FLASH BOOTLOADER SOFTWARE DOCUMENT

Table of Contents

Content entered directly below the header but before the first section heading is called the preamble.

1. Introduction and goal

This is a paragraph about role of **Flash bootloader software**.

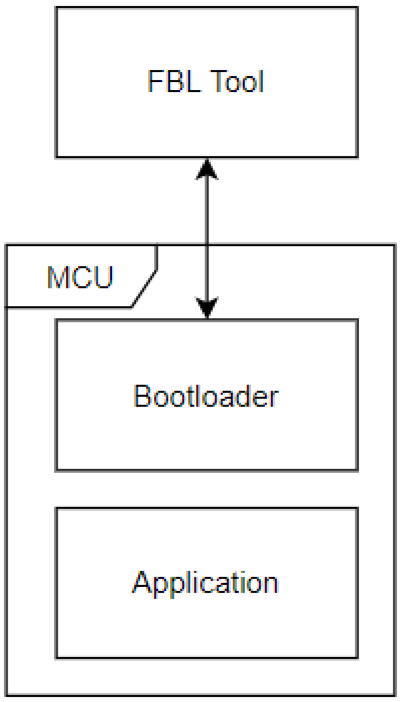


Figure 1. Image

1.1 Requirements overview

Flash bootloader is a software in flash. It allows to reflash application software over CAN communication. Tool can be used to send request and receive respond from ECU.

Main features:

- It is the first software when MCU power-on
- It is responsible for FBL service check validation and respond
- It control checking and jumping to application software

Table 1. The following goals have been established for this system:

Priority	Explicite functional Requirement
1	FBL SW shall support reflashing service via Tool
1	FBL SW shall support checking validity of application software before decision whether jump to app or not
2	FBL SW shall have size less than 10K (*)
2	FBL SW shall support checking validity of receiving message in specical case (**)
1	FBL SW shall fit with MCU without Non-volotile memory

