Change request log

1 Team

Team name:

Fari-Marylou

Team members and roles:

- Farzaneh Kadkhodaie

Role: Investigate and fix ps-2. Implement the code and document the investigation and steps taken to solve the problem. Complete the change request log form. Provide UML diagrams.

- Marylou Nash

Role: Provide support. Answer the questions and collaborate to design the solutions. Review and verify the fix.

2 Change Request

- Change request ps-2:

Currently, the merge module is not able to merge page ranges that intersects, and then it throws and exception upon attempting to merge page ranges that have overlap.

The ps-2 required us to fix this issue and let merge page ranges that intersect. Additionally, the merge should be able to put redundant/duplicate pages (overlaps) in the final created .pdf file in the same order provided by users. Moreover, we noticed that original PDFSam version automatically removes duplicated pages through merging, as it stores them in set data structure and set automatically remove its duplicated items.

3 Concept Location

PDFSam source code packages are very well named, organized and structured. So, for the concept location, the first step to find the relevant source code was to check different packages (names) and find which package is more relevant to the change request.

Step #	Description	Rationale
1	*I ran the system on the version v4.0.5.*	*To open pdfSam application.*
2	*I interacted with the system. After pdfsam is opened, I went to the "Merge" module and start to play with it.*	*To get familiar with merge module and its behavior and features and try to provide a merge through intersection situation to inspect how pdfsam behaves.*
3	*I observe that pdfsam throws an exception when there is an overlap in page ranges, e.g. [1-7, 3-6]. So, to fix this issue, we decided to propose an approach in such way that whenever the page range includes "-", we flatten the range to their pages. (our first approach)*	*Because we tried this approach and seems it worked. In other words, flattening the page range allows merge through intersection without throwing any exception.*

	For instance, if we have [1-7, 3] in page range, we flatten and convert it to 1,2,3,4,5,6,7,3.	
4	*At this point, I decided to search for the originate of this throw exception and to find where intersection (overlap) in page ranges is checking and detecting.*	
5	*I went to the source code to find the package in associated with merge module! The package "pdfsam-merge" seems where the merge module is implemented. The package includes 4 different classes, including* • *MergeModule.java* • *MergeOptionsPane.java* • *MergeParameterBuilder.java* • *MergeSelectionPane.java*	
6	*Inspecting these four classes did not lead me to any useful function that is dealing with intersection and merge with overlap. So, I decided to use IDE's search tools to see find any relevant function, classes related to page range.*	
7	*I searched for some keywords, like" page range", "range", etc.*	*Since this query is what exactly I am looking for to find any function/classes that checked intersection.*
8	*Among all different functions and classes listed in search result, I clicked on toRangePage(string) and it navigated me to "conversionUtils.java" class.*	*This function seems very relevant and promising, specially, because its input is string type, which seems fairly matched to what we entered to page ranges box over performing merging.*
9	*I inspected class carefully to locate where I should start to flatten my page range.*	
10	*I realized that toPageRange(String) the function that is checking page range to find if it includes any page range with hyphen or "-".*	
11	*Then, I started to find who is calling this function. I realized that toPageRangeSet(string) function is calling toPageRange(String).*	
12	*I found that toPageRangeSet(string) function is where exactly the page ranges filed (located in*	*Because the function's input (Selection variable) contains all page ranges entered by users.*

