Change request log

1 Team

Team Name: Swathy & Rachit

Swathy Hari Prasad : Change request log tracking

Rachit Dalal: Code inspection, Location identification of concept location and fixing the issue

2 Change Request

Change Request ID: #ps2

The Merge module throws an exception upon attempting to merge page ranges that intersect (see Figure 7 and Figure 8). You are to fix this issue by allowing intersection of ranges during the merging operation.

Code Location in Github:

parent location of our branch:

https://github.com/SwathyAravind/cs515-001-s20-Swathy-Rachit-pdfsam/tree/ps2-TheMergeModuleShouldNotThrowAnException-rachitSwathy

Commit 1: To show the diff

https://github.com/SwathyAravind/cs515-001-s20-Swathy-Rachit-pdfsam/commit/545009335f58e4cf0781d22bd5ece9e3d9914abe

Note: In our first commit we accidently added comment as ps1 - the latest commit below has the updated message and pom.xml settings .

Commit 2: To show the diff

https://github.com/SwathyAravind/cs515-001-s20-Swathy-Rachit-pdfsam/commit/8c6dc5960553bdbe77e60c41d4f1f50d59e6fc6a

3 Concept Location

The following table below describes each step we followed when performing concept location for this change request. This table has various approaches such as IDE features that we used, search queries, class navigation, system workflow and execution, debugging, etc.

Step #	Description	Rationale
1	We ran the pdfsam to check how it is working which helped us to figure out the functionality and control flow.	The rationale behind this is to get an idea about the location where we can change the code and probable areas to look into code.
2	We put debug points in the code and also put some log statements to identify the exact location of the fix. Also we started application in debug mode to hit the debug point.	To get the exact concept location of the particular bug.

3	After debugging we found org.pdfsam.merge and org.pdfsam.core package and its code base which have files that I need to change. Files include 1> SelectionTableRowData.java 2> MergeSelectionPane.java 3> ConversionUtils.java 4> AbstractParametersBuilder.java	We also then started identifying the probable location in the code base to and from those files where we can change the code to make it actual fix.
4	We began looking through the org.pdfsam.pdfsam-fx and org.pdfsam.pdfsam-gui package for files that may be related to rendering and the potential pageRange related stuff. PageRangesColum.java,SelectionTableColumn.java and SelectionTableRow.java.	By going through this we identify that we can actually change around this. But, later we decided not to change this rather the class which calls this should be changed to accommodate the fix.
5	Finally we realized that MergeSelectionPane.java in the org.pdfsam.merge package should be changed. As this is the code related to the merge module so we can make changes that do not have global side-effects across the application. Therefore, we did not change ConversionUtils.java file.	As we got the exact concept location we were going through the lines of code and especially apply function which performs the operation.
6	MergeSelectionPane.java file is the "concept location" for us.	This is the place we made modification for the change request.
7	We have also added a method to SelectionTableRowData.java file currentPageSelection() which returns the page selection for each raw input.	This method acts as a helper method for complete functionality.

Time spent (in minutes): 212

4 Impact Analysis

The following table describes each step you follow when performing impact analysis for this change request. This contains the set of the classes that are related to the marked in the concept location and some of them which are unchanged, changed or some may have side effects due to ripple effect depending on their impact which we have estimated.

Step #	Description	Rationale
1	We have searched for the different functions in the MergeSelectionPane.java file and we have understood that apply(MergeParametersBuilder, Erro, Consumer<String>) function needs to change.	This place is the place which I think is the right fix to achieve the change request fix.
2	Also we have added a helper method into SelectionTableRowData.java class.	We added to get the currently entered paged for the given input pdf file per row.

Time spent (in minutes): 45

5 Prefactoring (optional)- Not Done

Using the table below, describe each step you follow to prefactor the code. Include as many details as possible, including the refactoring operations used (e.g., move method, extract class, etc.) and classes/methods/fields that were modified, added, removed, renamed, etc.

Make sure you time yourselves when going through this process and provide the total time spent below.

Step #	Description	Rationale

Time spent (in minutes): x

6 Actualization

Use the table below to describe each step you followed when changing the code. Include as many details as possible, including why classes/methods were modified, added, removed, renamed, etc.

Step #	Description	Rationale
1	We are calling the helper method currentPageSelection() from SelectionTableRowData.java which we created to get current raw's page range and then we are splitting it using comma.	The rationale behind this is to extract range at the beginning and apply it to the builder.addInput function to allow pdf intersection range addition at the end of the pdf.
2	This is basically for each input we are generating new pdf and adding it at the end of the file.	To fix the intersection range exception.
3	If we have uploaded a pdf but no ranges will be provided it should create a pdf with all the pages which was there in the original pdf. To maintain this we have added else condition in MergeSelectionPane.java file.	To not affect the actual functionality we did this
4	MergeSelectionPane.java file and we have understood that apply(MergeParametersBuilder, Erro, Consumer<String>) function needs to change with the ranges we provided	This place is the place which I think is the right fix to achieve the change request fix. Because here we are creating new pdf for the ranges that we are getting.
5	For this use case we have not created unit cases.	We have thoroughly tested the functionality after fix. We did manual testing to verify this fix.