(*) Accepted Maximum size is 14K

(**) specical case: receive new application SW

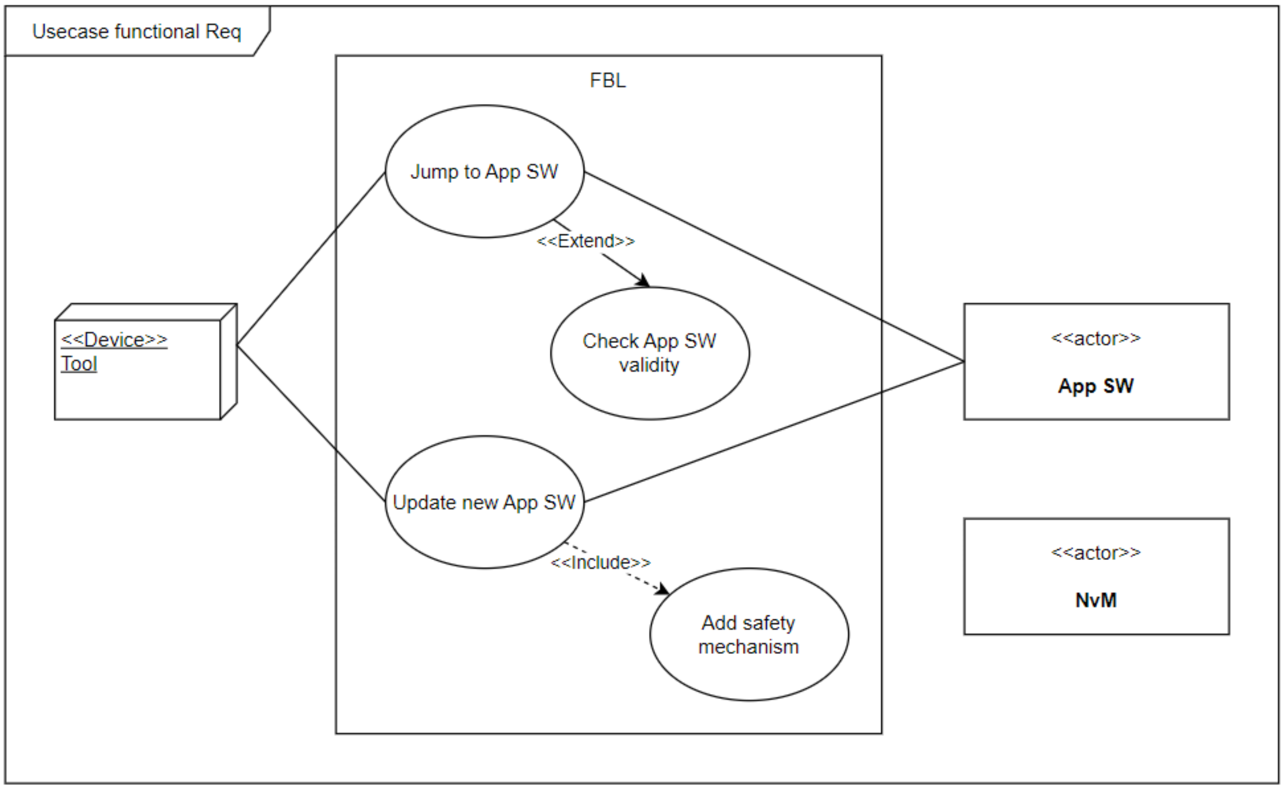


Figure 2. Image

1.2 Quality goals

Table 2. Table title

Quality attribute	Description	Quality scenarios
Performance - Resource utilization	SW has size less than 10K	SC1
Reliability - Availability		CS2
Portability - Adaptability	SW shall be easy to adapt between HW with Nvm and without Nvm	CS3

CS1

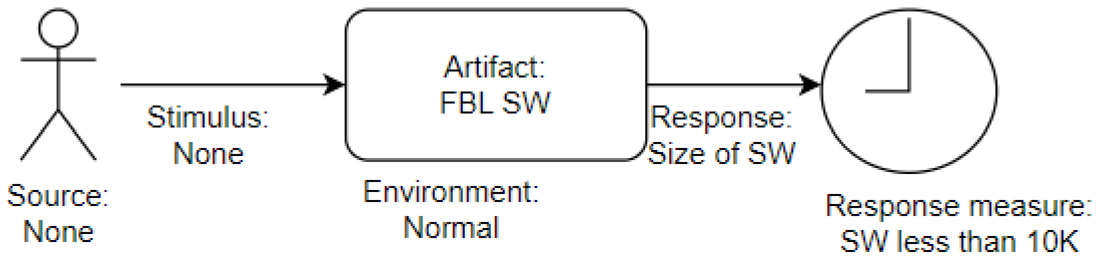


Figure 3. Performance scenario

CS2

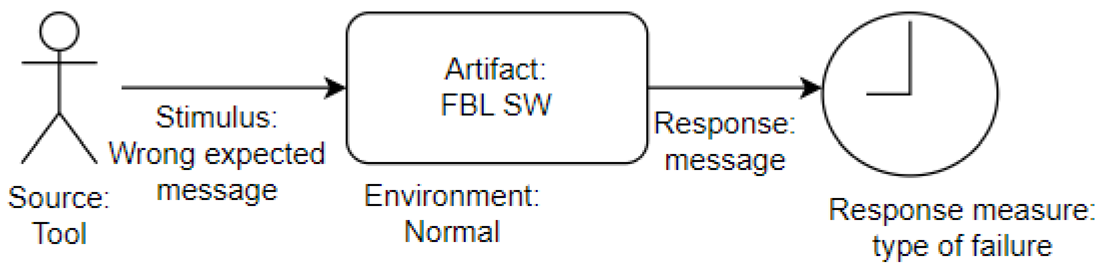


Figure 4. Reliability scenario

CS3

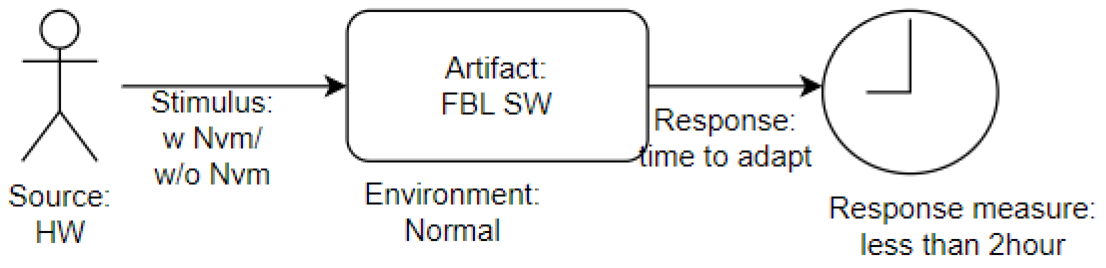


Figure 5. Portability scenario

2. Contraints

- Only support STM32F103C8T6 Hardware
- Porting hardware. Hardware must satisfy:
    - RAM minimunsize: 20K
    - ROM minimunsize: 4K
    - Support CAN standard hardware
    - Integration MCAL and Application. Required MCAL:
        - STD Driver library

3. Context and Scope

3.1 Business context

Business context descript data exchanged with the environment of the system.

