



UTM
UNIVERSITI TEKNOLOGI MALAYSIA

SECJ 2253 Requirements Engineering & Software Modeling

Sem.2 2020/2021

PHASE 3: REQUIREMENTS MODEL DOCUMENT (FUNCTIONAL, DATA, BEHAVIORAL PERSPECTIVES)

Mipanzu Online File Management System

<Mipanzu>

Team Members:

- | | |
|--------------------------------------|-----------|
| 1. HUSNY MUSHARRAF BIN SHAMSUL KAMAL | A19EC0051 |
| 2. TAN CHIAW TORNG | A19EC0167 |
| 3. SEE WEN XIANG | A19EC0206 |
| 4. WAN LUQMAN BIN WAN ZULLKEFLI | A19EC0209 |

Table of Contents

Item	Page No	Prepared by	Moderated by
Introduction	1	Husny, Wan Luqman	Chiaw Torng
Functional Perspective: Use Case (UC) Documentation (Diagram – UML UCD)	2	Wen Xiang, Chiaw Torng	Chiaw Torng
Functional Perspective: Use Case (UC) Documentation (Specification – UCS) & UML Activity Diagram (AD)		All members	
UC001 <Open new file>UCS & AD	3-6	Wan Luqman	Chiaw Torng
UC002 <Close file> UCS & AD	7-10	Husny	
UC003 <Track file location> UCS & AD	11-12	Wan Luqman	
UC004 <Report the loss of file>UCS & AD	13-15	Husny	
UC005 <Dispose of file>UCS & AD	16-20	Chiaw Torng	
UC006 <Borrow file>UCS & AD	21-24	Wen Xiang	
UC007 <View file access record> UCS & AD	25-26	Chiaw Torng	
UC008<Detect the late return of file> UCS & AD	27-28	Wan Luqman	
UC009 <Return file> UCS & AD	29-30	Wen Xiang	
Data Perspective: UML Domain Class Diagram (DCD)	31	All members	Chiaw Torng
Behavioral Perspective: State-Transition Diagram (STD)	32-36	All members	Chiaw Torng
Conclusion	37	All members	Chiaw Torng
References	37	All members	Chiaw Torng

Business Process Mapping to Use Cases

Which Business Process related to the use case	UC ID	UC	Use case specification	PIC
Handle the open file procedure	UC001	Open new file	<i>Use Case Specification for UC001 <Open new file></i>	Wan Luqman
Handle the close file procedure	UC002	Close file	<i>Use Case Specification for UC002 <Close file></i>	Husny
Handle the borrow file procedure	UC003	Track the file location	<i>Use Case Specification for UC003 <Track file location></i>	Wan Luqman
	UC006	Borrow file	<i>Use Case Specification for UC006 <Borrow file></i>	See Wen Xiang
Handle the return file procedure	UC009	Return file	<i>Use Case Specification for UC009 <Return file></i>	See Wen Xiang
Handle the dispose file procedure	UC005	Dispose of file	<i>Use Case Specification for UC005 <Dispose of file></i>	Tan Chiaw Torng
Monitor file access	UC004	Report the loss of file	<i>Use Case Specification for UC004 <Report the loss of file></i>	Husny
	UC007	View file access record	<i>Use Case Specification for UC007 <View file access record></i>	Tan Chiaw Torng
	UC008	Detect the late return of file	<i>Use Case Specification for UC008 <Detect the late return of file></i>	Wan Luqman

Remark: *Italic text* is cross-reference hyperlink.

Use cases mapping with Activity, and Sequence Diagram, State, Domain

Use Cases	Activity Diagram	Sequence Diagram	State diagram (ONE) Give the ONE controller class in sequence diagram which you you will represent in state diagram	Domain Model Give the name of entity class in sequence that which you will include in domain model
UC001	<i>Activity Diagram for UC001 <Open new file></i>	<i>Figure 2.1.1: Sequence Diagram for UC001 <Open new file></i>	<i>State-Transition Diagram for <File Class></i>	File SAOfficer RUClerk Letter
UC002	<i>Activity Diagram for UC002 <Close file></i>	<i>Figure 2.2.1: Sequence Diagram for UC002 <Close file></i>	<i>State-Transition Diagram for <File Class></i>	File SAOfficer RUClerk
UC003	<i>Activity Diagram for UC003 <Track file location></i>	<i>Figure 2.3.1: Sequence Diagram for UC003 <Track file location></i>	<i>State-Transition Diagram for <File Class></i>	File SAOfficer RUClerk
UC004	<i>Activity Diagram for UC004 <Report the loss of file></i>	<i>Figure 2.4.1: Sequence Diagram for UC004 <Report the loss of file></i>	None	borrowFileRecord File SAOfficer RUClerk
UC005	<i>Activity Diagram for UC005 <Dispose of file></i>	<i>Figure 2.5.1: Sequence Diagram for UC005 <Dispose of file></i>	<i>State-Transition Diagram for <disposedFileHandler Controller></i> <i>State-Transition Diagram for <FileDisposedRecord Class></i>	fileDisposeRecord File SAOfficer RUClerk
UC006	<i>Activity Diagram for UC006 <Borrow file></i>	<i>Figure 2.6.1: Sequence Diagram for UC006 <Borrow file></i>	<i>State-Transition Diagram for <borrowFileHandler Controller></i>	borrowFileRecord Staff
UC007	<i>Activity Diagram for UC007 <View file access record></i>	<i>Figure 2.7.1: Sequence Diagram for UC007 <View file access record></i>	<i>State-Transition Diagram for <BorrowedFileRecord Class></i>	borrowFileRecord SAOfficer RUClerk
UC008	<i>Activity Diagram for UC008 <Detect the late return of file></i>	<i>Figure 2.8.1: Sequence Diagram for UC008 <Detect the late return of file></i>	<i>State-Transition Diagram for <BorrowedFileRecord Class></i>	borrowFileRecord Staff
UC009	<i>Activity Diagram for UC009 <Return file></i>	<i>Figure 2.9.1: Sequence Diagram for UC009 <Return file></i>	<i>State-Transition Diagram for <BorrowedFileRecord Class></i>	borrowFileRecord Staff

Remark: *Italic text* is cross-reference hyperlink.

1. Introduction

The purpose of the RMD created is used in order for us to describe all the requirements elicited from the file management system in JKSNJ. This is because we want to create and develop a new file management system that will be changed from manual system to online system. RMD is best used for us to document the functional requirements of the system that need to be developed. The document will specify all the user expectations toward what the software is able to do. In the RMD, we will use UML Use Case diagram, UC specifications, UML activity diagram, UML domain class diagram, UML state diagram in order to get a more detailed understanding towards each process in the system. All the documents related towards the file management system also will be recorded in the RMD. It is also easy for us to see clearly all the issues and improvements that can be made towards each process. All the requirements gathered in the RMD are elicited from the stakeholders of JKSNJ on 11 April 2020. The parties involved for this RMD are all staffs of JKSNJ including the Clerk of Record Unit (CRU) and Syariah Assistant Officer (SAO).

The software product is Mipanzu Online File Management System to replace the current manual file management system from JKSNJ. This file management system has the function of keeping track of the location of the files and records. So, when the user wants to borrow the files, they need to key in the code of the file. All file access information including the user identity, date and time of borrowing and returning and file code will be stored online therefore it can replace the physical version of "Kertas Minit". Therefore, it is easier for SAO as they can know who borrowed the file and can keep track of the file location.

All the staff need to log in with passwords to verify their identities. Different groups of staff can access different types of files according to their security levels. When the staff return files, the system will record the date and time of return. The staff can also report the loss of file in the system. The system will automatically notify the officers about the late return of files. The system can also be used to dispose of files, open new files and close files. When they want to dispose of files, they need the approval from Arkeb Negara Malaysia (ANM). The system will notify the officers of the files that shall be disposed of according to Jabatan Pelupusan Record (JPR).

On the other hand, viewing files and store files does not need approval from ANM, instead the CRU only needs to update the file status in the system. This system brings benefits to all JKSNJ staffs, CRU and SAO where they can manage the file records and the daily procedures like borrowing, returning, disposal and classification of files more efficiently. All the file records can be backup to secure those records and prevent them from missing. Our goal is to produce an online file management system that provides one-stop services to all JKSNJ staff with an objective to offer a path for the users to have an easier work in managing the files and records while maintaining its integrity.

2. Functional Perspective: Use Case (UC) Documentation (Diagram – UML UCD)

The system features include several diagrams that helps us to create a good to-be system in the future. There will be several diagrams created and all of them are used to explain all the requirement and process needed for Mipanzu File Management System. The diagram that will be created are use case diagram to show interaction between user and all the use case in the system, activity diagram to documents the action sequences in the use case, domain model to show all the class with their attributes in the system and state machine diagram to show the state of the class when there is a trigger that made the state of the class to change if any. For every use case in the use case diagram, we will create use case description and sequence diagram to see the flow and process for every use case.

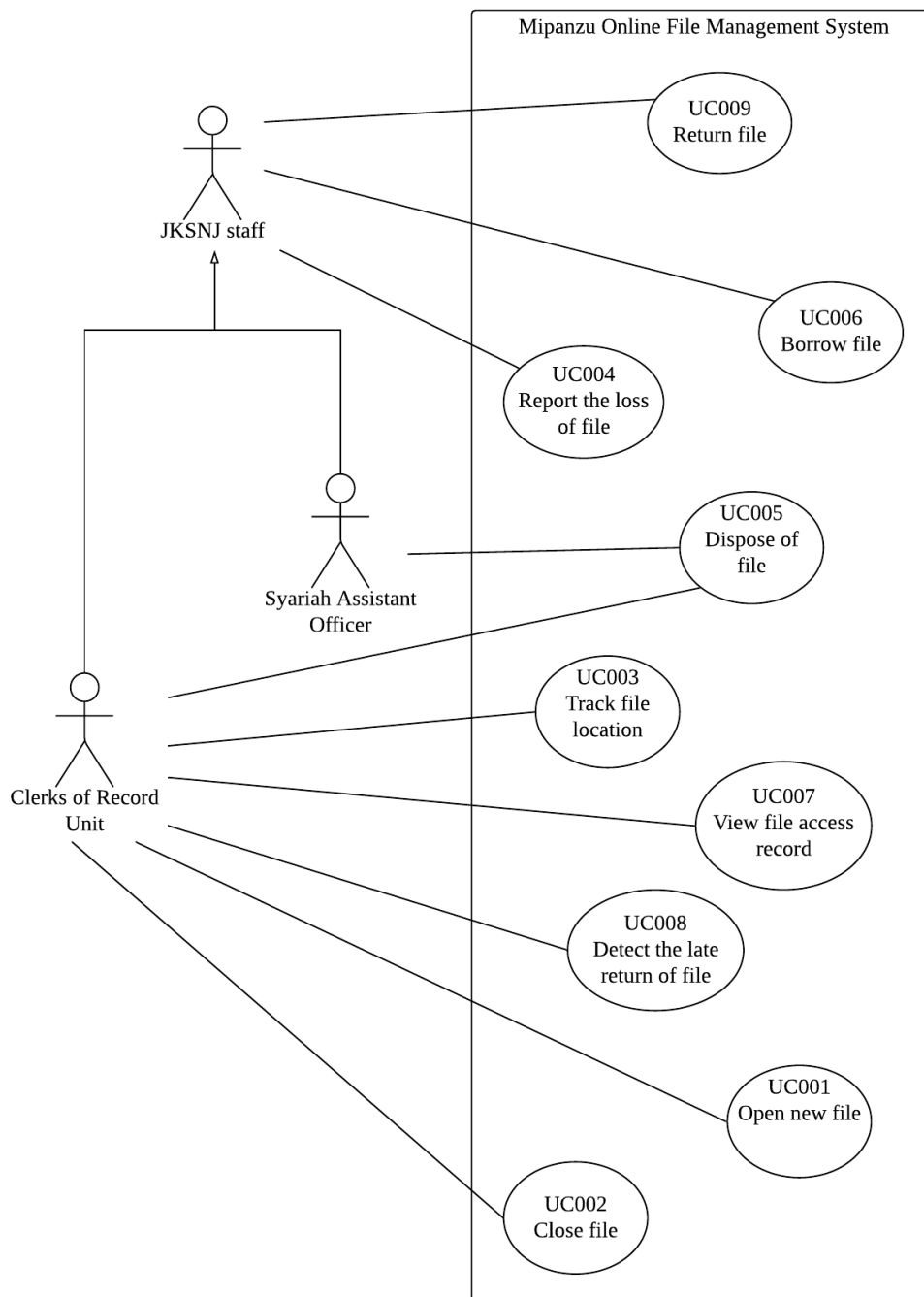


Figure 1: Use Case Diagram for Mipanzu Online File Management System

2.1 Functional Perspective: Use Case (UC) Documentation (Specification – UCS) & UML Activity Diagram (AD)

Table 2.1 Use Case Specification for UC001 <Open new file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal and alternative flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC001		
Use Case Name	Open new file		
Created by:	Luqman	Last Updated by:	Luqman
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit to open new file in Mipanzu Online File Management System.		
Actor(s)	Clerks of the record unit		
Pre-condition(s)	A valid clerk of the record unit is logged on to the system		
Normal Flow(s)- NF	<ol style="list-style-type: none"> 1. The clerk of the record unit keys in the file code of the file. 2. The system automatically checks whether the file with the file code entered exists. [A1] 3. The system displays file does not exist message. 4. The clerk of the record unit keys in the file code, security level of file, room number and shelf number of file storage space. 5. The system registers the new file with the file information entered by the clerk of the record unit. 6. The clerk of the record unit keys in the reference number, title and judge note of a letter. 7. The system registers the letter in the letter database. 8. The system adds the letter information into the file created in the file database. 9. The system updates the file database. 		
Alternative Flow(s) - AF	<p>A1: File exists already</p> <ol style="list-style-type: none"> 1. The system displays the file already exist message and the file information. 2. Clerk of record unit keys in the reference number of the letter. 3. The system checks whether the letter with the reference number entered exists. 		