Time spent (in minutes): 88

7 Post Factoring (optional) - Not Done

Use the table below to describe each step you followed to postfactor the code. Include as many details as possible, including the refactoring operations used (e.g., move method, extract class, etc.) and classes/methods/fields that were modified, added, removed, renamed, etc.

Make sure you time yourselves when going through this process and provide the total time spent below.

Step #	Description	Rationale

Time spent (in minutes): x

8 Validation

the below table describes validation activities (e.g., testing, code inspections, etc.) we performed for this change request. it includes the description of each test case, the result (pass/fail) and its rationale and after the actualization.

Step #	Description	Rationale
1	Test case defined: ● Run the mvn project. ● Open the merge menu. ● We have added a pdf having 9 pages and specify intersecting ranges comma separated. ● Inputs: We specify the range as 2-8, 5-7 Expected output: The merged document should have pages 2, 3, 4, 5, 6, 7, 8, 5, 6, 7	This is the regular expected behavior. The test passed. No exception observed. On the click of the run button we got the output with pdf having 2, 3, 4, 5, 6, 7, 8, 5, 6, 7 pages. Test result: Passed
2	Test case defined: ● Run the mvn project. ● Open the merge menu.	We want all the pages from start - end of the file to be appended to the new pdf.

	• We have added pdf with 11 pages and specify the range 2-9 and then intersection range without specifying end range • Inputs: We specified the range with 2-9, 5-. Expected Output: document should have 2,3,4,5,6,7,8,9,5,6,7,8,9,10,11	The test passed.
3	Test case defined: • Run the mvn project. • Open the merge menu. • We add two different pdfs, and both input had intersection range • Inputs: The range for the first pdf was 7-9, 8. The second pdf range was 1-6, 4 Expected output: Final pdf should have pages 7,8,9,8 for the first pdf followed by pages 1,2,3,4,5,6,4 for the second pdf.	For multiple documents that we are inserting, it should work if we provide an intersecting range for them. Merge pdf should have all the ranges. The test passed.
4	Test case defined: • Run the mvn project. • Open the merge menu. • Add in two pdf's which has been shown into A2-001.pdf. • Also add page ranges provided there. • Merge the existing document. • Validate the merge document should have all the should have all the pages from both the pdf and the ranges provided as a input.	The overlapping ranges should not throw and exception and it should have document generated with the provided ranges. The test passed.

Time spent (in minutes): 45

9 Timing

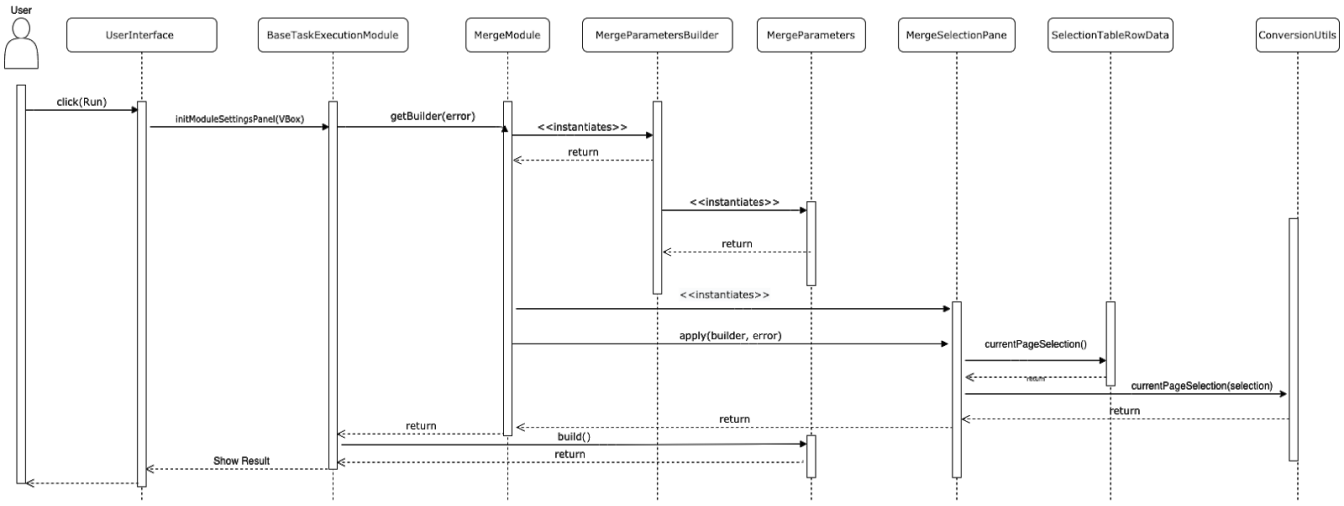
Summarize the time spent on each phase.

