LATEX EDITOR REFACTORING

OVERALL REPORT

VERSION <1.1>

Δημήτριος Βαγενάς ΑΜ:2941
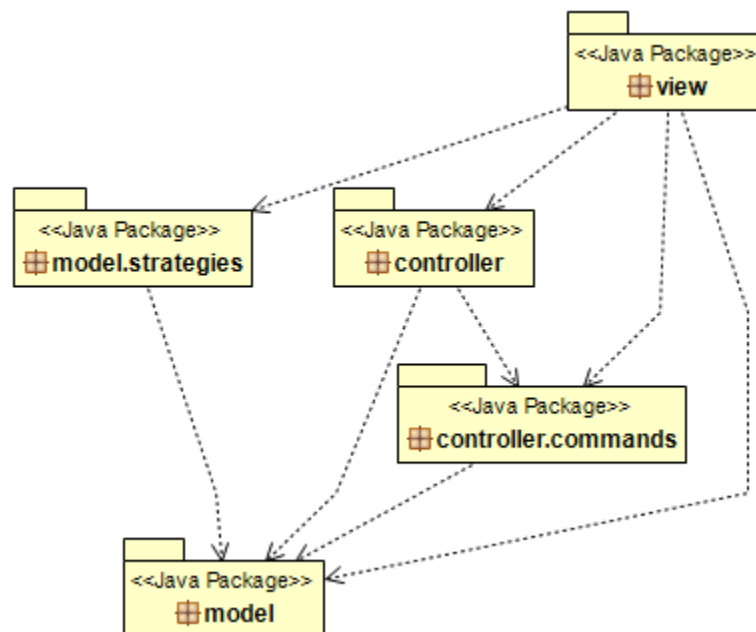
Ευτυχία Μπουρλή ΑΜ:4441

-

TABLE OF CONTENTS

INTRODUCTION

The goal of this project is to reengineer and extend an existing Java application that was produced in rush and with poor design management. This application facilitates the writing and maintenance of Latex documents. Below we describe the architecture, the detailed design and the CRC cards of the refactored application.
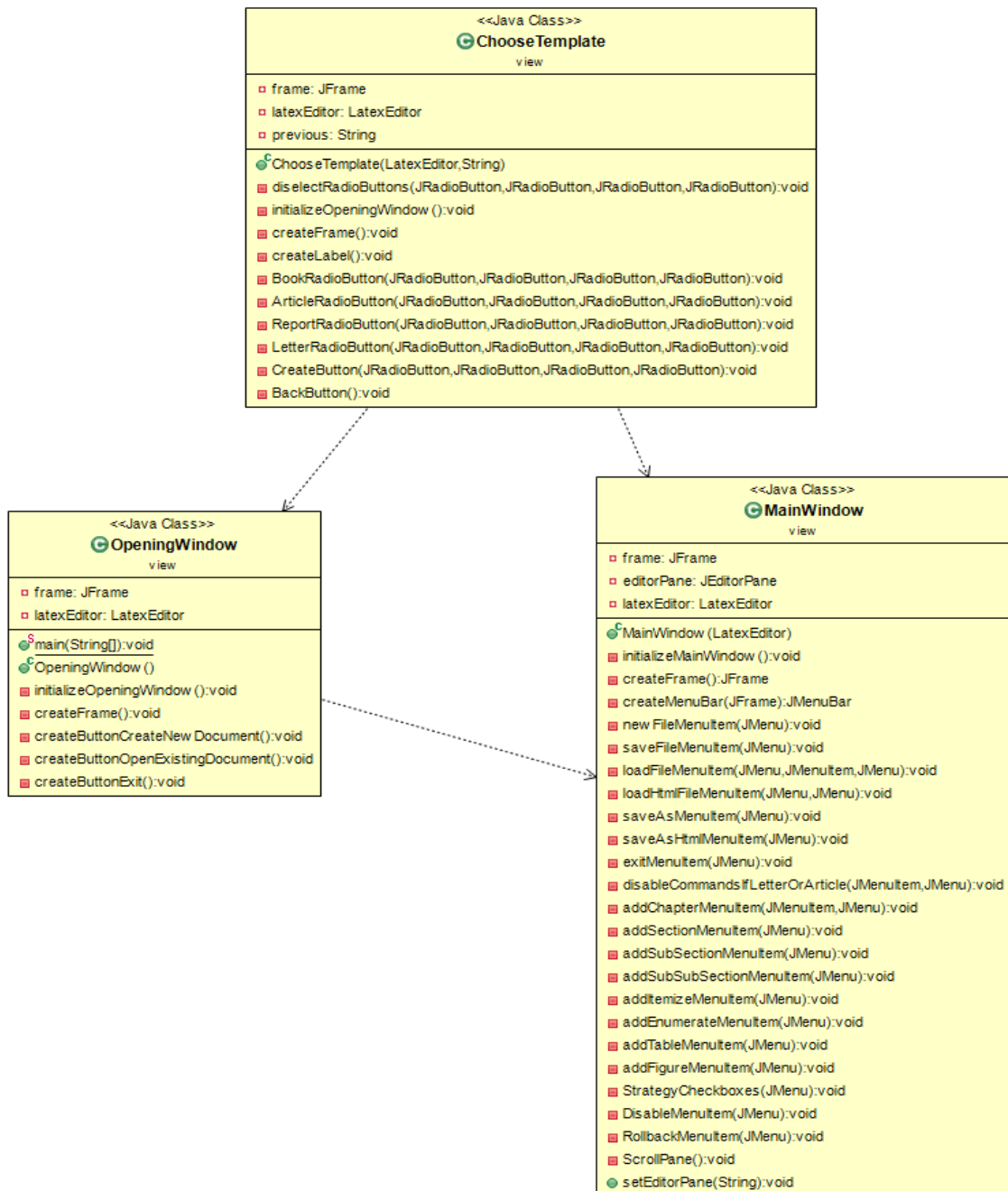
REFACTORED DESIGN