	<p>4. If the letter does not exist resume step 6</p> <p>5. The system display letter already exists and the letter information.</p> <p>6. The use case ends.</p>		
Post-condition(s)	A new file is successfully created by the Clerks of the record unit.		
Related Requirement (s):	ID	Requirement	Priority
	CR UC001-01	The system shall follow the Malaysian standard for Record Management guideline or acted by Arkib Negara Malaysia for the file code	Basic
	CR UC001-02	The system shall allow the user to open the file during the work hour only	Basic
	CR UC001-03	The system shall allow the user to open the file using any devices such as desktop and mobile devices	Basic

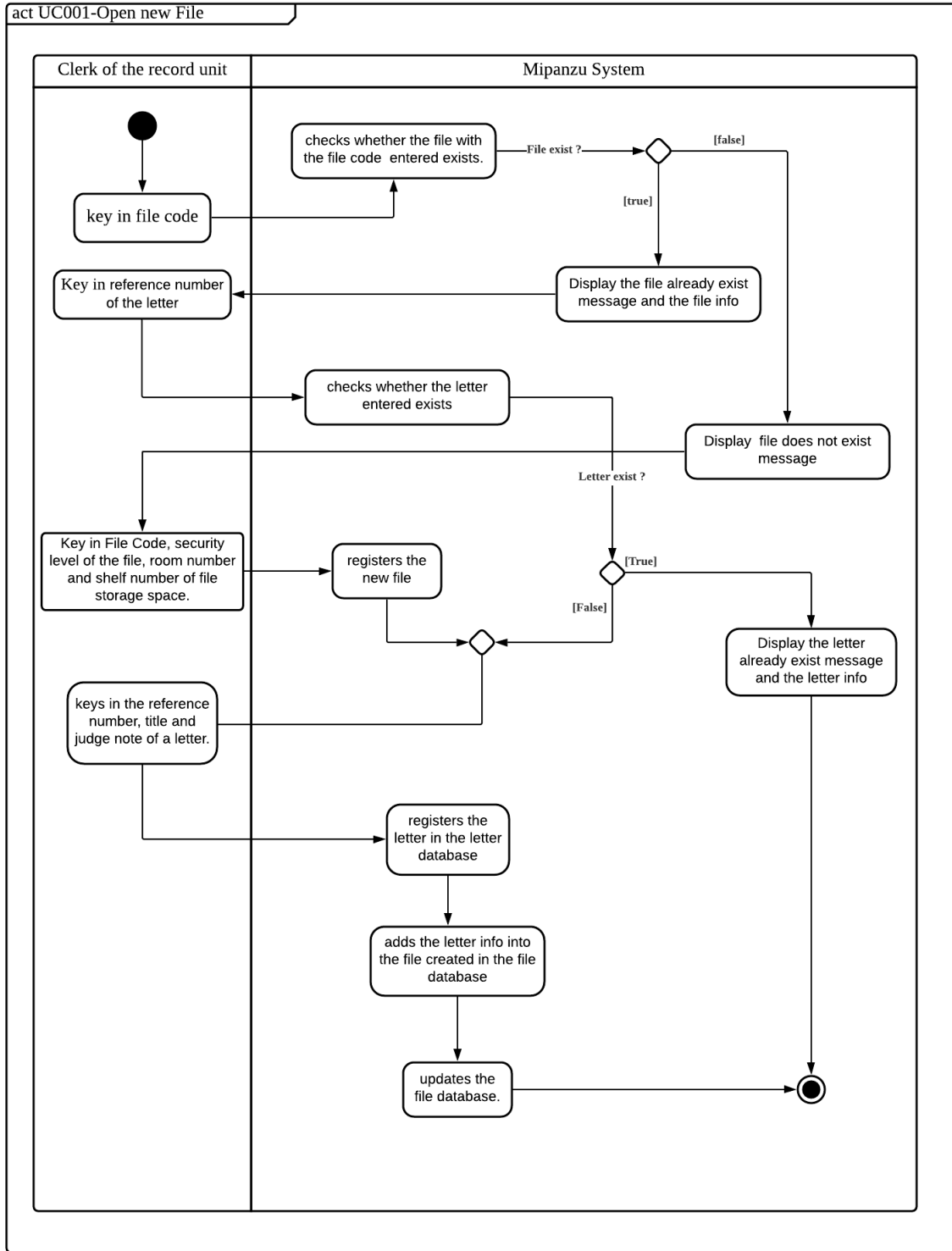


Figure 2.1: Activity Diagram for UC001 <Open new file>

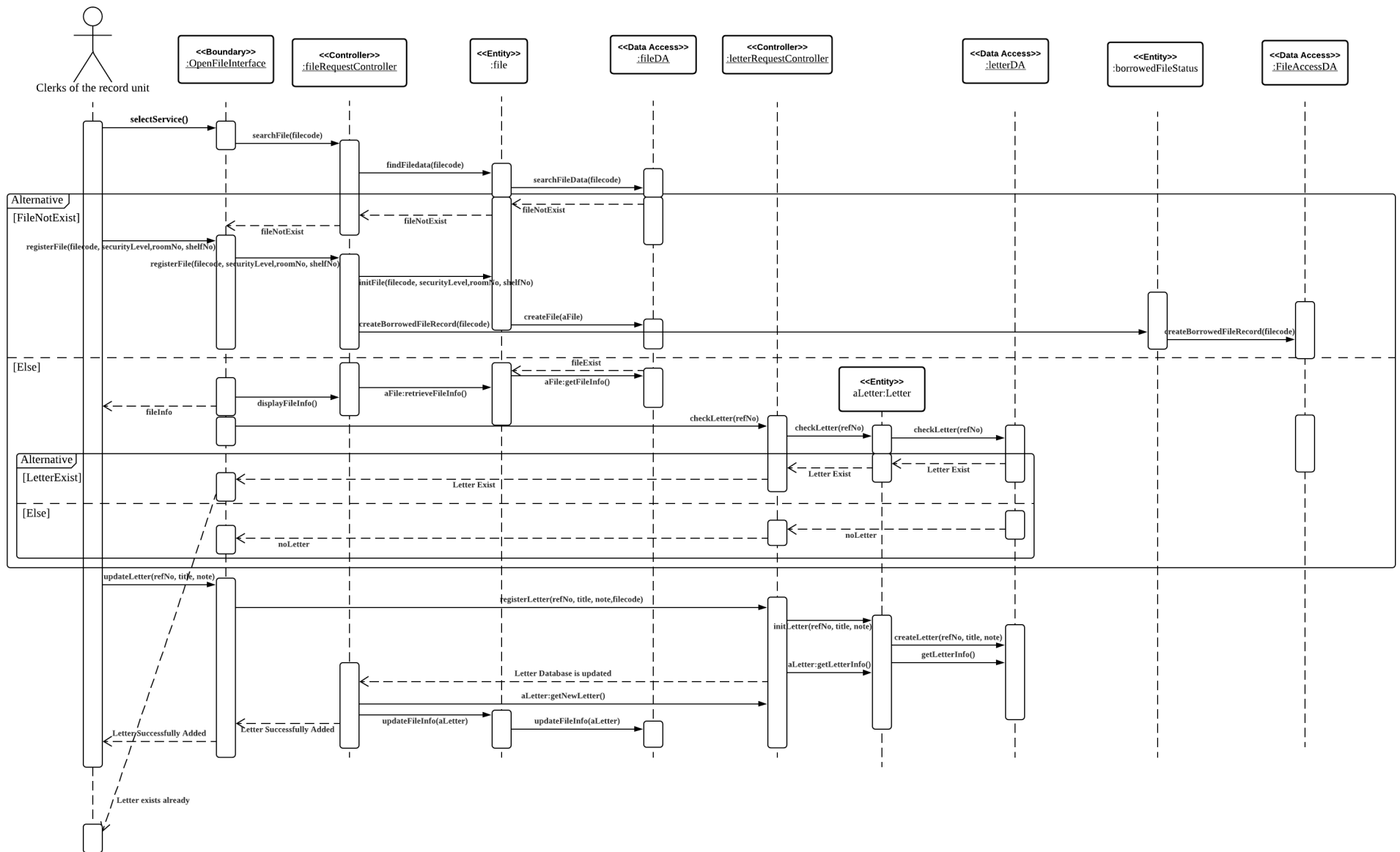


Figure 2.1.1: Sequence Diagram for UC001 <Open new file>

Table 2.2 Use Case Specification for UC002 <Close file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal and alternative flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC002		
Use Case Name	Close file		
Created by:	Husny	Last Updated by:	Husny
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit to close file in Mipanzu Online File Management System.		
Actor(s)	Clerks of the record unit		
Pre-condition(s)	A valid clerk of the record unit is logged on to the system.		
Normal Flow(s)- NF	<ol style="list-style-type: none"> 1. The clerk of record unit keys in the file code of the file. 2. The system automatically checks whether the file with the file code entered exists. 3. If the file does not exist, Exception E1 is performed. 4. The system displays file exists message. 5. The clerk of the record unit clicks on the close file button. 6. The system updates the file status to “Closed” in the file database. 7. The system automatically registers a new file with the same file info as the original closed file in the file database. 8. The system sets the subsequent file code to the new file. 9. The system displays the new file code. 10. The clerk of the record unit clicks on delete file to delete the original file info. [A1] 11. The system deletes the original file info from the file database. 		
Post-condition(s)	File is successfully closed by the clerk of the record unit		
Alternative Flow(s) - AF	[A1: The clerk of the record unit doesn’t click on delete file button] <ol style="list-style-type: none"> 1. The use case ends. 		
Post-condition(s)	File is successfully closed by the clerk of the record unit		

Exception Flow(s) - EF	E1: File does not exist 1. The system displays an error message. 2. The use case ends.		
Related Requirement (s):	ID	Requirement	Priority
	FR UC002-01	The system shall register a new file with the same file info as the original closed file in the file database.	Basic