	merge module window) is receiving and started to be split and processed. Therefore, here is when my new implementation to flatten the page range field should be added.	
13	*I added a function called removedDash(string) to check the selection variable and flatten its ranges, if it contains any hyphen "-" in its page ranges.*	
14	*This function was working fine, but we noticed that through flattening the range, we are nothing getting the exception anymore, but the overlap pages won't be added to the end merge file.* *For example, for [1-4, 3], the final created merged file only contains page 1, 2,3,4. However, we expect to have page 1,2,3,4,3!*	
15	*This seemed like a confusion and unclear concept. We Could fixed not to get throw exception but overlap pages are not appeared/merged to the final created file. We decided to talk with Dr. Moreno for further clarification.*	
16	*Dr. Moreno confirmed that this change request extremely required to contain the overlap/duplicate pages in the same provided by users. Therefore, we decided not to proceed this approach* ***(first approach failed)****.*	*because at this point, we understood that our approach did not satisfied this requirement and it failed. Therefore, we decided to start to find another solution.*
17	*After discussing a lot with Marylou and observing multiple solutions, we decided to propose our* ***second approach*** *as following:* *As intersection with previous page ranges cause the problem, whenever a file has more than one page range to merge (e.g. file X with page range [1-4, 3]) we create a copy of our file for each page rage and send them separately for merge.* *Example: file X with page range [1-4, 3] will be converting to file X [1-4] and file X [3].*	
18	*I again started to inspect 4 classes provided in pdfsam-merge package carefully to find and locate where exactly I can apply this strategy.*	*To find and approach the solution.*

19	*After inspecting mergeParameterBuilder.java class carefully, I realized that addInput(input:pdfMergeInput) method is where exactly pdf merge input files are added to a set to be merged together!* *Adding all pages to a set data structure will remove the duplicated pages and that cause problem.*	
20	*I modified addInput(input:pdfMergeInput) function in a way that it checked each input pageSelection field, and if its length is greater than one, it creates a copy of file (with same source) for each page range and add to the set.*	*This approach ensures that all the page ranges provided for a single file will merged separately, and hence the final pdf merged file contains all the pages, including the repeated pages.*

Time spent (in minutes): 300

4 Impact Analysis

Step #	Description	Rationale
1	*Start to list all the functions, classes, etc. used by addInput(input:pdfMergeInput) function.*	*To track the classes that could be impacted by the change.*
2	*After inspecting my modifications carefully, I realized that my changes (extending addInput function) are implemented completely internal, and they did not impact any other classes. Therefore, extending addInput(input: pdfMergeInput) method does not have any impact on other classes.*	*Since all changes contained inside the class.*
3	*No class needs to be inspected for impact analysis for ps-2.*	

Time spent (in minutes): 20

5 Prefactoring (optional)

Step #	Description	Rationale
1	*No prefactoring was done.*	*This change might just require a simple method to receive pdf merge input file and return all the copies*

		of single file, with the same source but different page selection.

Time spent (in minutes): 0

6 Actualization

Step #	Description	Rationale
1	*After finding the desired class and function to modify (through concept location) we started to modify/extend addInput(input: pdfMergeInput) class.*	*This class is marked to be changed to provide ps-2.*
2	*I extended the function to:* • *check pageSelection field for each input* • *If its length is greater than one, it creates a copy of file (with same source) for each range in the file page selection and add that to a set data structure to be merged in final merge file.* • *Otherwise, file has only one page range in its page selection field, and it will be added to final merge file.*	
3	*After confirming code on my system, push them to my local branch on github to be verified by my teammate.*	

Time spent (in minutes): 40

7 Postfactoring (optional)

Step #	Description	Rationale
1	*No postfactoring was done.*	*I would say changes were fairly simple and required no big modifications after prefactoring.*