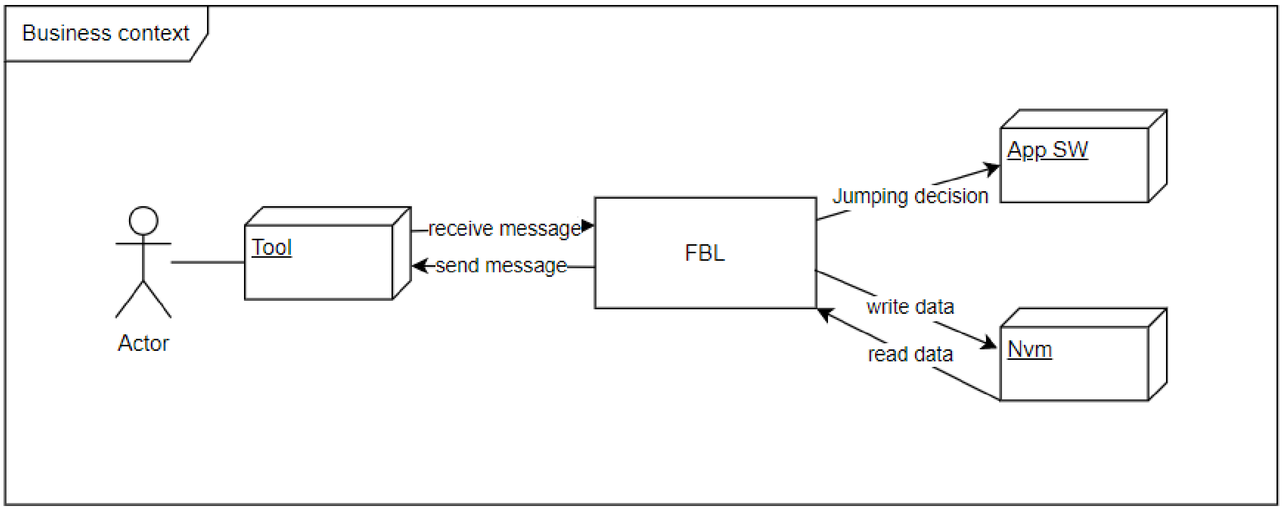


Figure 6. Business context

3.2 Technical context

Technical context descript technical interfaces (channels and transmission media) between the system and its context.