ARCHITECTURE

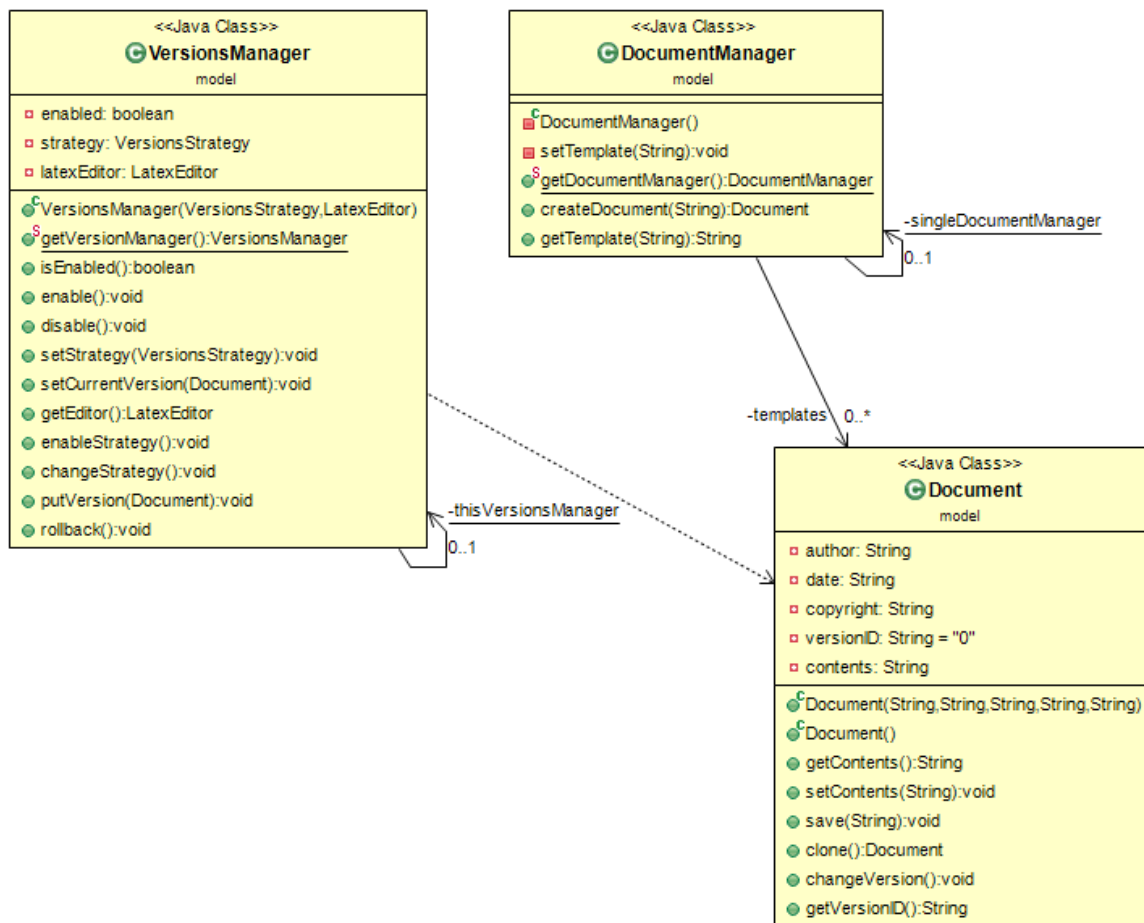- Here is the overall UML package diagram that shows the architecture of the refactored application