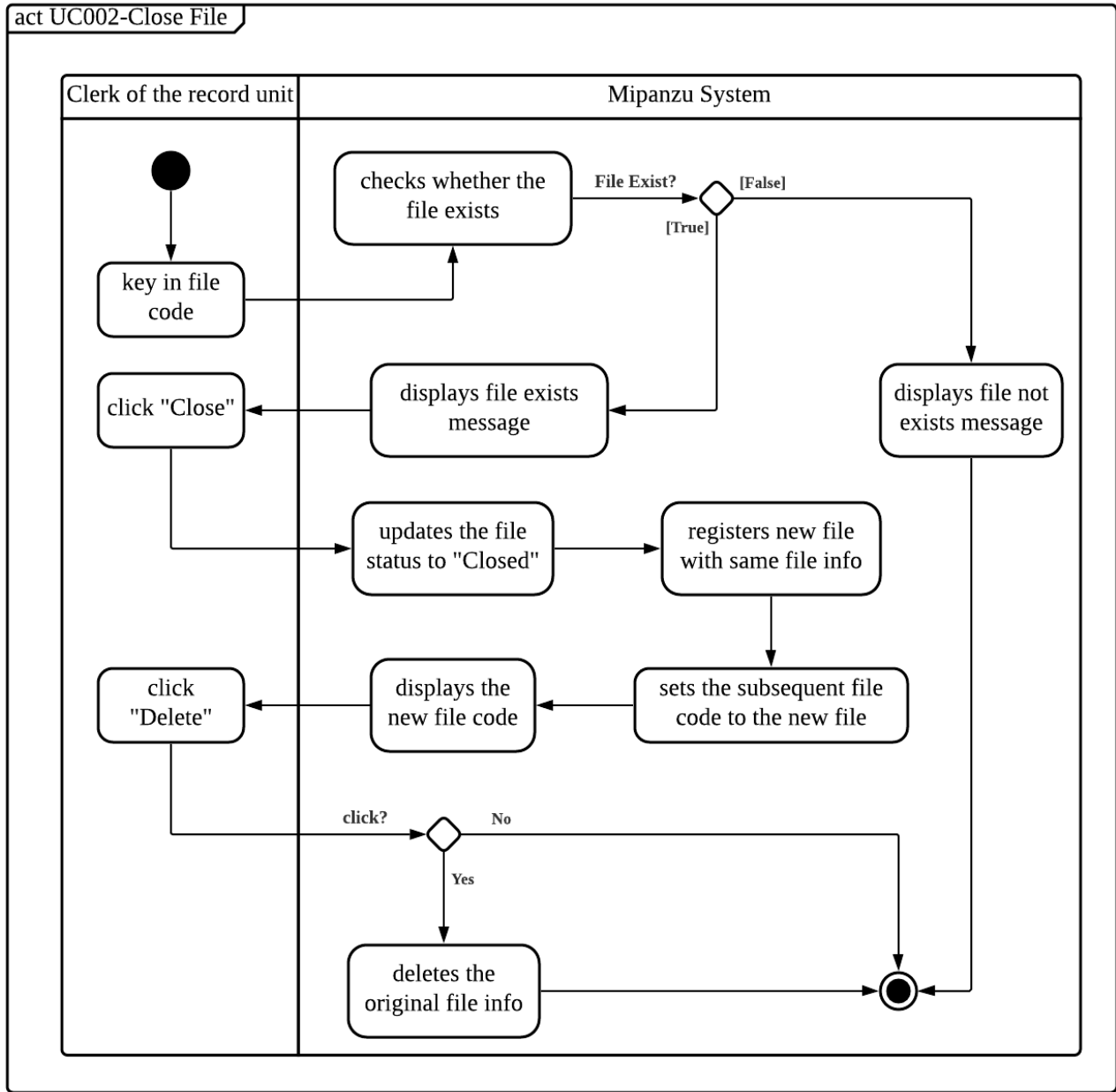


Figure 2.2: Activity Diagram for UC002 <Close file>

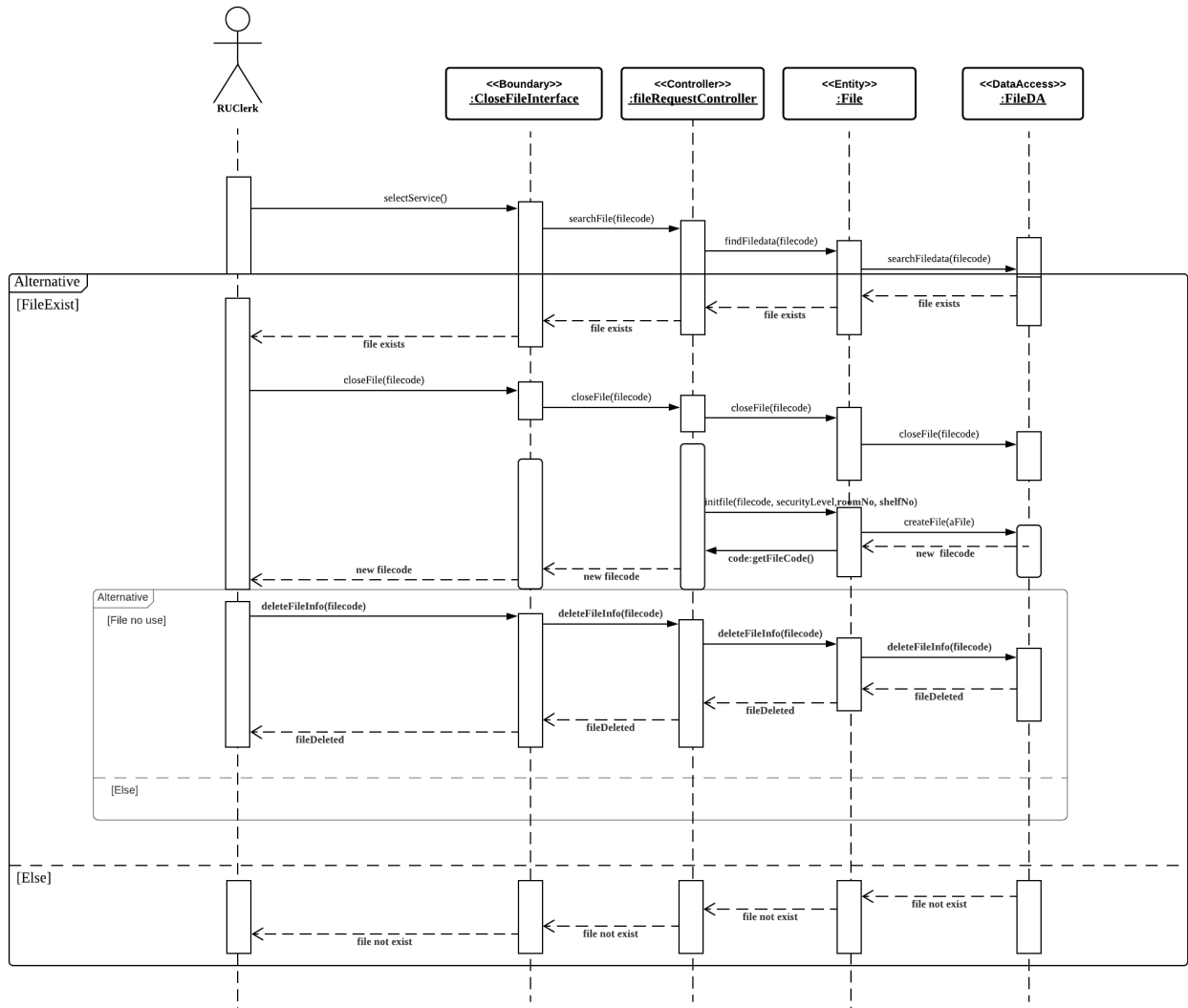


Figure 2.2.1: Sequence Diagram for UC002 <Close file>

Table 2.3 Use Case Specification for UC003 <Track file location>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal and exception flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC003		
Use Case Name	Track file location		
Created by:	Luqman	Last Updated by:	Luqman
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit to track file location in Mipanzu Online File Management System.		
Actor(s)	Clerks of the record unit		
Pre-condition(s)	A valid clerk of the record unit is logged on to the system.		
Normal Flow(s)- NF	<ol style="list-style-type: none"> 1. The clerk of record unit keys in the file code of the file. 2. The system automatically checks whether the file with the file code entered exists. 3. If the file does not exist, Exception E1 is performed. 4. The system displays file exists message. 5. The clerk of the record unit clicks on the search file button. 6. The system displays the location of the file (room number and shelf number). 		
Exception Flow(s) - EF	<p>E1: File does not exist</p> <ol style="list-style-type: none"> 1. The system displays an error message. 2. The use case ends. 		
Post-condition(s)	The file location is successfully tracked.		
Related Requirement(s):	ID	Requirement	Priority
	CR UC003-01	The system shall track 100 files location at one time.	Basic
	CR UC003-02	The location of the file displayed in the system must be match with the exact location of the file in the room number and shelf number	Basic

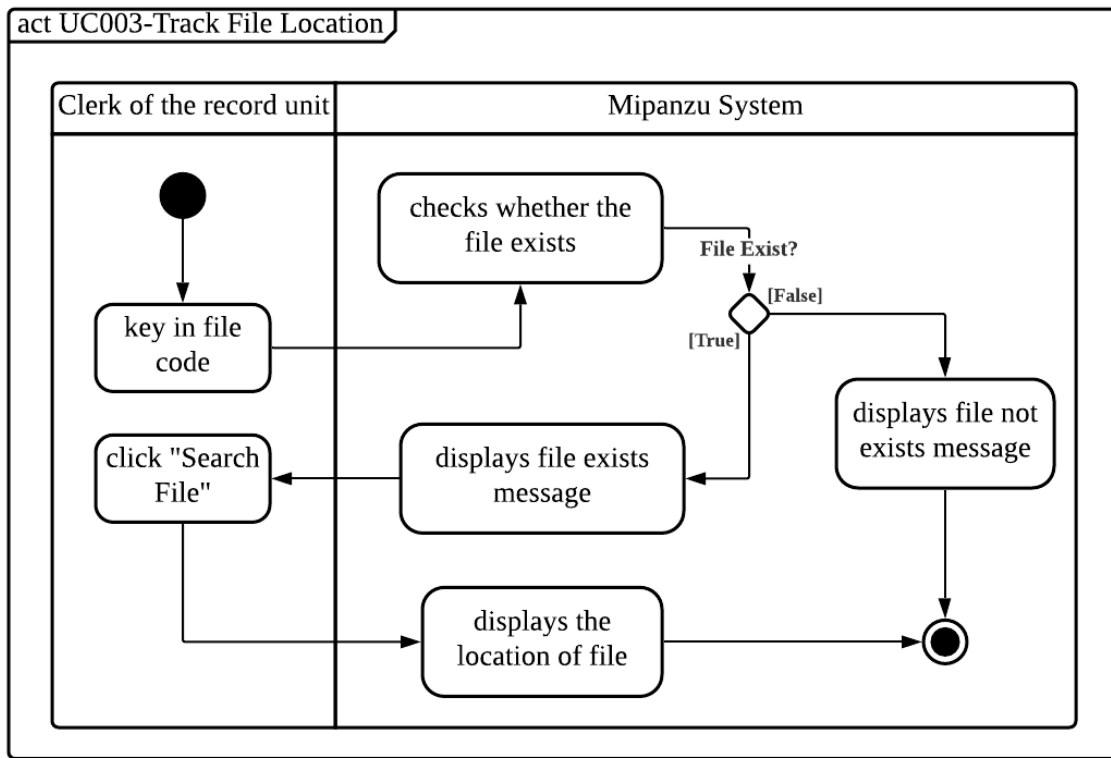


Figure 2.3: Activity Diagram for UC003 <Track file location>

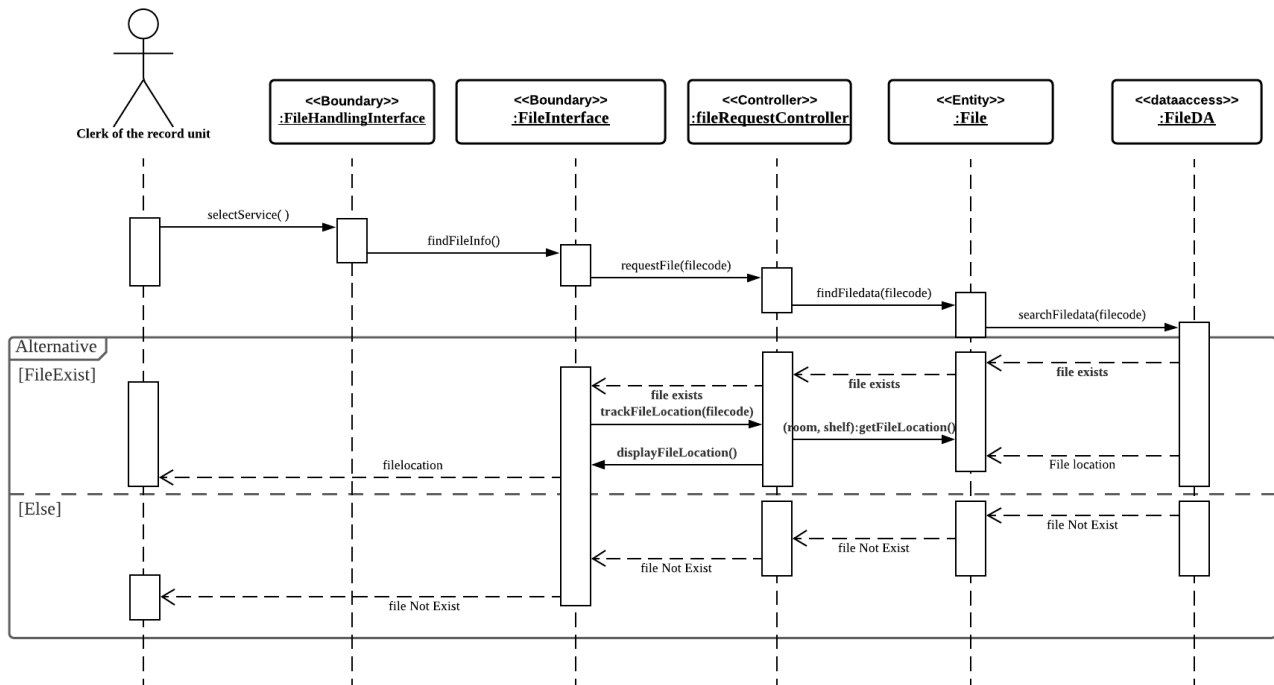


Figure 2.3.1: Sequence Diagram for UC003 <Track file location>

Table 2.4 Use Case Specification for UC004 <Report the loss of file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal and exception flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC004		
Use Case Name	Report the loss of file		
Created by:	Husny	Last Updated by:	Husny
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit and JKSNJ staff to report the loss of file in Mipanzu Online File Management System.		
Actor(s)	Clerk of record unit Syariah Assistant Officer JKSNJ staff		
Pre-condition(s)	A valid clerk of the record unit is logged on to the system.		
Normal Flow(s)-NF	<ol style="list-style-type: none"> 1. The JKSNJ staff keys in the file code. 2. The system automatically checks whether the borrow record with the staff's ID and the file code entered exists. 3. If the borrow record does not exist, Exception E1 is performed. 4. The system displays borrow record found message. 5. The JKSNJ staff clicks on the report file lost button. 6. The system updates the borrow status as lost in the file access database. 7. The system updates the file status as lost in the file database. 8. The system sends notification to Syariah Assistant Officer and the clerk of Record Unit about the file lost with the file code and the info of staff who lost the file. 		
Exception Flow(s) - EF	<p>E1: File does not exist</p> <ol style="list-style-type: none"> 1. The system displays an error message. 2. The use case ends. 		
Post-condition(s)	The loss of file is successfully reported.		
Related Requirement(s):	ID	Requirement	Priority
	FR UC004-01	The system shall send notification to Syariah Assistant Officer and the clerk of Record Unit about the file lost information.	Basic