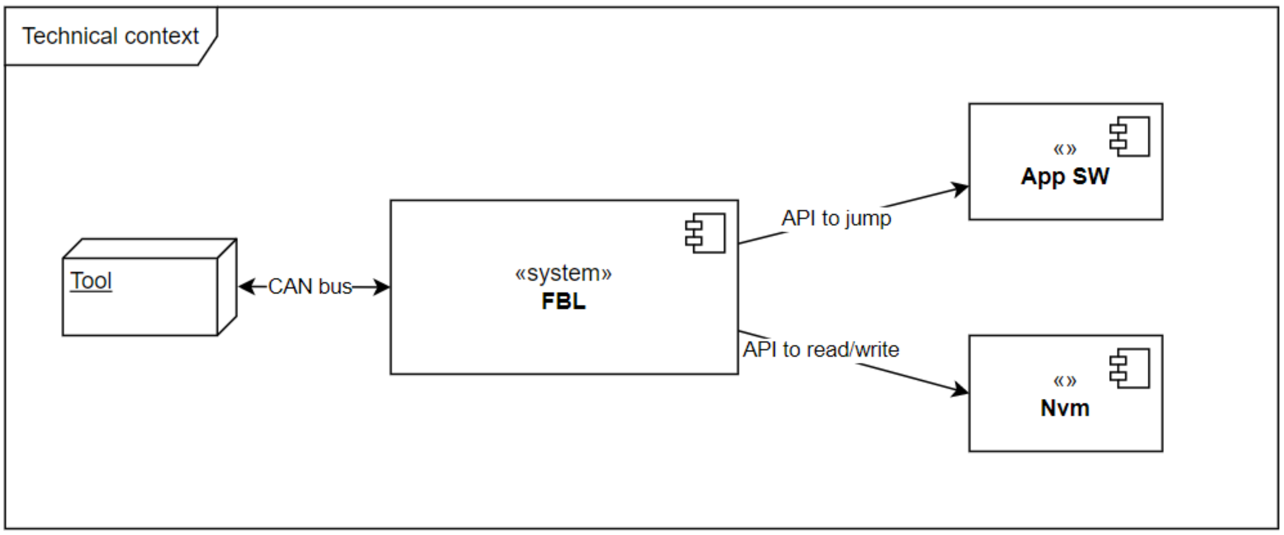


Figure 7. Technical context

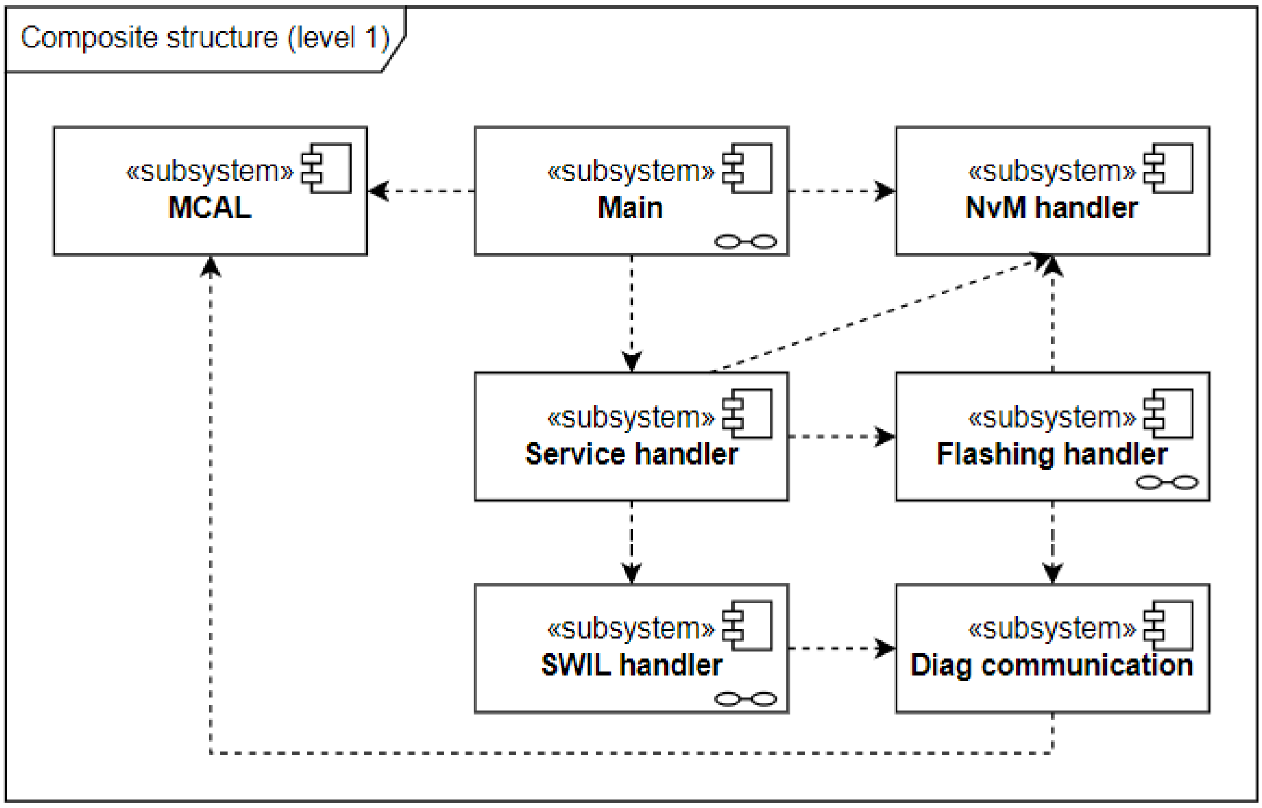
4. Solution strategy

Scenario	Solution approach	Link to Details
Development environment	Setup package include all tool and use bat script to build sw	DDD-Development environment SDD-Development environment
Communication between modules in software	Use global variable to share data between modules with safety machanism	DDD-Communication between modules SDD-Communication between modules