Time spent (in minutes): 0

8 Validation

Step #	Description	Rationale

1	*I provided manual testing for this part.*	*To make sure system satisfies the change request requirements and it works properly.*
2	*First, I run the system, navigate to merge module and tried to perform merge for different situations mentioned below.*	*To make sure updated merge madule works properly.*
3	***Test case defined:*** *Try to merge a range of pages in a single file upon an intersection.* ***Inputs:*** *File X.pdf (optional file) with page ranges [1-4, 3]* ***Expected output:*** *PDFSam_merge.pdf file contains 5 pages from page 1 to 4 and 3.*	*I expected to see a created pdf file including 5 pages, i.e. 1,2,3,4,3.* *The test passed.*
4	***Test case defined:*** *Try to merge a range of pages in two files, upon an intersection.* ***Inputs:*** *File X.pdf (optional file) with page ranges [2-7, 4-6]* *File Y.pdf (optional file) with page range [2-4]* ***Expected output:*** *PDFSam_merge.pdf file including 12 pages with the same order mentioned in page range field.*	*I expected to see a created pdf file contains 12 pages, i.e. 2,3,4,5,6,7,4,5,6 (all from file X) and ,2,3,4(All from file Y).* *The test passed.*
5	***Test case defined:*** *Try to merge a range of pages in two files when upper bound and lower bound of the range is not specified. (file Y has 6 pages).* ***Inputs:*** *File X.pdf (optional file) with page ranges [2-7,-6]* *File Y.pdf (optional file) with page range [2-,3]*	*I expected to see a created file contains 18 pages, i.e. pages 2,3,4,5,6,7,1,2,3,4,5,6 (from file X) and pages 2,3,4,5,6,3 (from file Y).* *The test passed.*

	Outputs: *PDFSam_merge.pdf file including 18 pages.*	
6	*All tests are passed, therefore, seems code is ready to be metged to master branch.*	

Time spent (in minutes): 45

9 Timing

Phase Name	Time (in minutes)
Concept location	300
Impact Analysis	20
Prefactoring	0
Actualization	40
Postfactoring	0
Verification	45
Total	405

10 Reverse engineering

UML sequence diagram:

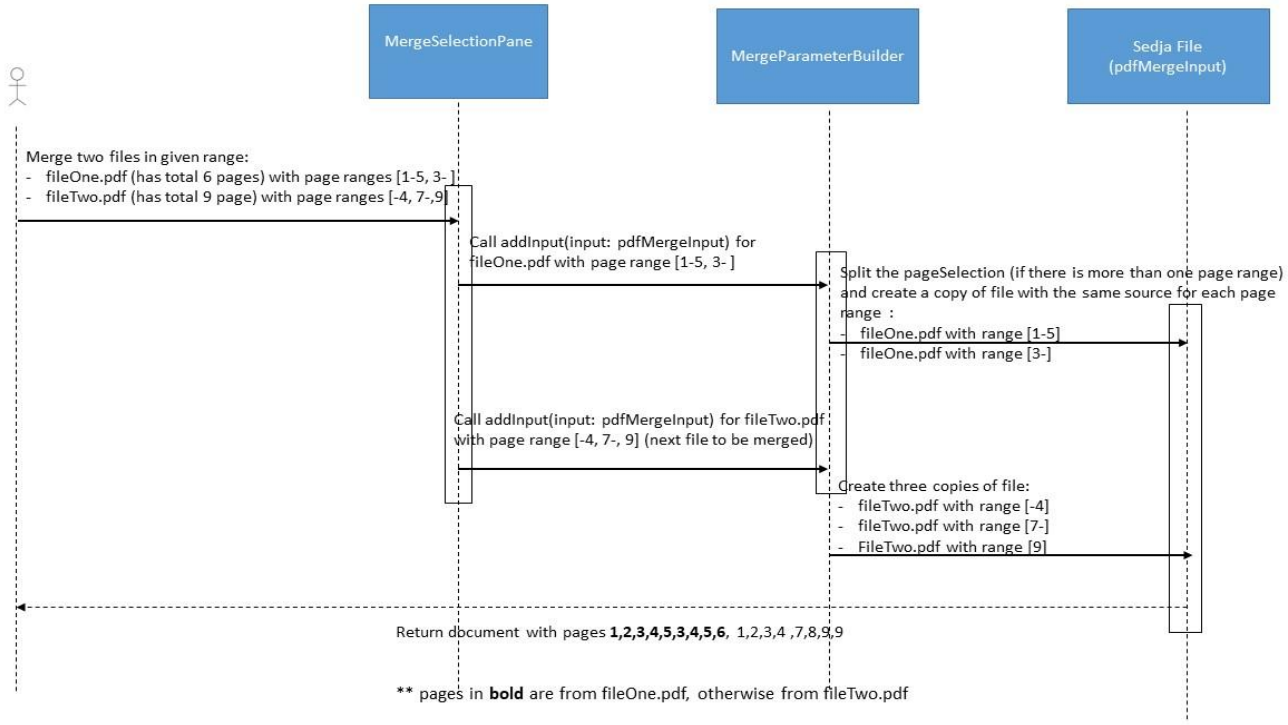

Figure1. UML Sequence Diagram for ps-2

Partial UML class diagram:

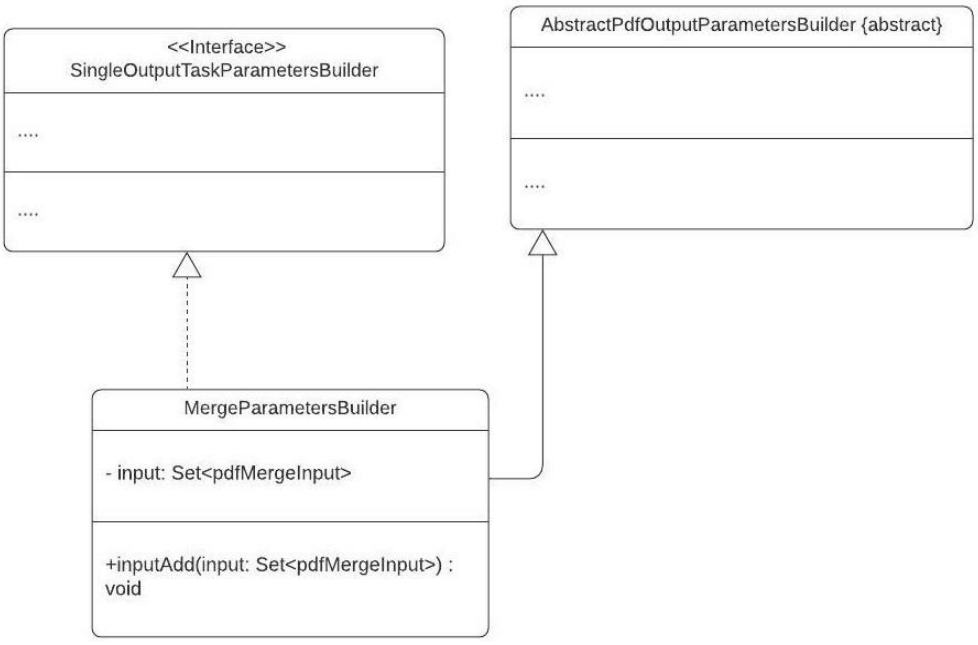

Figure2. Partial UML Class Diagram for ps-2

11 Conclusions

The code for PDFSam is very well organized and unique, listed in different well-named packages for each task. Tracking codes to locate the right class/place to implement new codes (for concept location phase) was quite challenging, as most classes are interacting with sedja objects to implement their features, and we did not want to modify/ extend sedja objects and methods.

Moreover, the description provided for ps-2 was not quite clear and precise, and that led us to propose a wrong approach to follow (the first approach explained in concept location), which consumed a big portion of our time.

Also, impact analysis phase was fairly straightforward, because the provided changes were entirely internal with no interfere with any other classes. Similarly, actualization phase was quite simple and short, as there was only one method, i.e. addInput(input: pdfMergeInput), to be extended. Also, using debugger in this phase was dramatically helpful to track our changes and to make sure we are taking right steps to approach the solution.

Furthermore, validation and testing performed manually for this change request (different tested cases are listed in validation phase). We tried to merge file upon different page ranges and observe the created merged file contens to make sure that our expected output is the same as actual output.

Classes and methods changed:

- pdfsam-merge/src/main/java/org/pdfsam/merge/MergeParametersBuilder.java
    - void AddInput(input:pdfMergeIput)