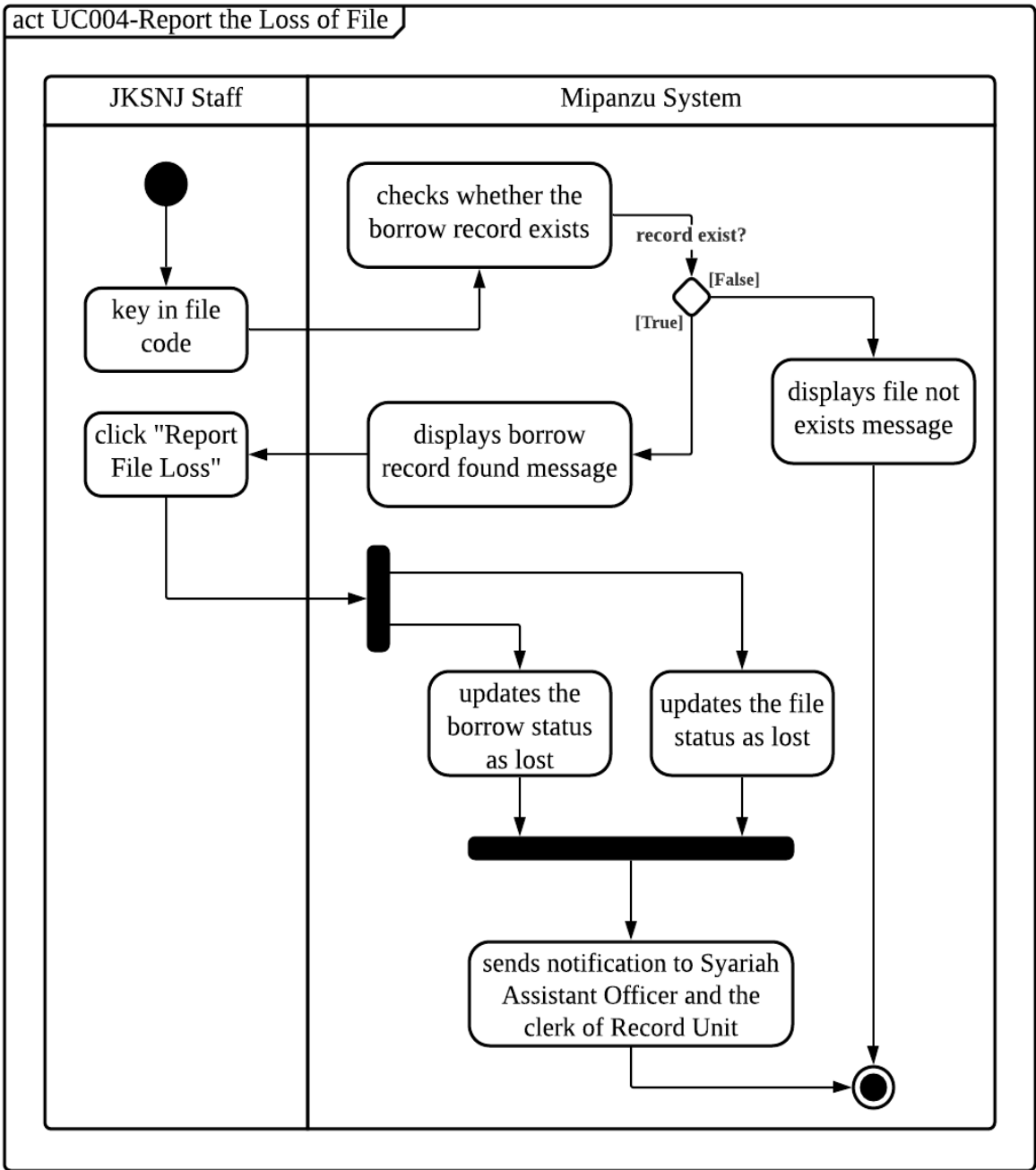


Figure 2.4: Activity Diagram for UC004 <Report the loss of file>

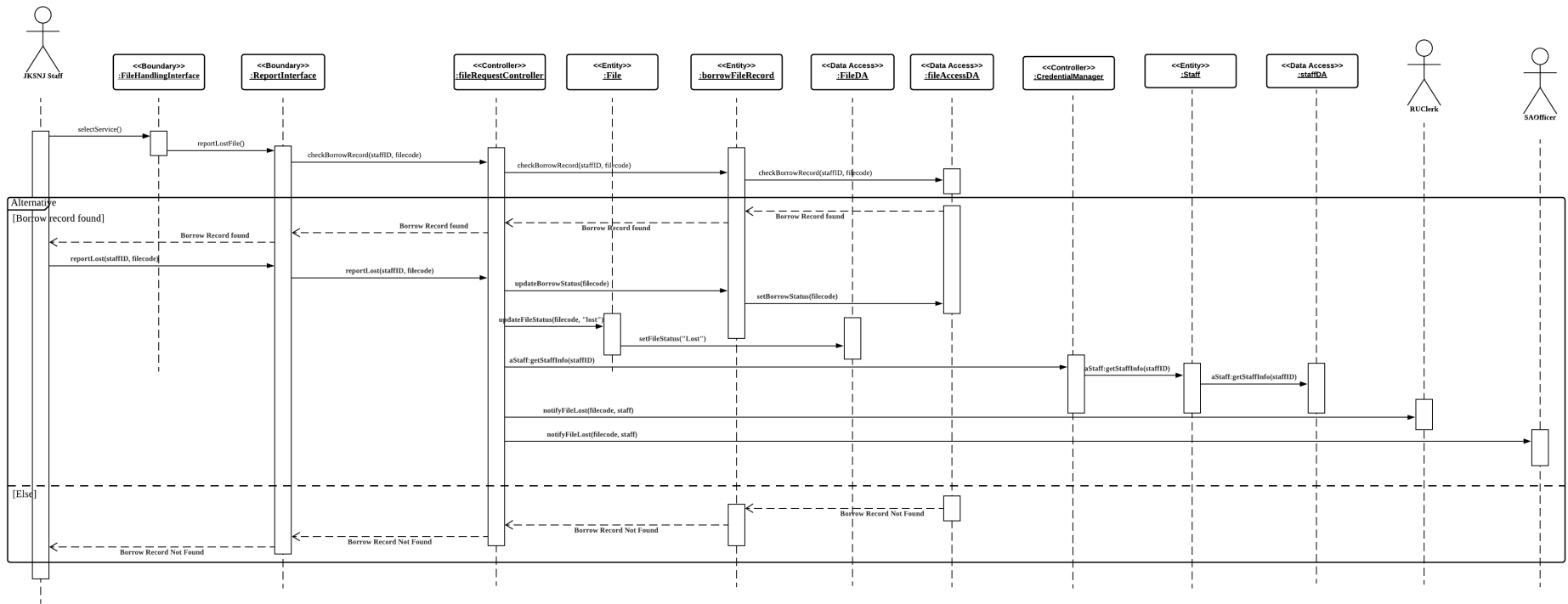


Figure 2.4.1: Sequence Diagram for UC004 <Report the loss of file>

Table 2.5 Use Case Specification for UC005 <Dispose of file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal, exception and alternative flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC005		
Use Case Name	Dispose of file		
Created by:	Tan Chiaw Torng	Last Updated by:	Tan Chiaw Torng
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit and the Syariah Assistant Officer to dispose file in Mipanzu Online File Management System.		
Actor(s)	Clerk of record unit Syariah Assistant Officer		
Pre-condition(s)	A valid clerk of the record unit and a Syariah Assistant Officer are logged on to the system. The clerk of the record unit and the Syariah Assistant Officer select “File Disposal” in the home page.		
Normal Flow(s)-NF	<ol style="list-style-type: none"> 1. The clerk of record unit clicks on “Outdated Files”. 2. System displays and loads the files that exceed the maximum storage time. 3. The clerk of the record unit enters the file codes of other damaged files to be disposed of and clicks on Search. 4. The system checks whether the file code exists in the system. 5. If the file code does not exist, A1 is performed. 6. The system displays a “File is found!” message. 7. The clerk of the record unit clicks the add button to add file to the list of files to be disposed of. 8. The clerk of the record unit clicks on Submit to send the file dispose list through the system to Syariah Assistant Officer. 9. System displays “Submitted Successfully!” and sends the file dispose list to Syariah Assistant Officer. 10. Syariah Assistant Officer views the file dispose list and updates the thickness of files from the file dispose list. 11. Syariah Assistant Officer clicks on “Download Disposal Forms”. 12. The system checks whether the thickness of all files is entered. 13. If the thickness of files is not fully filled in, A2 is performed. 14. System updates the file thickness in database. 		

	<p>15. System generates File Disposal Application Forms filled in with the details of each file in the file disposal list and are downloaded by Syariah Assistant Officer in zip file.</p> <p>16. Syariah Assistant Officer clicks on “Email” which links to the Outlook mailbox to apply for the file disposal with Arkib Negara Malaysia.</p> <p>17. Syariah Assistant Officer returns to the Mipanzu system.</p> <p>18. Syariah Assistant Officer clicks on Add to Disposal Record.</p> <p>19. System updates the disposed file record and the result page with the files from the file disposal list.</p> <p>20. System displays “Added Successfully!”.</p> <p>21. After receiving email of file disposal list with approval status and disposal method from Arkib Negara Malaysia, Syariah Assistant Officer clicks on “Result” to update the application status and disposal method.</p> <p>22. If the application of file disposal is rejected, Exception E1 is performed.</p> <p>23. Syariah Assistant Officer selects the file dispose status as “Approved”.</p> <p>24. Syariah Assistant Officer selects the file disposal method.</p> <p>25. Syariah Assistant Officer clicks on “update”.</p> <p>26. System displays “Updated Successfully”.</p> <p>27. System updates the file disposal status and disposal method.</p> <p>28. The clerk of the record unit clicks on “Approved list”.</p> <p>29. The clerk of the record unit views the list of files with approved file disposed status.</p> <p>30. The clerk of the record unit selects the file disposal date.</p> <p>31. The clerk of record unit selects the file dispose status as “Disposed”.</p> <p>32. The clerk of record unit clicks on “Update”.</p> <p>33. System displays “Updated Successfully”.</p> <p>34. System updates the file disposal date and status as “Disposed”.</p>								
<p>Alternative Flow(s):</p>	<p>A1: The file code entered is not found</p> <ol style="list-style-type: none"> 1. The system displays an error message 2. Step 3 resumes. <p>A2: The thickness of all files is not fully filled in</p> <ol style="list-style-type: none"> 1. The system displays an error message. 2. Step 10 resumes. 								
<p>Exception Flow(s) - EF</p>	<p>E1: The application of file disposal is rejected</p> <ol style="list-style-type: none"> 1. Syariah Assistant Officer updates the file dispose status as “Rejected”. 2. System updates the file dispose status. 3. The use case ends. 								
<p>Post-condition(s)</p>	<p>File is successfully disposed.</p>								
<p>Related Requirement(s):</p>	<table border="1"> <thead> <tr> <th data-bbox="428 1732 626 1791">ID</th> <th data-bbox="626 1732 1203 1791">Requirement</th> <th data-bbox="1203 1732 1453 1791">Priority</th> </tr> </thead> <tbody> <tr> <td data-bbox="428 1791 626 1953">FR UC005-01</td> <td data-bbox="626 1791 1203 1953">The system shall provide the clerk of the record unit with the ability to view outdated files which are sorted according to Record Disposal Schedule.</td> <td data-bbox="1203 1791 1453 1953">Basic</td> </tr> </tbody> </table>	ID	Requirement	Priority	FR UC005-01	The system shall provide the clerk of the record unit with the ability to view outdated files which are sorted according to Record Disposal Schedule.	Basic		
ID	Requirement	Priority							
FR UC005-01	The system shall provide the clerk of the record unit with the ability to view outdated files which are sorted according to Record Disposal Schedule.	Basic							

	FR UC005-02	The system shall provide Syariah Assistant Officer with the ability to download the file disposal application forms for the files in the latest file disposal list with the details of files filled in automatically by the system based on the format of Archive Form 5/08 and Archive Form 2/08ada from Arkib Negara Malaysia..	Basic
--	-------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------

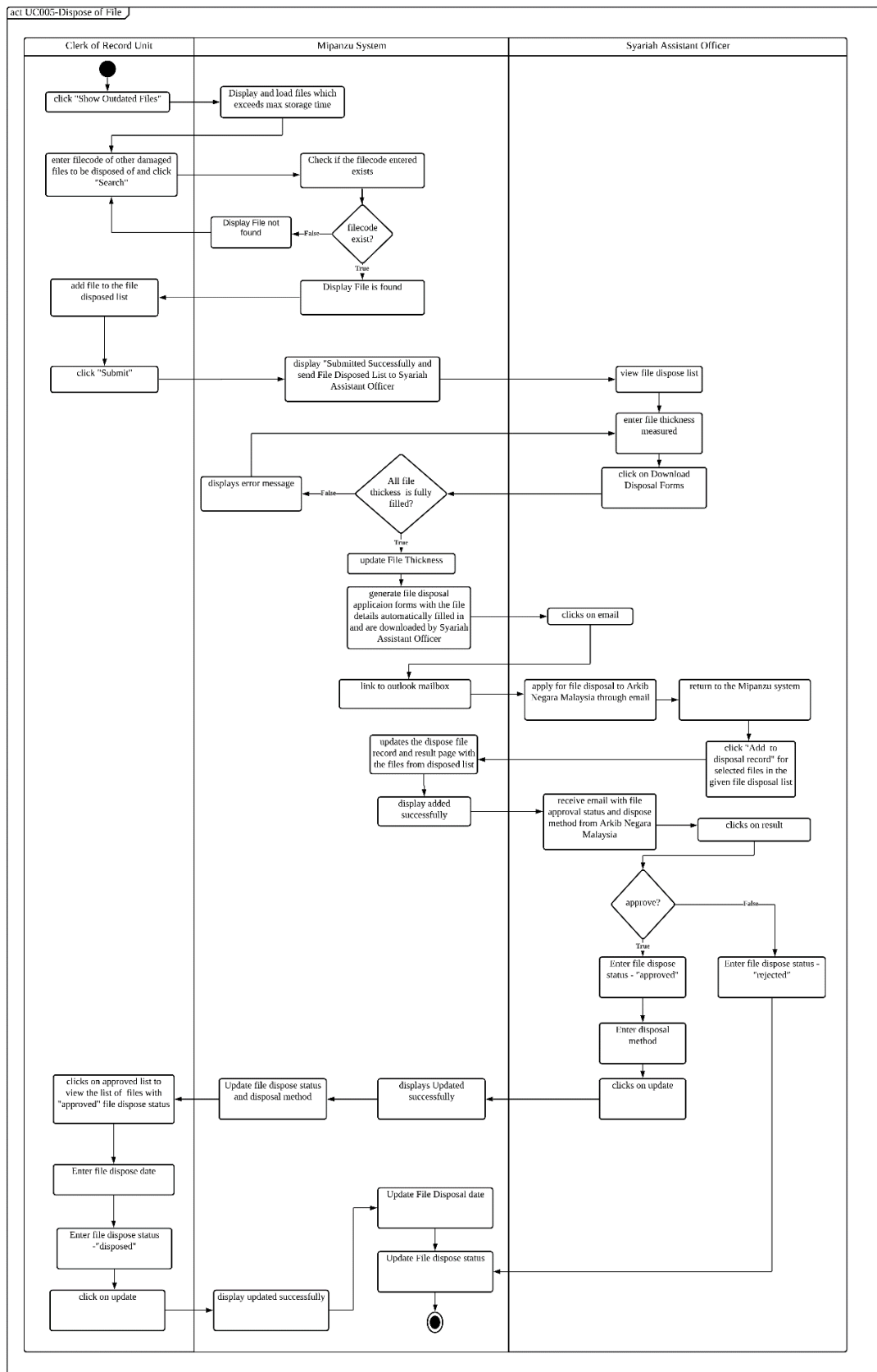


Figure 2.5: Activity Diagram for UC005 <Dispose of file>

Table 2.6 Use Case Specification for UC006 <Borrow file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Add information according to requirements management	
Version:	1.1		
Use Case ID	UC006		
Use Case Name	Borrow file		
Created by:	See Wen Xiang	Last Updated by:	See Wen Xiang
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by JKSNJ staff to borrow file from Clerk of the record unit in Mipanzu Online File Management System.		
Actor(s)	Clerk of record unit JKSNJ staff		
Pre-condition(s)	A valid JKSNJ staff is logged on to the system.		
Normal Flow(s)-NF	<ol style="list-style-type: none"> 1. The JKSNJ staff keys in the file code. 2. The system automatically checks whether the file with the file code entered exists. 3. If the file does not exist, Exception E1 is performed. 4. The system automatically checks whether the file with the file code entered is still available (not borrowed). 5. If the file is not available, Exception E2 is performed. 6. The system automatically checks whether the staff has the privilege to borrow the file. 7. If the staff does not have the privilege to borrow the file, Exception E3 is performed. 8. The system creates a borrow ID. 9. The system updates the file access database. 10. The system shows the borrow ID to the staff on the borrow file interface. 11. The clerk of the record unit keys in the borrow ID. 12. The system displays the borrow record. 		
Exception Flow(s) - EF	<p>E1: File does not exist</p> <ol style="list-style-type: none"> 1. The system displays File not found message. 2. The use case ends. <p>E2: File is not available</p> <ol style="list-style-type: none"> 1. The system displays the last borrower's name and staff ID. 2. The system displays File is not available message. 3. The use case ends. <p>E3: The staff does not have the privilege to borrow the file</p> <ol style="list-style-type: none"> 1. The system displays message No right to borrow the file. 		