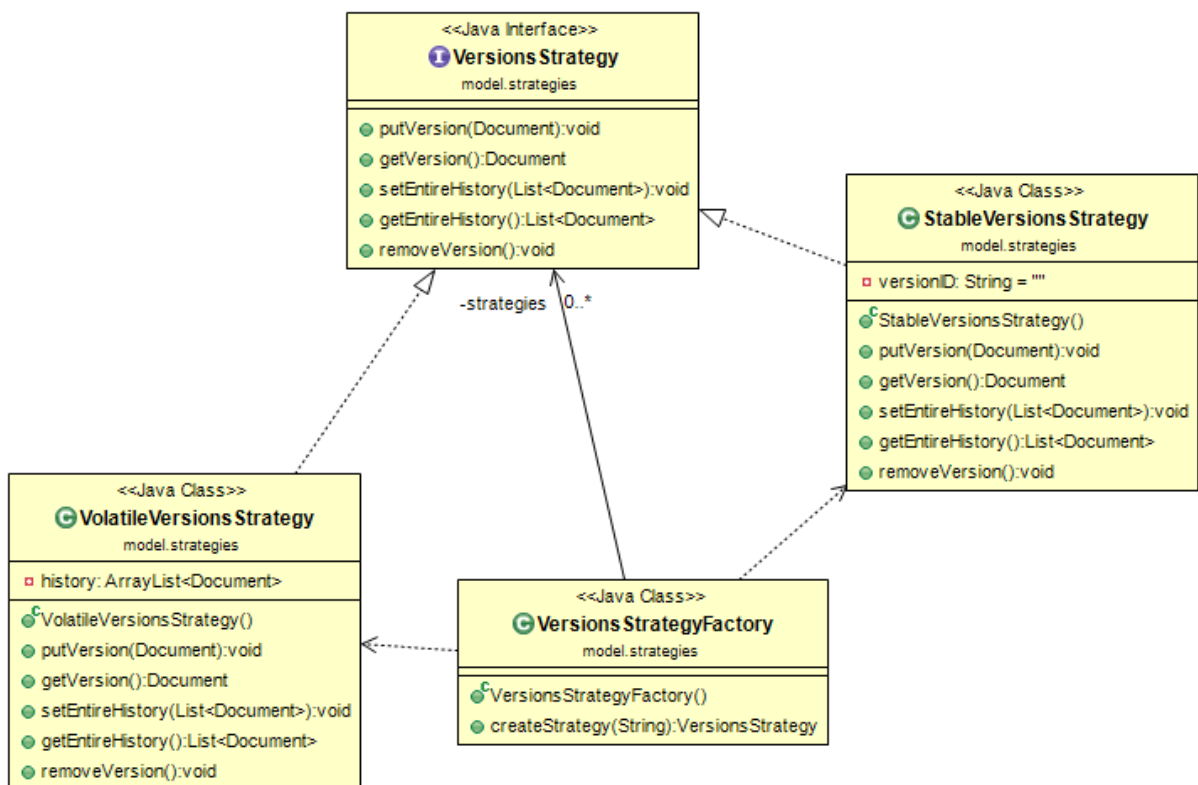- Here are the UML class diagrams of each package of the refactored application