Scenario	Solution approach	Link to Details
Communication protocol with external tool	Provide messsage format and sequence of request	IDD-Tool communication protocol
Software safety flashing machanism	Flash driver will be stored in tool and update to FBL when request	DDD-software interlock SDD-software interlock

5. Building block view

5.1 Building Block View Level 1



Composite structure (level 1)
«subsystem» MCAL
«subsystem» Main
«subsystem» NvM handler
«subsystem» Service handler
«subsystem» Flashing handler
«subsystem» SWIL handler
«subsystem» Diag communication

Figure 8. Building Block View Level 1

Main module	Setup hardware configuration, run main state machine
MCAL module	Provide driver API for hardware configuration
Service handler module	Check validity of request, call servide
SWIL handler module	Handle write SWIL to RAM
Flashing handler module	handle erase application software and flash new application
Diag communication module	Provide API for diag communication
Nvm handler	Provide API to read/write data in Nvm

5.2 Building Block View Level 2

5.2.1 Main module

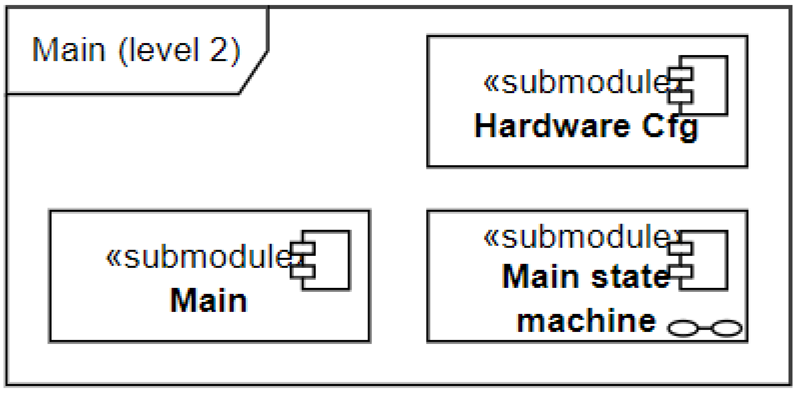


Figure 9. Main module

5.2.2 MCAL module

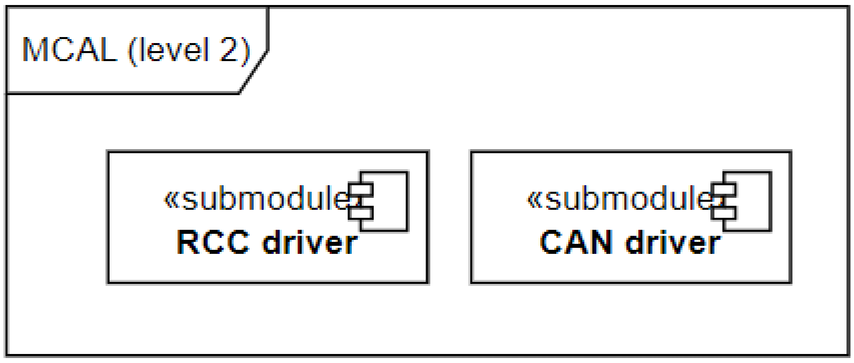


Figure 10. MCAL module

5.2.3 Service handler module

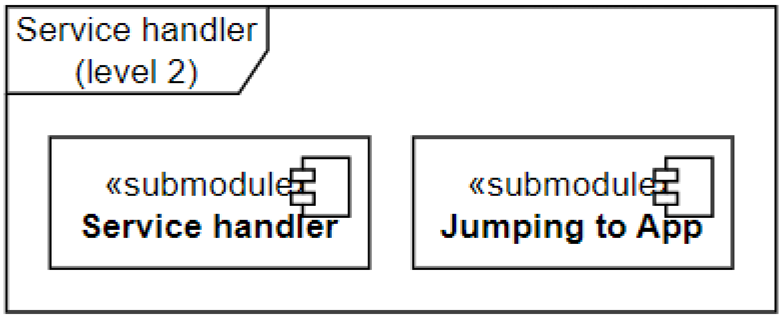


Figure 11. Service module

5.2.4 SWIL handler module

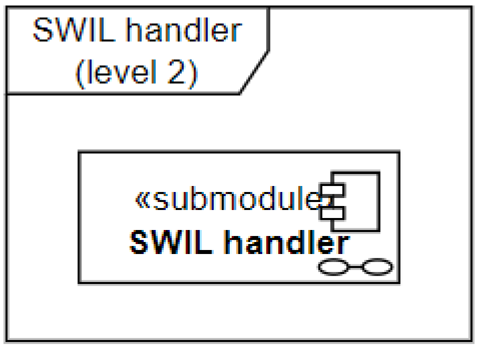


Figure 12. SWIL module

5.2.5 Flashing handler module

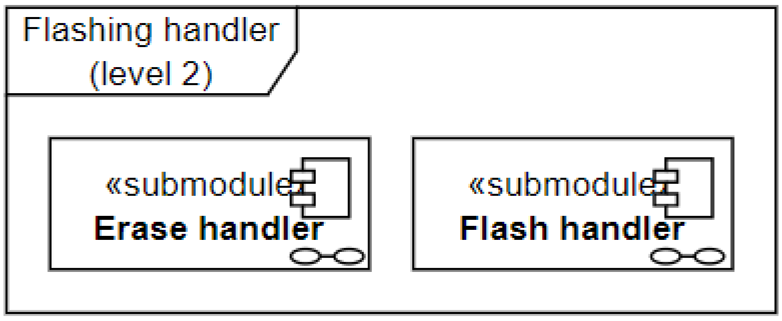


Figure 13. Flash module

5.2.6 Diag communication module

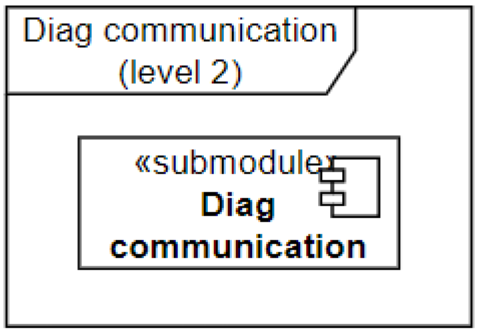


Figure 14. Flash module