	2. The use case ends.		
Post-condition(s)	File is successfully borrowed by the JKSNJ staff.		
Related Requirement(s):	ID	Requirement	Priority
	CR UC006-01	The system shall allow the JKSNJ staff to borrow a maximum of 5 files.	Basic

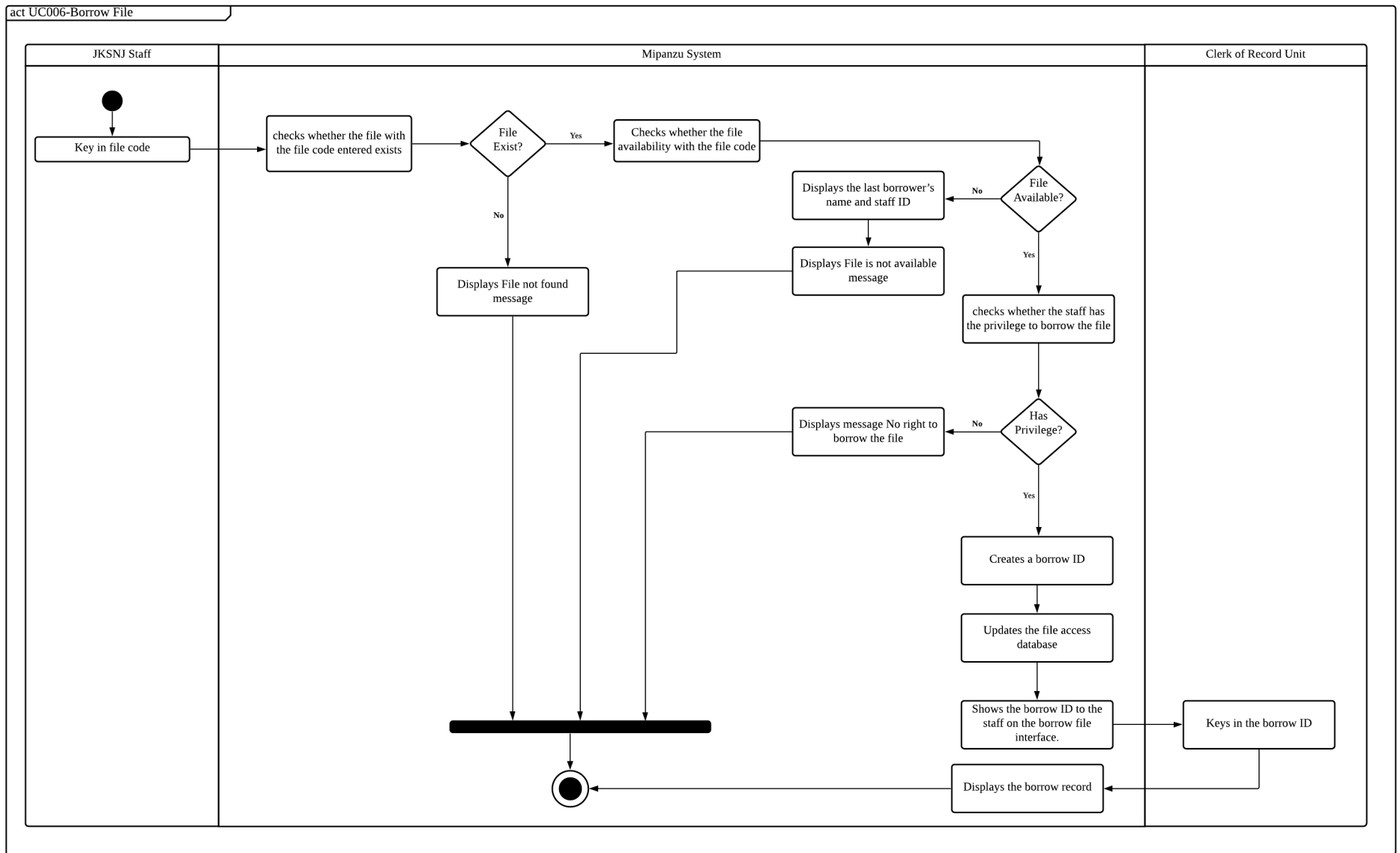


Figure 2.6: Activity Diagram for UC006 <Borrow file>

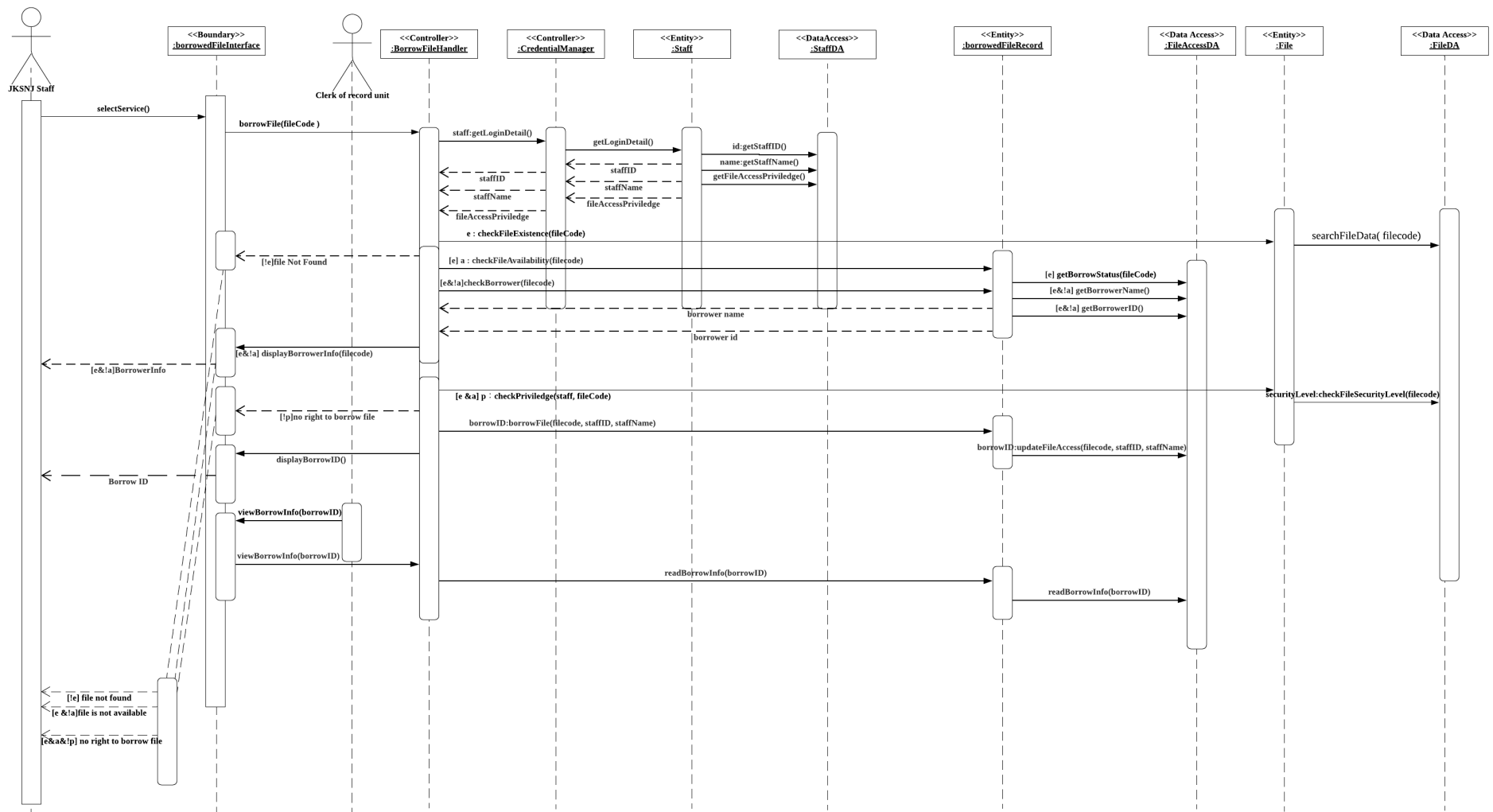


Figure 2.6.1: Sequence Diagram for UC006 <Borrow file>

Table 2.7 Use Case Specification for UC007 <View file access record>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal and exception flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC007		
Use Case Name	View file access record		
Created by:	Tan Chiaw Torng	Last Updated by:	Tan Chiaw Torng
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit t view file access record in Mipanzu Online File Management System.		
Actor(s)	Clerk of record unit		
Pre-condition(s)	A valid Clerk of record unit is logged on to the system.		
Normal Flow(s)- NF	<ol style="list-style-type: none"> 1. The clerk of record unit keys in the file code of the file. 2. The system automatically checks whether the file with the file code entered exists. 3. If the file does not exist, Exception E1 is performed. 4. The system displays the file access record. 		
Exception Flow(s) - EF	<p>E1: File does not exist</p> <p>The system displays an error message. The use case ends.</p>		
Post-condition(s)	File access record is successfully viewed by Clerk of the record unit		
Related Requirement(s):	ID	Requirement	Priority
	FR UC007-01	The system shall display the file access record including the name of the last borrower and datetime of borrow and return when the clerk of record unit entered a valid file code by keyboard of the terminal.	Basic

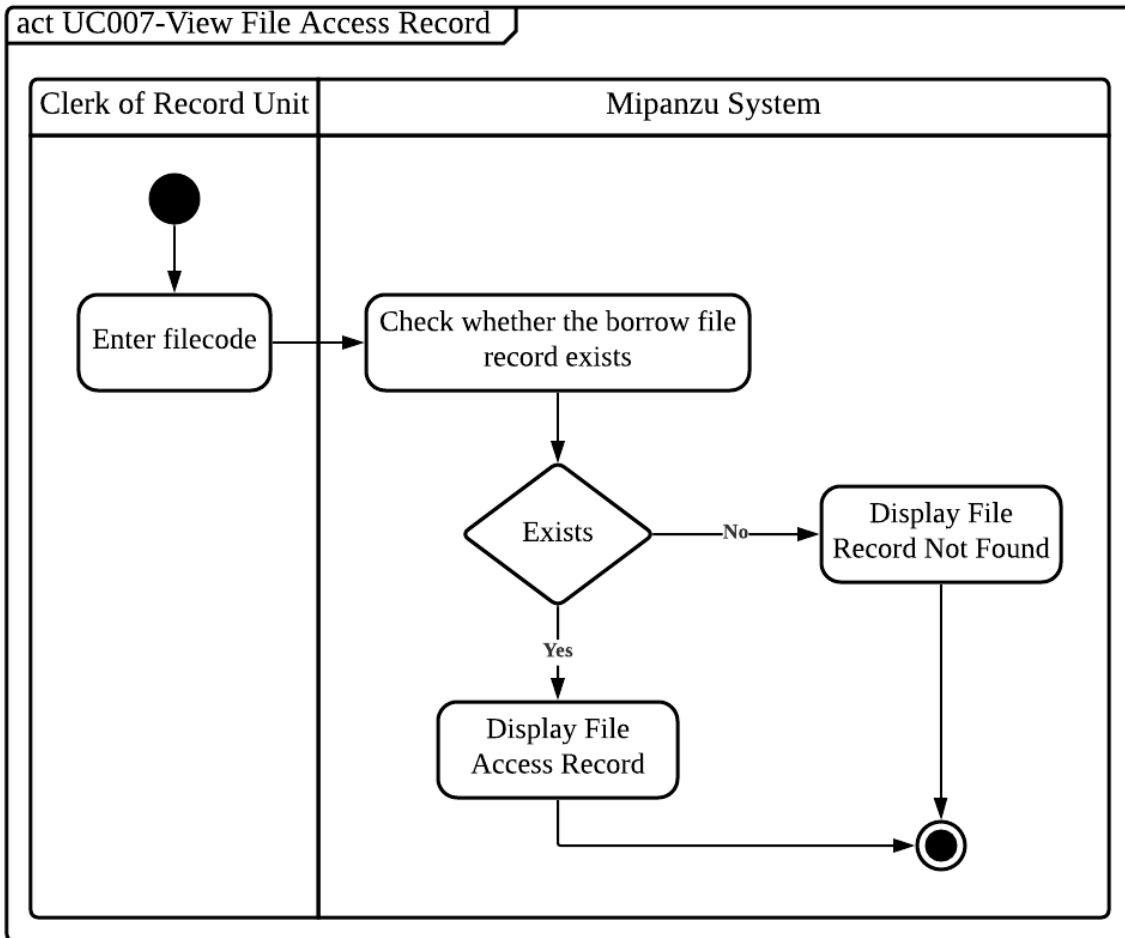


Figure 2.7: Activity Diagram for UC007 <View file access record>

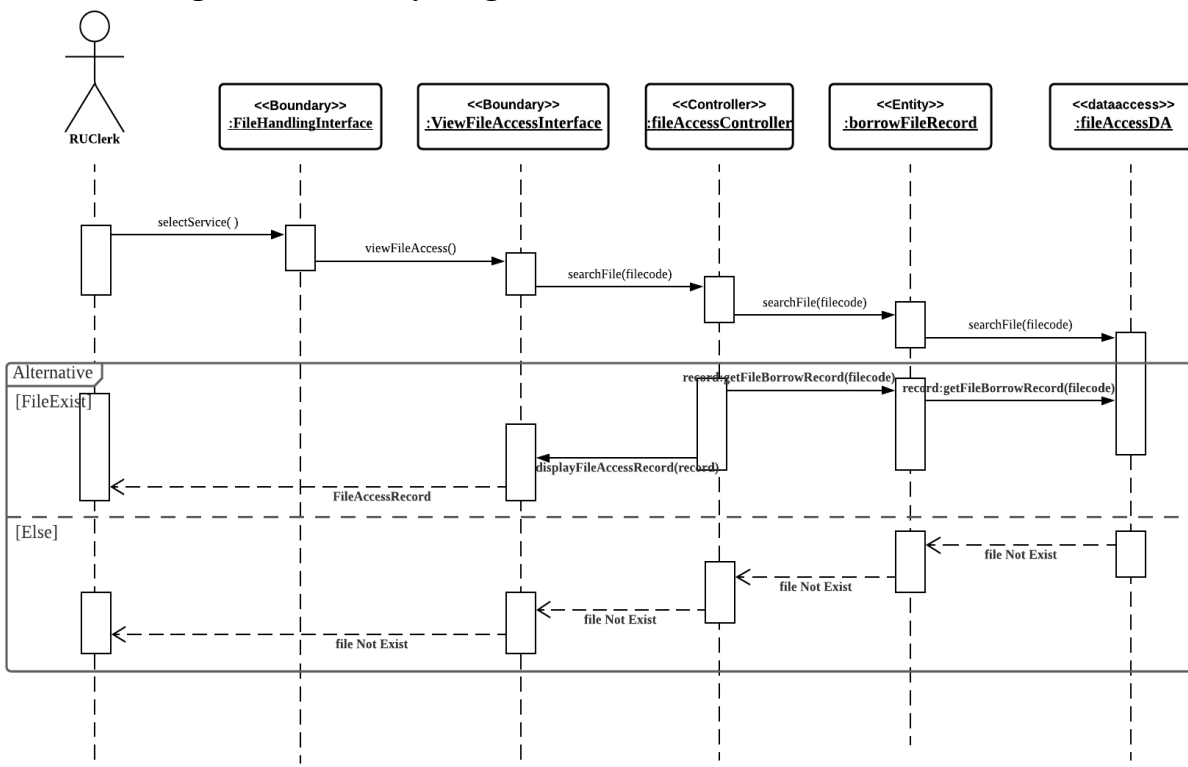


Figure 2.7.1: Sequence Diagram for UC007 <View file access record>

Table 2.8 Use Case Specification for UC008 <Detect the late return of file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Add information according to requirements management	
Version:	1.1		
Use Case ID	UC008		
Use Case Name	Detect the late return of file		
Created by:	Luqman	Last Updated by:	Luqman
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by Clerk of the record unit to detect the late return of file by JKSNJ staff in Mipanzu Online File Management System.		
Actor(s)	JKSNJ staff Clerk of the record unit		
Pre-condition(s)	A valid Clerk of record unit is logged on to the system.		
Normal Flow(s)- NF	<ol style="list-style-type: none"> 1. The clerk of the record unit clicks on the View File Late Return button. 2. The system displays the list of files that exceeds the return deadline with their file access record. 3. The system notifies the last borrower of the files in the late return file list. 		
Post-condition(s)	<ol style="list-style-type: none"> 1. The list of files that exceed the deadline of return is successfully viewed. 2. The return file notification is successfully sent to the last borrowers. 		
Related Requirement (s):	ID	Requirement	Priority
	CR UC008-01	The system shall notify the user about late return of file after 1 hours the user exceeds the return deadline.	Basic