➢ View package

\<\<Java Class\>\>
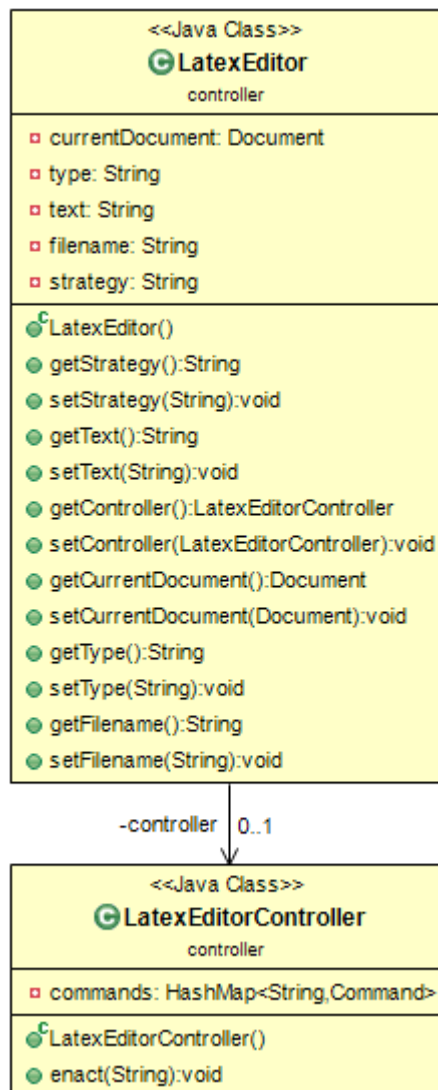**Ⓒ ChooseTemplate**
view

- ▫ frame: JFrame
- ▫ latexEditor: LatexEditor
- ▫ previous: String

- ♂ ChooseTemplate(LatexEditor,String)
- ▪ diselectRadioButtons(JRadioButton,JRadioButton,JRadioButton,JRadioButton):void
- ▪ initializeOpeningWindow ():void
- ▪ createFrame():void
- ▪ createLabel():void
- ▪ BookRadioButton(JRadioButton,JRadioButton,JRadioButton,JRadioButton):void
- ▪ ArticleRadioButton(JRadioButton,JRadioButton,JRadioButton,JRadioButton):void
- ▪ ReportRadioButton(JRadioButton,JRadioButton,JRadioButton,JRadioButton):void
- ▪ LetterRadioButton(JRadioButton,JRadioButton,JRadioButton,JRadioButton):void
- ▪ CreateButton(JRadioButton,JRadioButton,JRadioButton,JRadioButton):void
- ▪ BackButton():void

\<\<Java Class\>\>
Ⓒ OpeningWindow
view

- ▫ frame: JFrame
- ▫ latexEditor: LatexEditor

- ♂ main(String[]):void
- ♂ OpeningWindow ()
- ▪ initializeOpeningWindow ():void
- ▪ createFrame():void
- ▪ createButtonCreateNew Document():void
- ▪ createButtonOpenExistingDocument():void
- ▪ createButtonExit():void

\<\<Java Class\>\>
Ⓒ MainWindow
view

- ▫ frame: JFrame
- ▫ editorPane: JEditorPane
- ▫ latexEditor: LatexEditor

- ♂ MainWindow (LatexEditor)
- ▪ initializeMainWindow ():void
- ▪ createFrame():JFrame
- ▪ createMenuBar(JFrame):JMenuBar
- ▪ new FileMenuItem(JMenu):void
- ▪ saveFileMenuItem(JMenu):void
- ▪ loadFileMenuItem(JMenu,JMenuItem,JMenu):void
- ▪ loadHtmlFileMenuItem(JMenu,JMenu):void
- ▪ saveAsMenuItem(JMenu):void
- ▪ saveAsHtmlMenuItem(JMenu):void
- ▪ exitMenuItem(JMenu):void
- ▪ disableCommandsIfLetterOrArticle(JMenuItem,JMenu):void
- ▪ addChapterMenuItem(JMenuItem,JMenu):void
- ▪ addSectionMenuItem(JMenu):void
- ▪ addSubSectionMenuItem(JMenu):void
- ▪ addSubSubSectionMenuItem(JMenu):void
- ▪ addItemizeMenuItem(JMenu):void
- ▪ addEnumerateMenuItem(JMenu):void
- ▪ addTableMenuItem(JMenu):void
- ▪ addFigureMenuItem(JMenu):void
- ▪ StrategyCheckboxes(JMenu):void
- ▪ DisableMenuItem(JMenu):void
- ▪ RollbackMenuItem(JMenu):void
- ▪ ScrollPane():void
- ● setEditorPane(String):void

