created to achieve as much testing coverage as possible.

Acknowledgement

I would like to recognize the hard work and dedication put in by my group members Jose Camino (Presentation Parser) and Jimmy Mauri (User Interface) throughout the duration of this project.