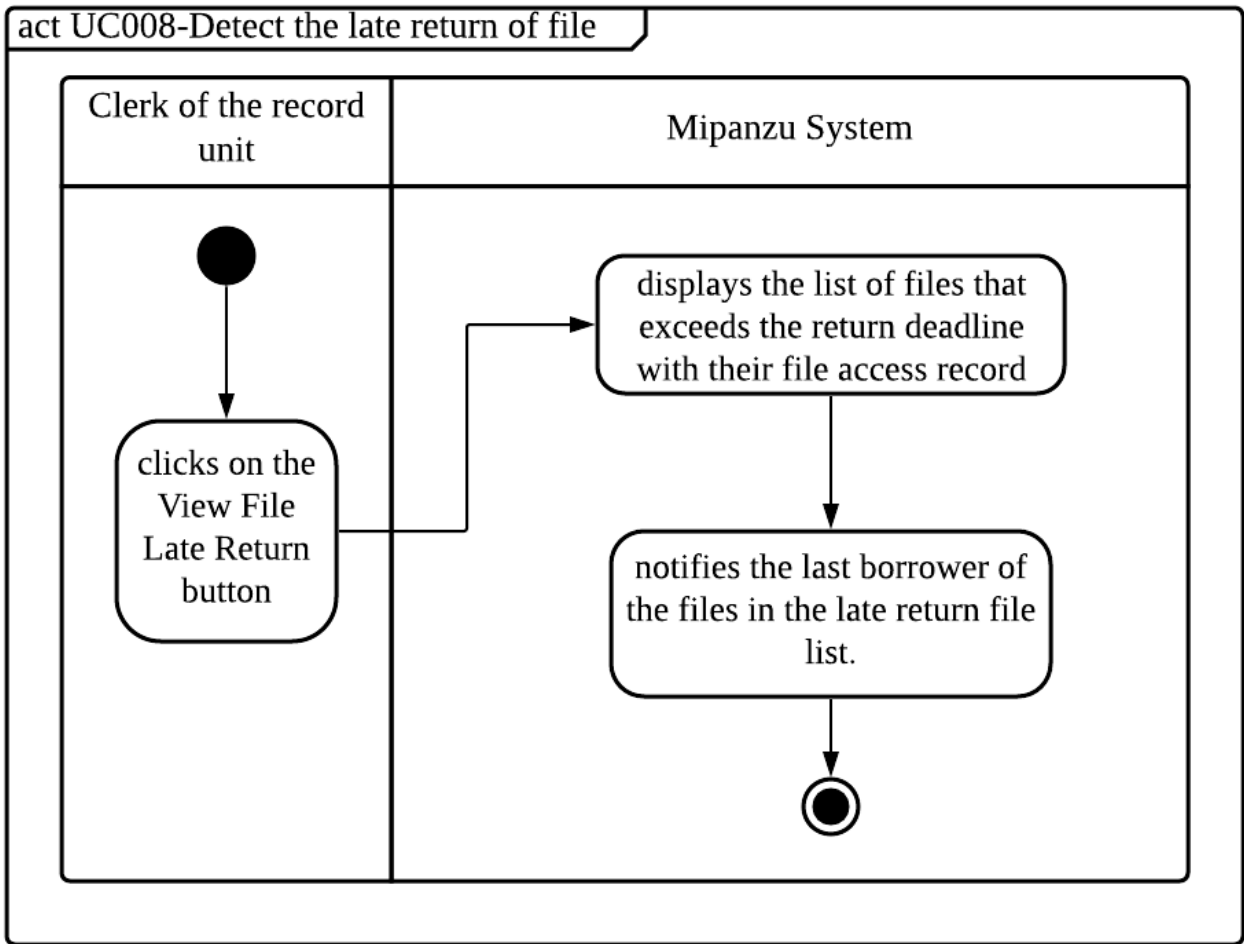


Figure 2.8: Activity Diagram for UC008 <Detect the late return of file>

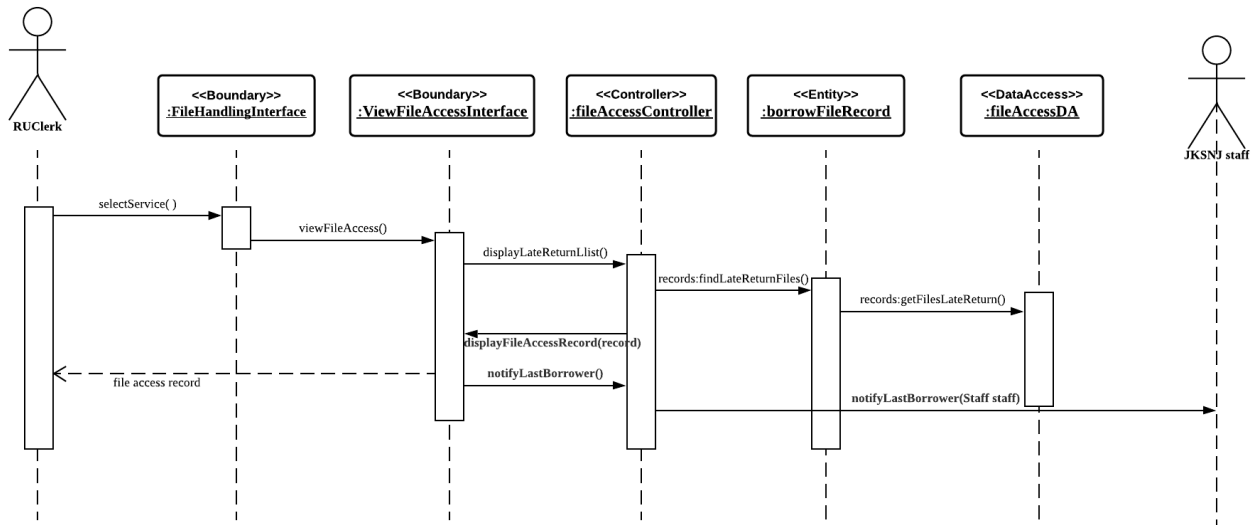


Figure 2.8.1: Sequence Diagram for UC008 <Detect the late return of file>

Table 2.9 Use Case Specification for UC009 <Return file>

History Log:	<ID>	<Changes>	
	1.0	Create initial use case	
	1.1	Fix the normal and exception flow	
	1.2	Add information according to requirements management	
Version:	1.2		
Use Case ID	UC009		
Use Case Name	Return file		
Created by:	See Wen Xiang	Last Updated by:	See Wen Xiang
Data Created:	14/4/2021	Last Revision Date:	13/6/2021
Description	This use case is used by JKSNJ staff to return file to Clerk of the record unit in Mipanзу Online File Management System.		
Actor(s)	JKSNJ staff Clerk of the record unit		
Pre-condition(s)	<ol style="list-style-type: none"> 1. A JKSNJ staff is taking the file that they borrowed to the counter of JKSNJ. 2. A valid clerk of the record unit is logged on to the system. 		
Normal Flow(s)- NF	<ol style="list-style-type: none"> 1. The clerk of the record unit keys in the borrow ID. 2. The system automatically checks whether the borrow record with the borrow ID entered exists. 3. If the borrow record does not exist, Exception E1 is performed. 4. The system displays the file borrow info. 5. The clerk of the record unit clicks on the Return File button. 6. The system updates the borrow status in the file access database. 		
Exception Flow(s) - EF	<p>E1: Borrow Record does not exist</p> <ol style="list-style-type: none"> 1. The system displays an error message. 2. The use case ends. 		
Post-condition(s)	File is successfully returned from the JKSNJ staff		
Related Requirement(s):	ID	Requirement	Priority
	FR UC009-01	The system shall display an error message which is “Borrow Record does not exist”.	Basic