➢ Model Package

VersionsManager

<<Java Class>>
Ⓒ **VersionsManager**
model

- ▫ enabled: boolean
- ▫ strategy: VersionsStrategy
- ▫ latexEditor: LatexEditor

- VersionsManager(VersionsStrategy,LatexEditor)
- getVersionManager():VersionsManager
- isEnabled():boolean
- enable():void
- disable():void
- setStrategy(VersionsStrategy):void
- setCurrentVersion(Document):void
- getEditor():LatexEditor
- enableStrategy():void
- changeStrategy():void
- putVersion(Document):void
- rollback():void

DocumentManager

<<Java Class>>
Ⓒ **DocumentManager**
model

- ▫ DocumentManager()
- ▪ setTemplate(String):void
- getDocumentManager():DocumentManager
- createDocument(String):Document
- getTemplate(String):String

-singleDocumentManager
0..1

-templates   0..*

Document

<<Java Class>>
Ⓒ **Document**
model

- ▫ author: String
- ▫ date: String
- ▫ copyright: String
- ▫ versionID: String = "0"
- ▫ contents: String

- Document(String,String,String,String,String)
- Document()
- getContents():String
- setContents(String):void
- save(String):void
- clone():Document
- changeVersion():void
- getVersionID():String

-thisVersionsManager
0..1

➢ Model Strategies Package

<<Java Interface>>
Versions Strategy
model.strategies

- putVersion(Document):void
- getVersion():Document
- setEntireHistory(List<Document>):void
- getEntireHistory():List<Document>
- removeVersion():void

<<Java Class>>
StableVersions Strategy
model.strategies

- versionID: String = ""

- StableVersionsStrategy()
- putVersion(Document):void
- getVersion():Document
- setEntireHistory(List<Document>):void
- getEntireHistory():List<Document>
- removeVersion():void

-strategies 0..*

<<Java Class>>
VolatileVersions Strategy
model.strategies

- history: ArrayList<Document>

- VolatileVersionsStrategy()
- putVersion(Document):void
- getVersion():Document
- setEntireHistory(List<Document>):void
- getEntireHistory():List<Document>
- removeVersion():void

<<Java Class>>
Versions StrategyFactory
model.strategies

- VersionsStrategyFactory()
- createStrategy(String):VersionsStrategy

➢ Controller Package

<<Java Class>>
Ⓒ **LatexEditor**
controller
▫ currentDocument: Document
▫ type: String
▫ text: String
▫ filename: String
▫ strategy: String
⚲ᶜ LatexEditor()
● getStrategy():String
● setStrategy(String):void
● getText():String
● setText(String):void
● getController():LatexEditorController
● setController(LatexEditorController):void
● getCurrentDocument():Document
● setCurrentDocument(Document):void
● getType():String
● setType(String):void
● getFilename():String
● setFilename(String):void

-controller   0..1

<<Java Class>>
Ⓒ **LatexEditorController**
controller
▫ commands: HashMap<String,Command>
⚲ᶜ LatexEditorController()
● enact(String):void

➢ Controller Commands Package

The old design had various problems that were solved, such as:

- Large classes

  These large classes had (large) methods that their responsibilities did not match with the class name. To solve this problem we redistributed these methods to other classes so their responsibilities match their name.

- Large methods

  Some of these large methods had duplicated code or large sequences of code. In both cases we extracted smaller methods which organized the code and made it easier to read.

- Names without information

  Some variable names were not very informative so we changed them in way so the code is easier to read.

- Comments

  There were comments without any value so we deleted them to make the code more compact. Also we added some comments in order to organize the code and make it easier to read.

- Dead code

  There were methods, classes, objects and variables that were not used by the program. So we deleted them to make the code more compact.

IMPLEMENTATION

Class Name: OpeningWindow	
Responsibilities	**Collaborations**
▪ This class implements the first window that is created when the application starts	▪ ChooseTemplate ▪ OpeningWindow ▪ Latexeditorcontroller ▪ LatexEditor ▪ VersionsManager ▪ Document ▪ EditCommand ▪ MainWindow

Class Name: ChooseTemplate	
Responsibilities	**Collaborations**
▪ This class implements the window that is created when the user starts wants to create a new latex template	▪ MainWindow ▪ OpeningWindow ▪ Latexeditorcontroller ▪ LatexEditor

Class Name: MainWindow	
Responsibilities	**Collaborations**
▪ This class implements the window that is created when the user creates a Latex template or opens an existing Latex file	▪ ChooseTemplate ▪ OpeningWindow ▪ Latexeditorcontroller ▪ LatexEditor ▪ EditCommand ▪ VersionsManager ▪ Document