Phase Name	Time (in minutes)
Concept location	212
Impact Analysis	45
Prefactoring	0
Actualization	88
Postfactoring	0
Verification	45
Total	390

10 Reverse engineering

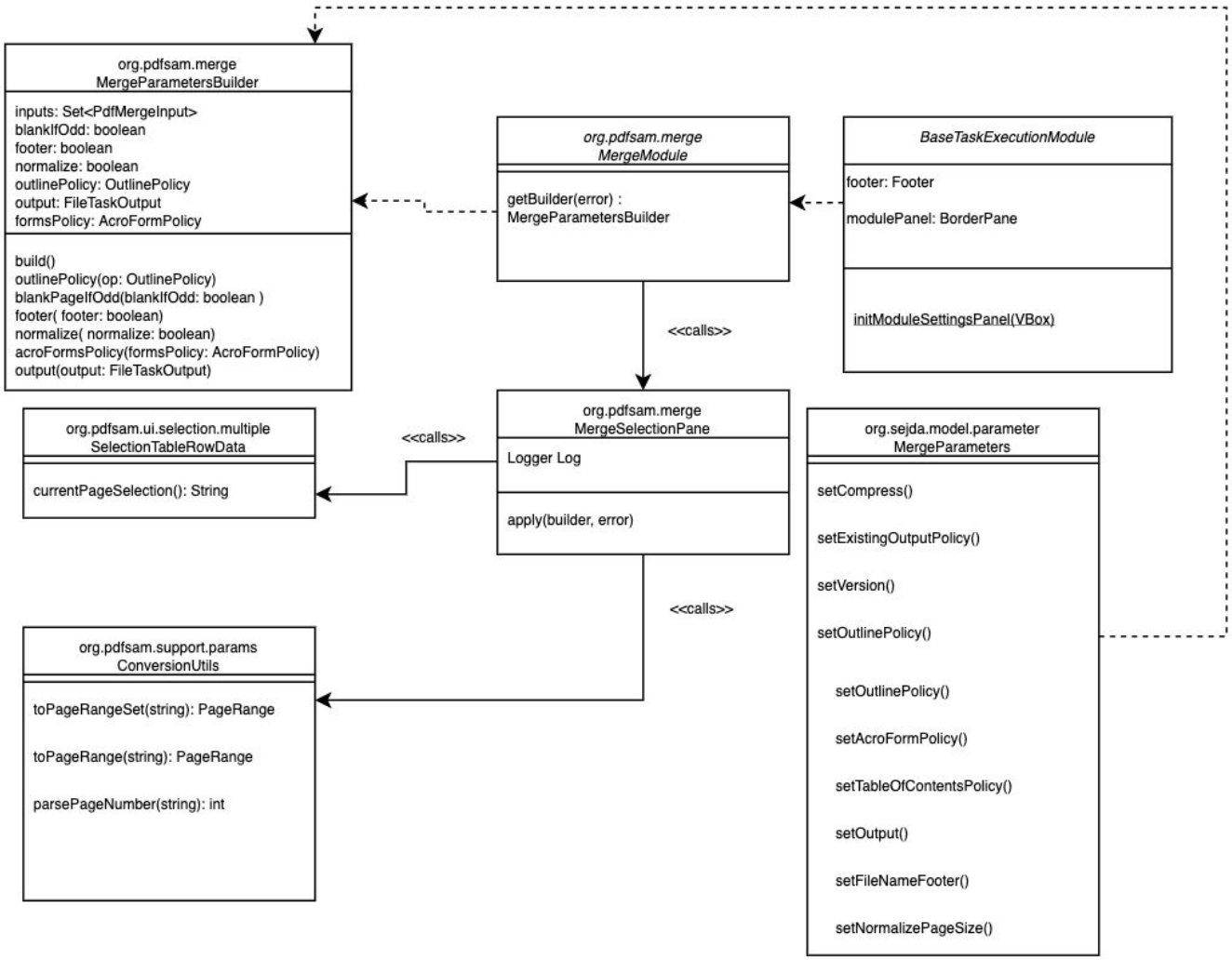
UML Sequence Diagram

Create a UML sequence diagram (or more if needed) corresponding to the main object interactions affected by your change.



UML Class Diagram

A UML class diagram of the classes which are visited during the task of identifying concept location. This includes the different types of relationship between the classes. (aggregation, inheritance, composition, dependency, etc..)



11 Conclusions

This change request was not easy as initially we are trying to put debug statements in the file and identifying concept location. That strategy was not so effective. After that we tried to run in debug mode to put some debug points and try to understand the workflow. With this strategy, we are able to identify some of the potential concept locations but

those were ending up in the class files. After talking with Dr. Moreno and with her advice, I tried again in the module package and I got the concept location finally. The change that we did is not big though. We have not utilized/used any of the tools to identify the concept location.We added one helper method to extract the page ranges from the provided ranges. We added changes to apply() method in MergeSelectionPane.java file and that is working fine without exception. After making changes we have done certain manual testing and verification part to make sure it should not break. To sum up, after taking the advice from the professor and the debugging technique has helped us a lot.

Classes and methods changed:

- pdfsam-merge/src/main/java/org/pdfsam/merge/MergeSelectionPane.java
    - void apply(MergeParametersBuilder builder, Consumer<String> onError) -- Modified
- pdfsam-merge/src/main/java/org/pdfsam/ui/selection/multiple/SelectionTableRowData.java
    - public String currentPageSelection() -> returns string of the page ranges -- Added