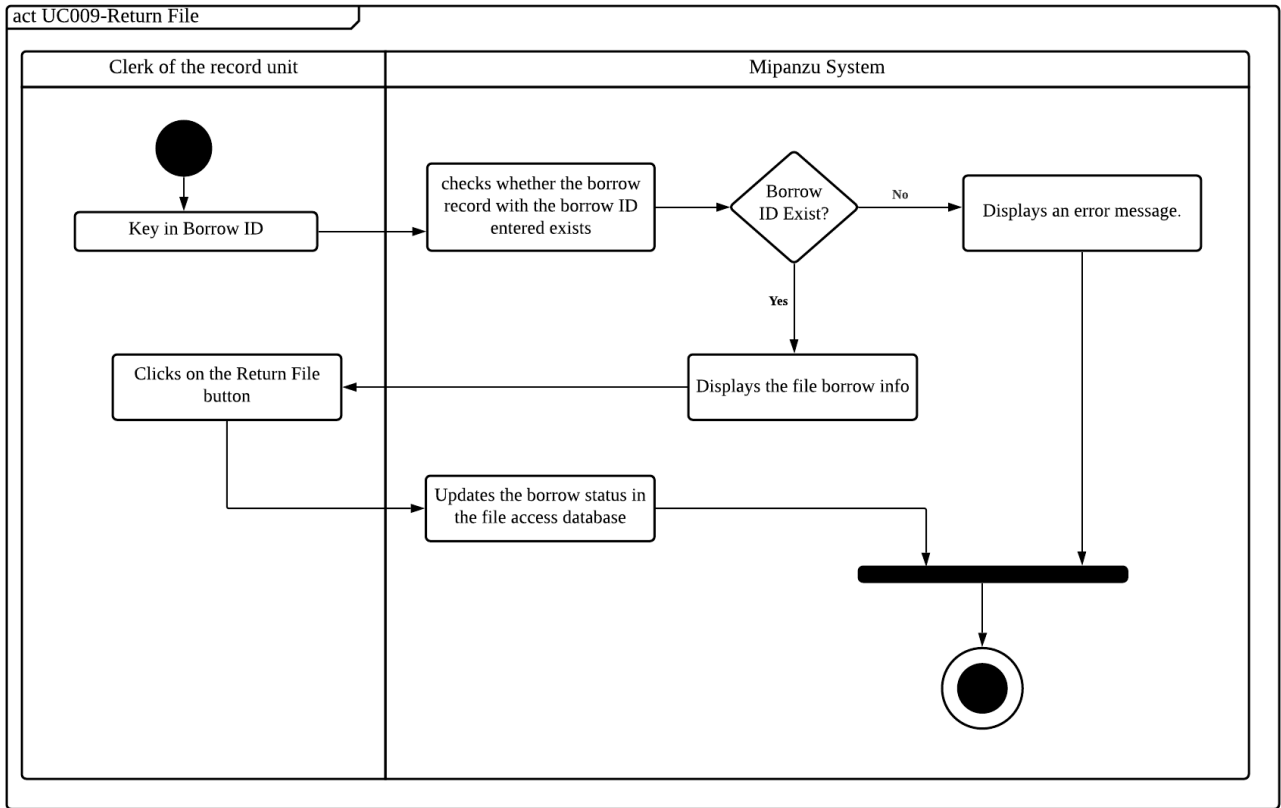


Figure 2.9: Activity Diagram for UC009 <Return file>

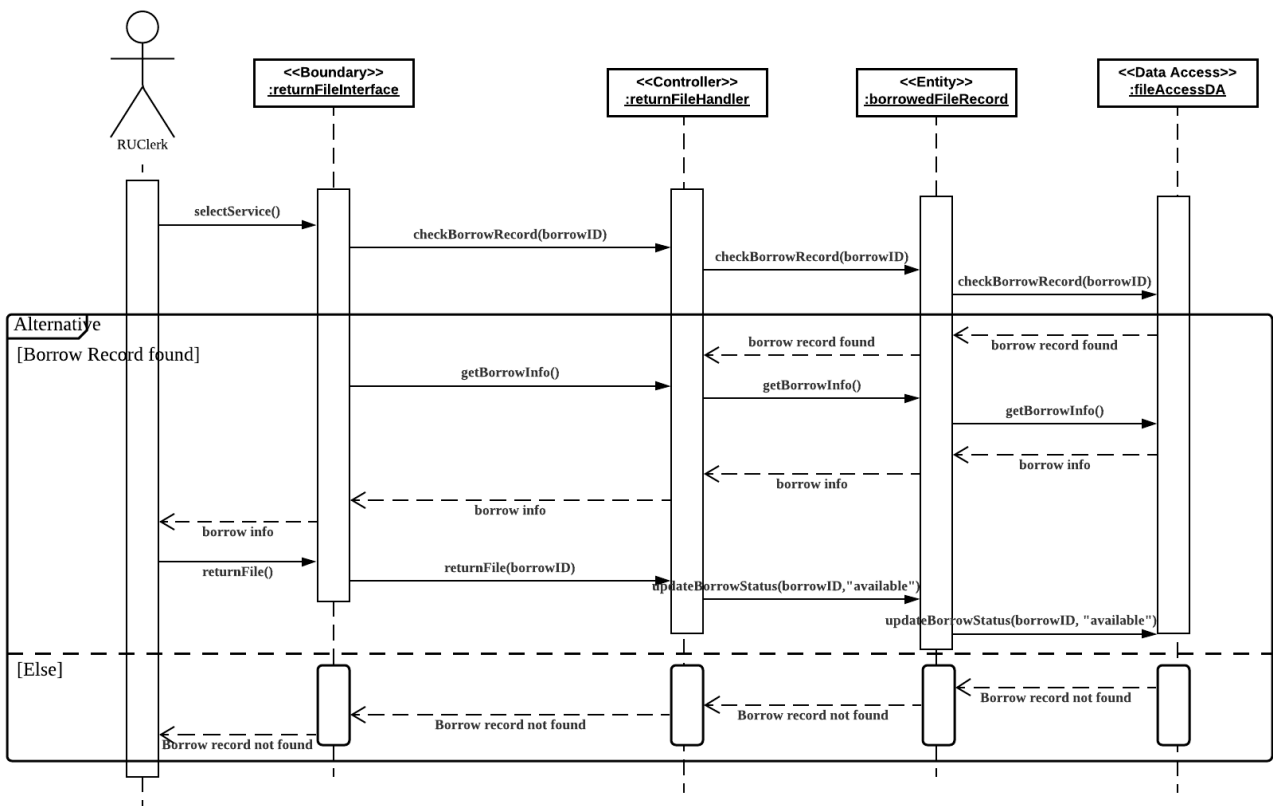


Figure 2.9.1: Sequence Diagram for UC009 <Return file>

3. Data Perspective: UML Domain Class Diagram (DCD)

For UML Domain Class Diagram, there are several entities that need to be created in the system which are Staff, File, borrowedFileRecord, file DisposedRecord and Letter. They are also two entity that will inherit the staff attributes which are the SAOfficer and RUClerk which is a more specific position that have their own unique attributes. For relationship, one staff may borrow and return one or many Files. At the same time, one Staff Administrator Officer (SAOfficer) may examine and monitor one or many borrowedFileRecord and fileDisposedRecord. Other than that, one Clerks of the record unit also can open and close one and many Files. File has a composition relationship with Letter which means that the Letter cannot exist if there are no File. One or many Files may contain one Letter. File also has an aggregation relationship with borrowedFileRecord therefore one or many Files will have one or many borrowedFileRecord. Meanwhile, fileDisposedRecord has an aggregation relationship with file so one or many fileDisposedRecord will has and list out one or many Files. The aggregation relationship means that the borrowedFileRecord belongs to File while File belongs to fileDisposedRecord.

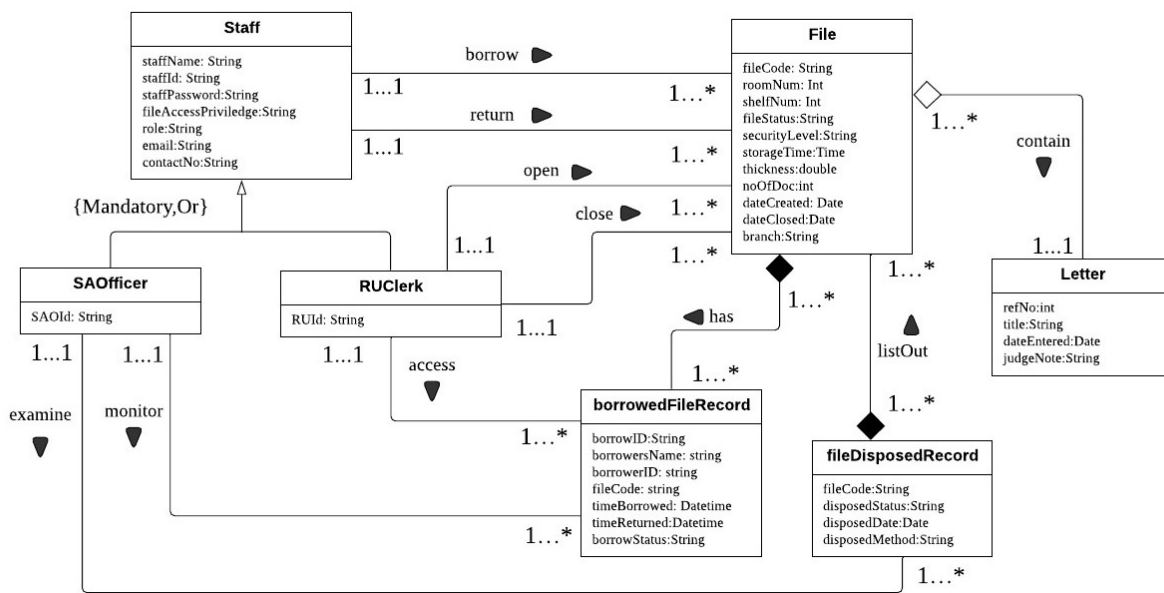


Figure 3.0: Domain Class Diagram for Mipanzu Online File Management System

4. Behavioral Perspective: UML State-Transition Diagram (STD)

For UML State-Transition Diagram, there are several entities class that is created for state-transition diagram which are File, borrowedFileRecord, file DisposedRecord. Not only that, there are two controller class that have been created as well which is disposeFileHandler and borrowFileHandler. These classes are chosen because each of these entities have their own states and will change between states when they are triggered.

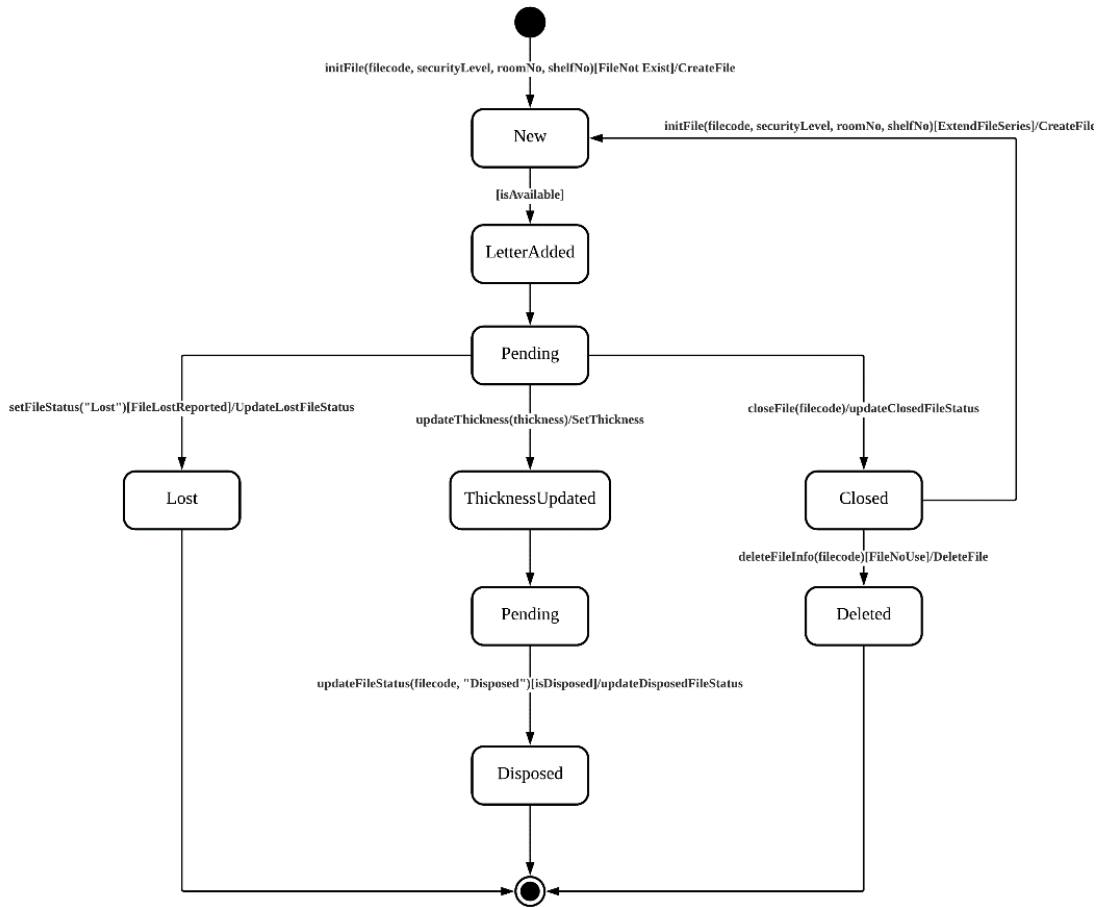


Figure 4.1: State-Transition Diagram for <File Class>

Figure 4.1 shows the state transition diagram for file class. A new file will be created first in the system. Then, the new state will change into letter added state as every file need to include a letter. Then, the file class will change the state into pending as they will be three different state that can happen to the file class based on the arguments and condition. The first condition is when the file status is lost. This will make the file class to change into lost and the state transition will end there. The second condition is when the file need to be disposed. First, the file state will change into thickness updated as we need to update the thickness of the file. Then, the file will change it state into pending as file class need to make sure that the file fulfills the condition to be disposed. Lastly, the file class will change its state into disposed and the state transition end there. The third condition made the file class to change its current state into closed. Then, the state will change either to a new file or deleted. The state change into a new file as the file that is closed will be used to attach to a new file created. For deleted, the file will be in the deleted state and the state transition end here.

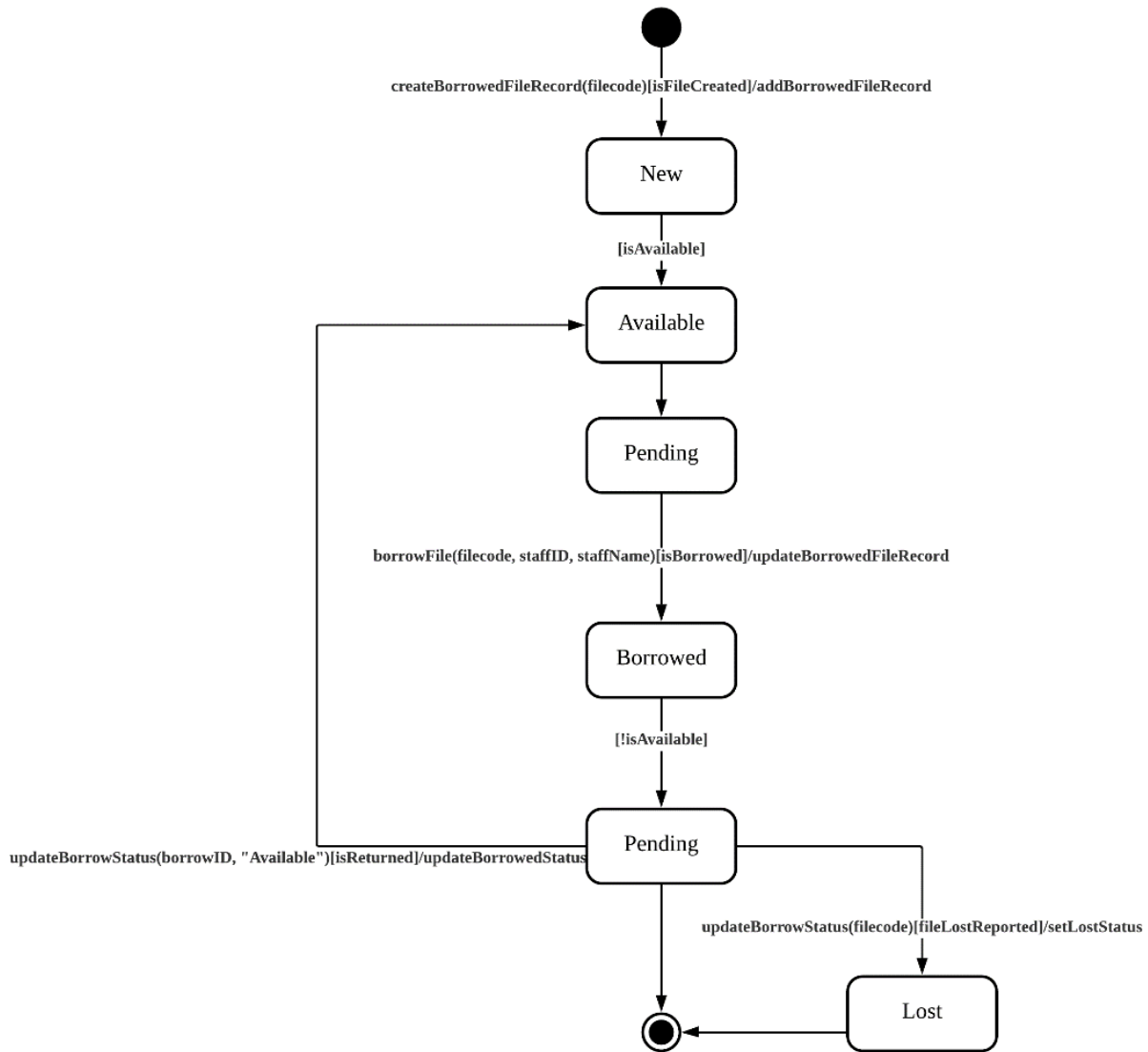


Figure 4.2: State-Transition Diagram for <BorrowedFileRecord Class>

Figure 4.2 shows the state-transition diagram for BorrowedFileRecord class. A new record will be created first in the system. Then, the new state will change into available state. It will be in pending state before it will be borrowed. After that, the state will change to borrowed state once it is borrowed. The state will be pending again when it is borrowed before returning it. If the borrow file is lost, the state will change to lost state and the state-transition diagram ends here.