Class Name: Document	
Responsibilities	**Collaborations**
▪ This class keeps information aboout the opened Latex file.	▪ VersionsManager ▪ DocumentManager ▪ StableVersionsStrategy

Class Name: DocumentManager	
Responsibilities	**Collaborations**
▪ This class recognizes and sets the template of the current opened Document.	▪ CreateCommand ▪ Document

Class Name: VersionsManager	
Responsibilities	**Collaborations**
▪ This class creates controls the strategy of the Document's Strategy.	▪ All the Command Classes (except CommandFactory) ▪ LatexEditor ▪ StableVersionsStrategy ▪ VolatileVersionsStrategy

	■ VersionsStrategy

Class Name: CommandFactory	
Responsibilities	**Collaborations**
■ This class implements the factory pattern for all the commands that the user can execute in the application.	■ LatexEditorController

Class Name: ChangeStrategyCommand	
Responsibilities	**Collaborations**
■ This class executes the change strategy command.	■ VersionsManager ■ Command

Class Name: CreateCommand	
Responsibilities	**Collaborations**
■ This class executes the create command.	■ VersionsManager ■ Command ■ LatexEditor

	▪ DocumentManager
	▪ Document

Class Name: DisableVersionsManagementCommand	
Responsibilities	**Collaborations**
▪ This class executes the disable of the versions management command.	▪ VersionsManager ▪ Command

Class Name: EnableVersionsManagementCommand	
Responsibilities	**Collaborations**
▪ This class executes the enable of the versions management command.	▪ VersionsManager ▪ Command

Class Name: RollBackToPreviousVersionCommand	
Responsibilities	**Collaborations**
• This class executes the roll back to previous version command.	• VersionsManager • Command

Class Name: EditCommand	
Responsibilities	**Collaborations**
• This class executes the edit command and changes the editing pane of the main window.	• VersionsManager • Command • Document • LatexEditor

Class Name: SaveCommand	
Responsibilities	**Collaborations**
• This class executes the save command.	• VersionsManager • Command • Document

	▪ LatexEditor

Class Name: SaveHtmlCommand	
Responsibilities	**Collaborations**
▪ This class executes the save HTML file command.	▪ VersionsManager ▪ Command ▪ Document ▪ LatexEditor

Class Name: LoadCommand	
Responsibilities	**Collaborations**
▪ This class executes the load command.	▪ VersionsManager ▪ Command ▪ Document ▪ LatexEditor

Class Name: LoadHtmlCommand	
Responsibilities	**Collaborations**
▪ This class executes the load HTML file command.	▪ VersionsManager ▪ Command ▪ Document ▪ LatexEditor

Class Name: LatexEditor	
Responsibilities	**Collaborations**
▪ This class sets and distributes the attributes of the editor pane.	▪ VersionsManager ▪ Document ▪ LatexEditorController ▪ OpeningWindow ▪ ChooseTemplate ▪ MainWindow ▪ LoadCommand ▪ LoadHtmlCommand

	▪ SaveCommand
	▪ SaveHTMLCommand
	▪ EditCommand
	▪ CreateCommand

Class Name: LatexEditorController	
Responsibilities	**Collaborations**
▪ This class creates the CommandFactory and executes all the commands.	▪ LatexEditorController ▪ OpeningWindow ▪ ChooseTemplate ▪ MainWindow ▪ CommandFactory

Class Name: StableVersionsStrategy	
Responsibilities	**Collaborations**
▪ This class implements the stable version strategy which for for each document change the mechanism keeps the previous version of the document on disk storage.	▪ VersionsManager ▪ Document ▪ VersionsStrategy ▪ VersionsStrategyFactory

Class Name: VolatileVersionsStrategy	
Responsibilities	**Collaborations**
▪ This class implements the volatile version strategy which for each document change the mechanism keeps the previous version of the document in a main memory list of subsequent document versions .	▪ VersionsManager ▪ Document ▪ VersionsStrategy ▪ VersionsStrategyFactory ▪ OpeningWindow

Class Name: VersionsStrategyFactory	
Responsibilities	**Collaborations**
This class implements the factory pattern for all the strategies that the user can select to use in the application.	- VolatileVersionsStrategy - StableVersionsStrategy - VersionsStrategy