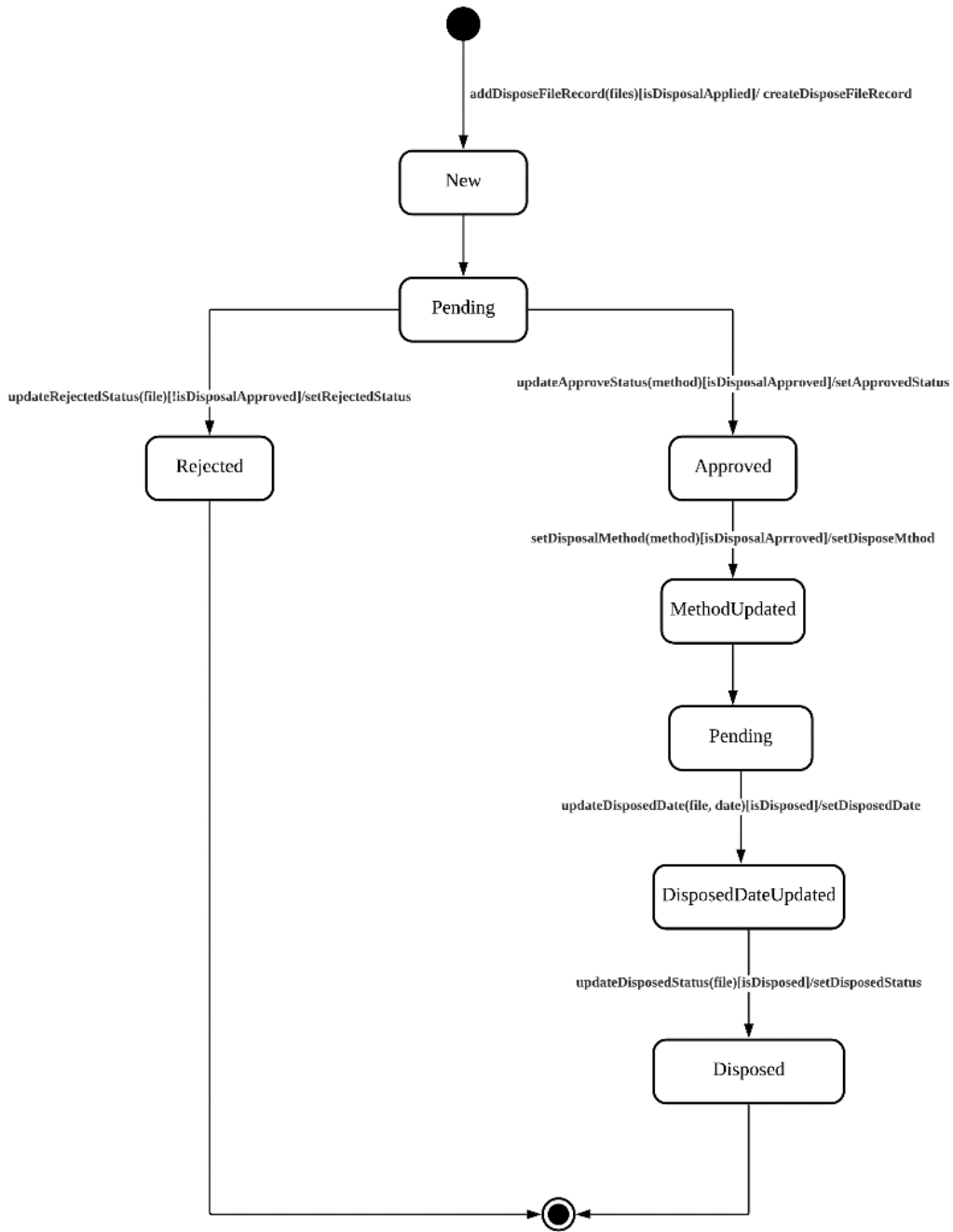


Figure 4.3: State-Transition Diagram for <FileDisposedRecord Class>

For Figure 4.3, a new FileDisposedRecord Class will be created when we add the dispose file record in the system. This made the state of the class to change into new. Then, the state will be change into pending as the FileDisposedRecord Class need to make sure that it can be disposed. After that, the FileDisposedRecord Class will be in two different state which is rejected that means the disposed file is rejected and approved that means the file is approved to be disposed. For rejected, the state transition end there meanwhile for approved the state then will change into methodUpdated. It will update the method and the state will change into pending to make sure the file can be disposed. Next, the state will change into DisposalDateUpdated to set the date of the file to be disposed and display the disposed date. Lastly, the state will change into disposed by update the disposed status and the state transition end here.

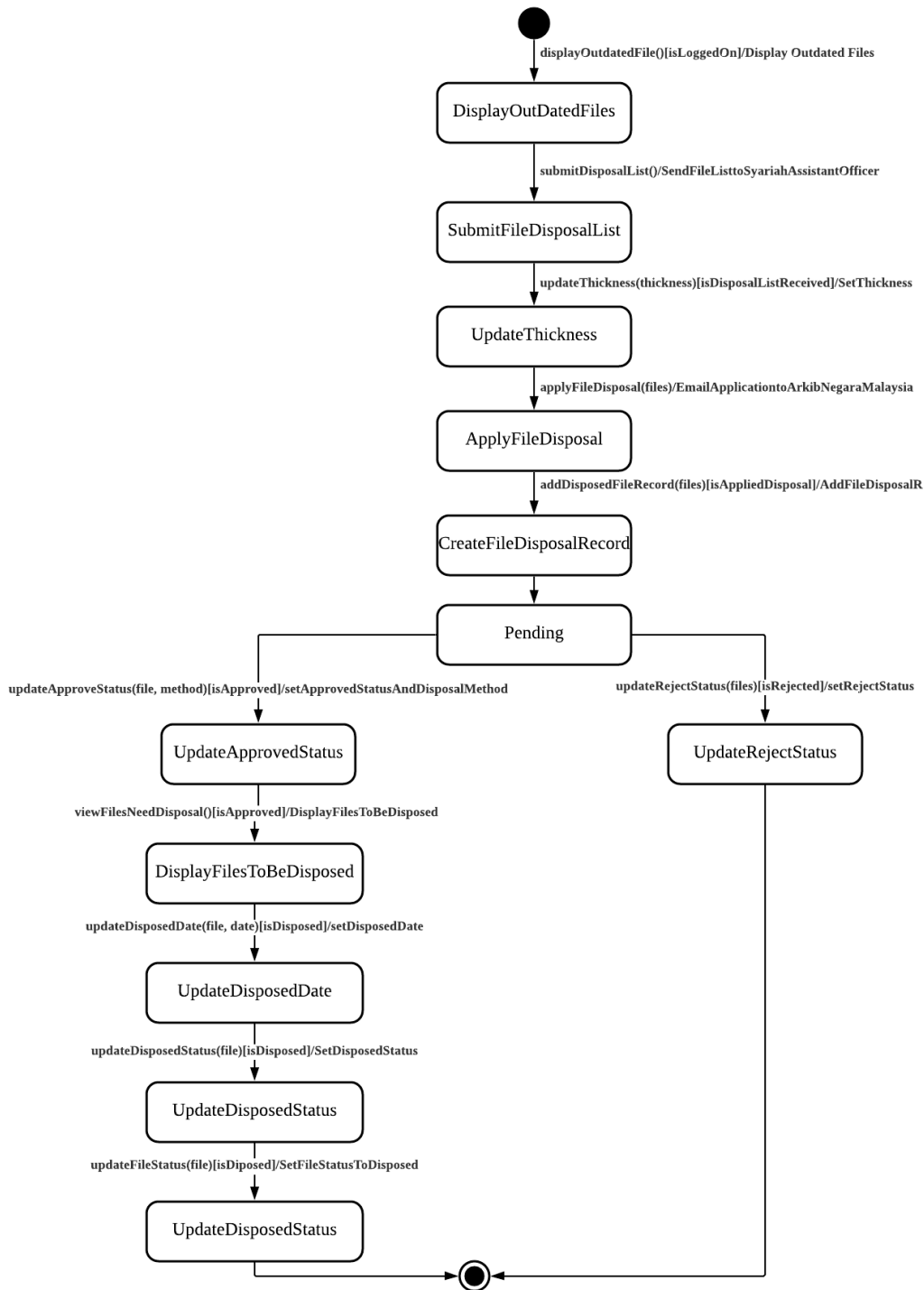


Figure 4.4: State-Transition Diagram for <disposedFileHandler Controller>

For Figure 4.4, the state of the disposedFileHandler Controller will change into DisplayOutDatedFiles. Then, it will change to SubmitFileDisposalList once the user submits it. After that, it will change to UpdateThickness, then ApplyFileDisposal state is triggered. Once apply, it goes to CreateFileDisposalRecord state and change to pending after that. If the request is rejected, it will change to UpdateRejectStatus state and end the transition. On the other hand, it will change to UpdateApprovedStatus state, then UpdateDisposedDate is triggered after the DisplayFilesToBeDisposed is triggered. Then the UpdateDisposedStatus is triggered finally, and the state transition ends here.

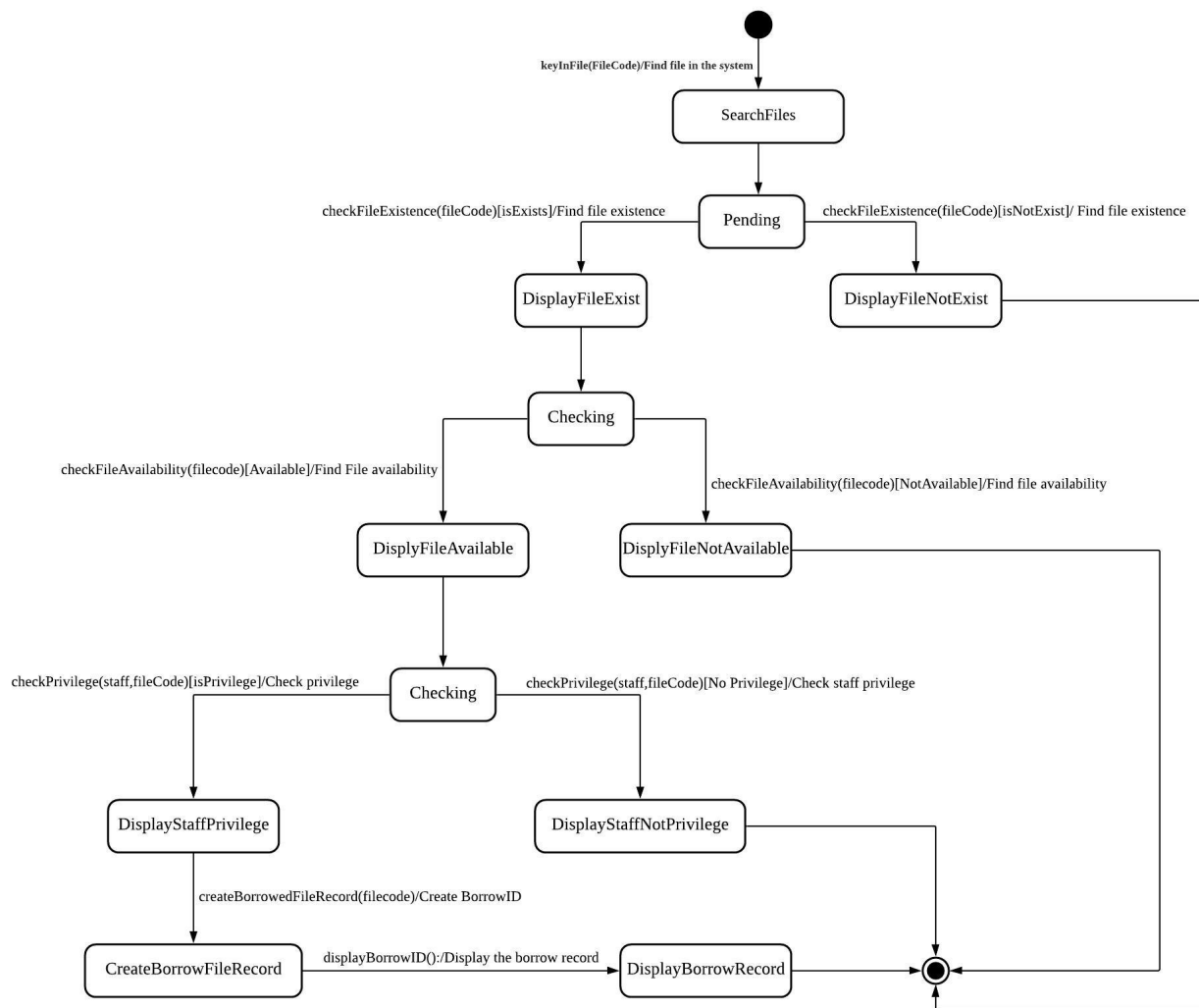


Figure 4.5: State-Transition Diagram for <borrowFileHandler Controller>

For Figure 4.5, the state of the borrowFileHandler Controller will change into SearchFile when the file code is fill in the system. Then, the state change into pending to check either the file exists or not. They will be two different states after pending which are DisplayFileNotExist to show that the file is not exist on the system and DisplayFileExist to show the file exist in the system. For DisplayFileNotExist, the state transition will end after that meanwhile for DisplayFileExist the state change into checking to check either the file is available or not. They will be two different states after checking which are DisplayFileNotAvailable to show that the file is not available right now and DisplayFileAvailable to show the file is available. For DisplayFileNotAvailable, the state transition will end after that meanwhile for DisplayFileAvailable the state change into checking to check either the staff has privilege or not to borrow the file. They will be two different states after checking which are DisplayStaffNotPrivilege to show that the staff has no privilege and DisplayStaffPrivelege to show the staff has privilege. For DisplayStaffNotPrivilege, the state transition will end after that meanwhile for DisplayStaffPrivelege the state change into CreateBorrowFileRecord to create a new borrowID. Lastly, the state will change into DisplayBorrowRecord to show the borrow record of the user and then the state transition end.

5. Conclusion

Based on Phase 2 report, there are many things that we have learned by doing this project. Firstly, we have learned on how to create a use case diagram. Other than that, we also learn on how to describe the use case, activity diagram and create a sequence diagram based on every use case. We also create a domain modal to describe all the entities included in the system. We also create a state diagram to show all the state and argument that made the state of the entity to change in the system. Take note that not all the entities have a different state therefore we only included all the entities that may have any change of state.

By doing phase 2 report, our group has learned many new things as we need to create many new diagrams. By creating this diagram, it helps us to organize and understand more about the information needed to be included in our system. This is because all the detailed information will be displayed in the diagram. Other than that, we also learn on teamworking skills. This is because they are many works needed to do in phase 2 so if they are only one person that do the work, we cannot finish this phase in time. Therefore, we need to divide the work equally. Other than that, besides doing the work that we need to do, we also always help each other when they are any team members that has any problem to do their works. We learn that communication is a key when doing the project with other people.

So, to conclude the Phase 2 report, they are many things that we have learned by doing this project and we are glad to learn all the lesson by doing all the works given in phase 2.

6. References

- Athuraliya, A. (2021, April 22). What is Sequence Diagram? Complete Guide with Examples. Retrieved from creately: <https://creately.com/blog/diagrams/sequence-diagram-tutorial/>
- Kruger, N. (2018, October 23). *How to Write a Software Requirements Specification (SRS Document)*. Retrieved from perforce: <https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document>
- Malaysia, A. N. (2021, April 1). *Dasar Garis Panduan Penilaian dan Pemisahan Rekod Elektronik*. Retrieved from Arkib Negara Malaysia: <http://www.arkib.gov.my/web/guest/penilaian-dan-pemisahan-rekod-elektronik>
- Sharma, P. (n.d.). *System Documentation: Features, Purpose and Contents*. Retrieved from YourArticleLibrary: <https://www.yourarticlelibrary.com/management/mis-management/system-documentation-features-purpose-and-contents-mis/70408>