5.2.7 Nvm handler module

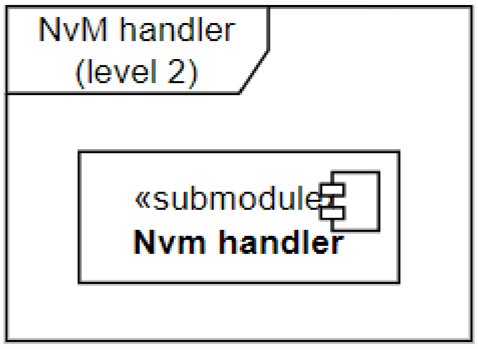


Figure 15. Nvm module

6. Runtime view

6.1 Main

6.1.1 Main statechart diagram

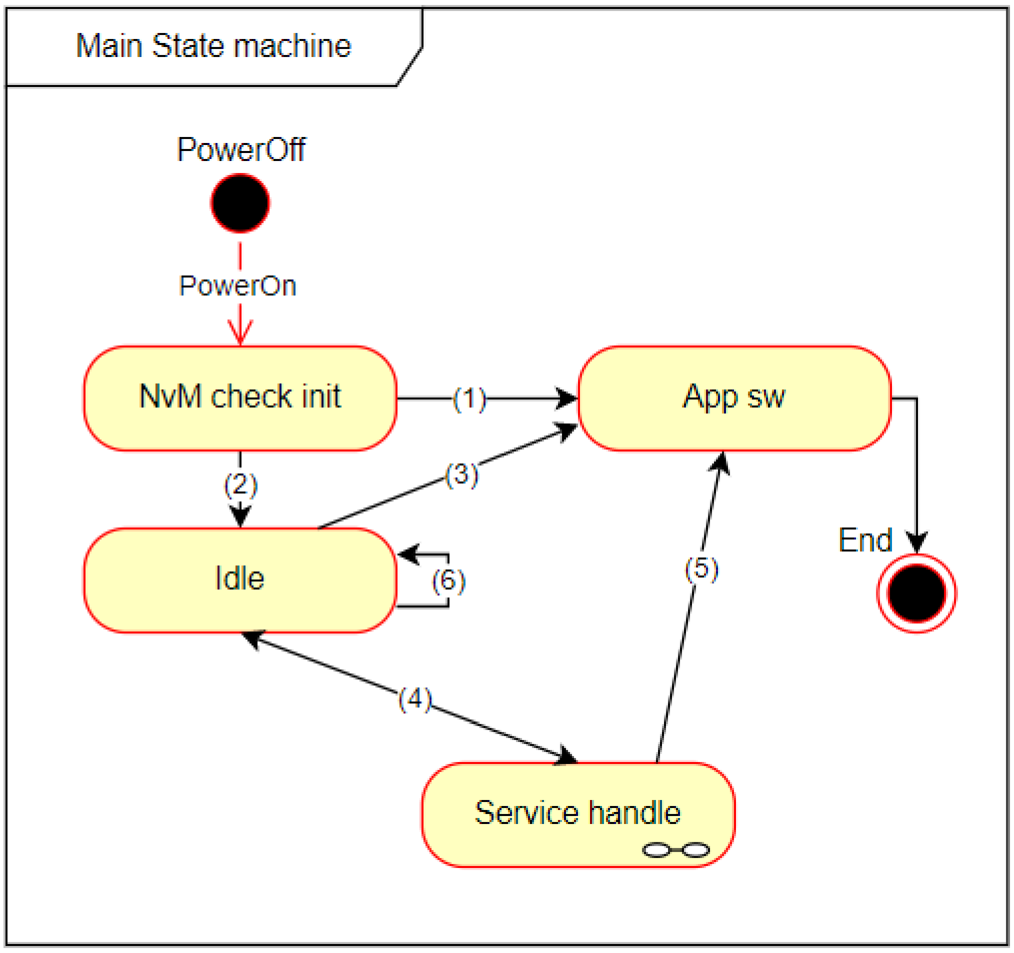


Figure 16. Main statechart

6.1.2 Main sequence diagram

6.1.3 Main activity diagram

6.2 Handling SWIL

6.2.1 Handling SWIL statechart diagram

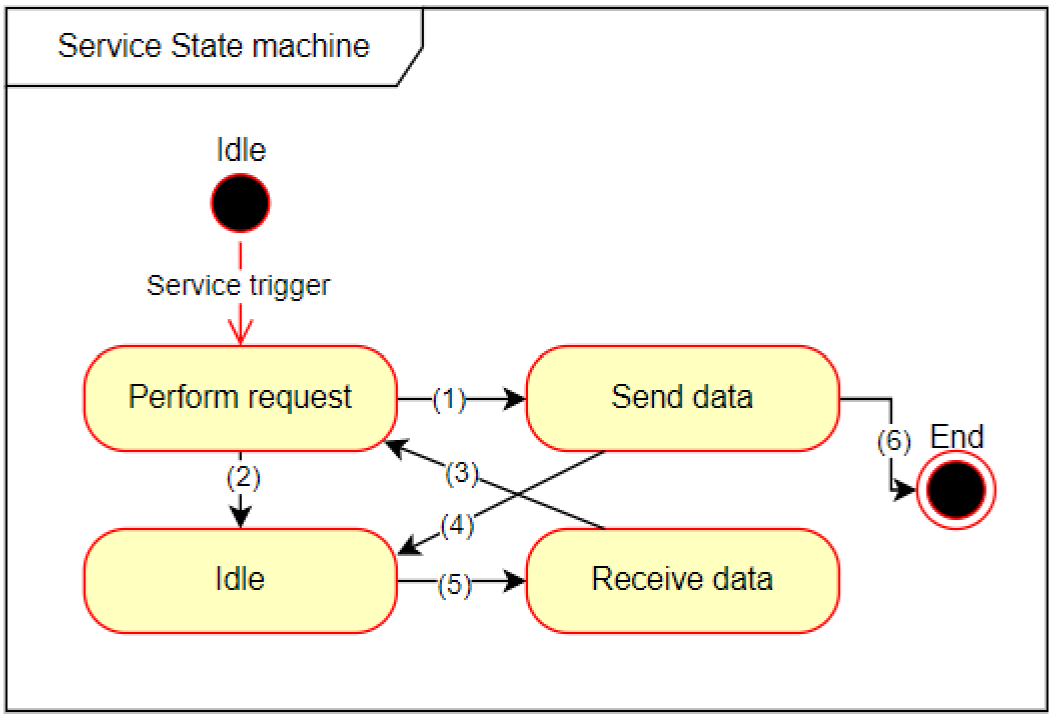


Figure 17. Handling SWIL statechart

6.2.2 Handling SWIL sequence diagram

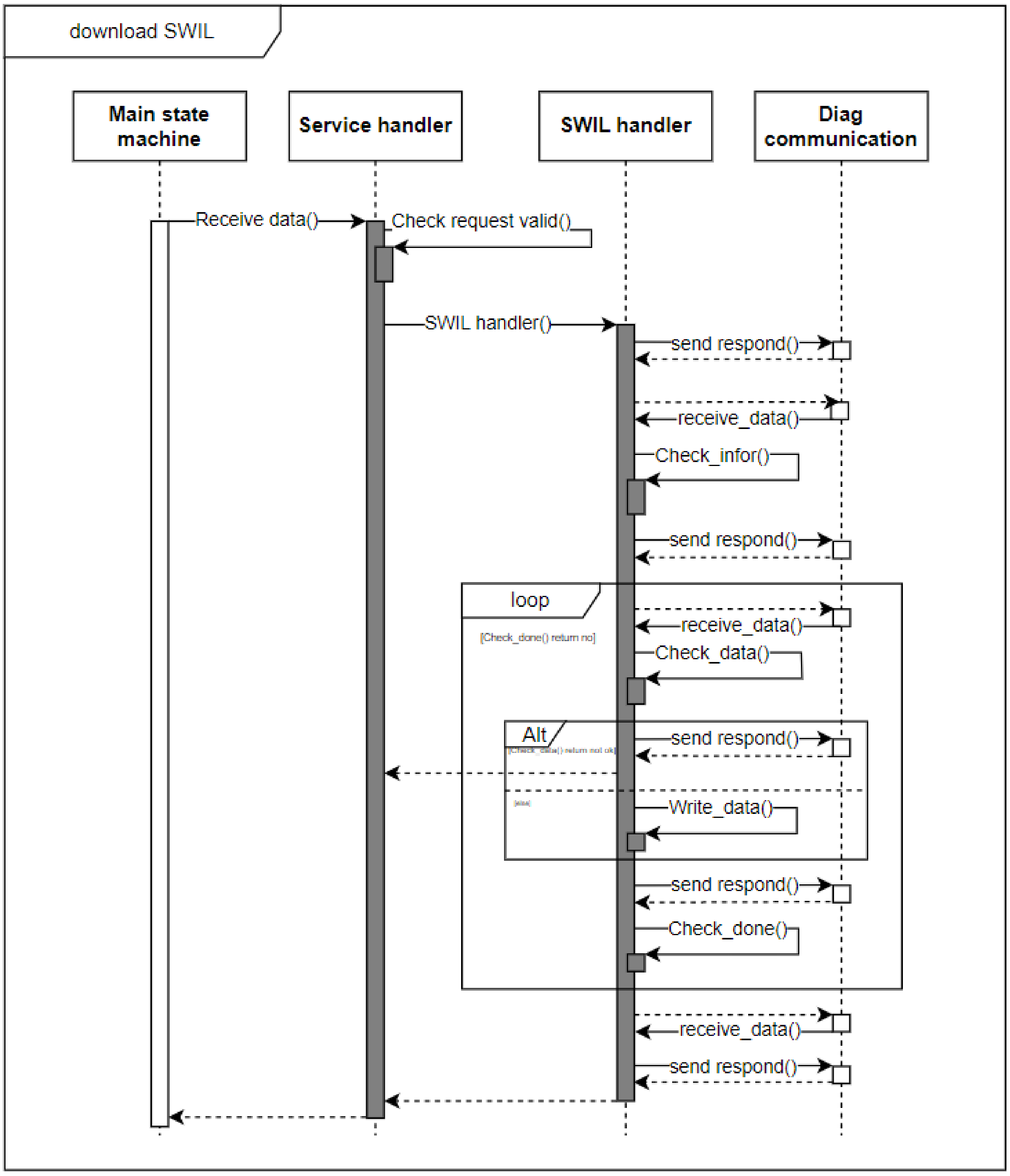


Figure 18. Handling SWIL sequence

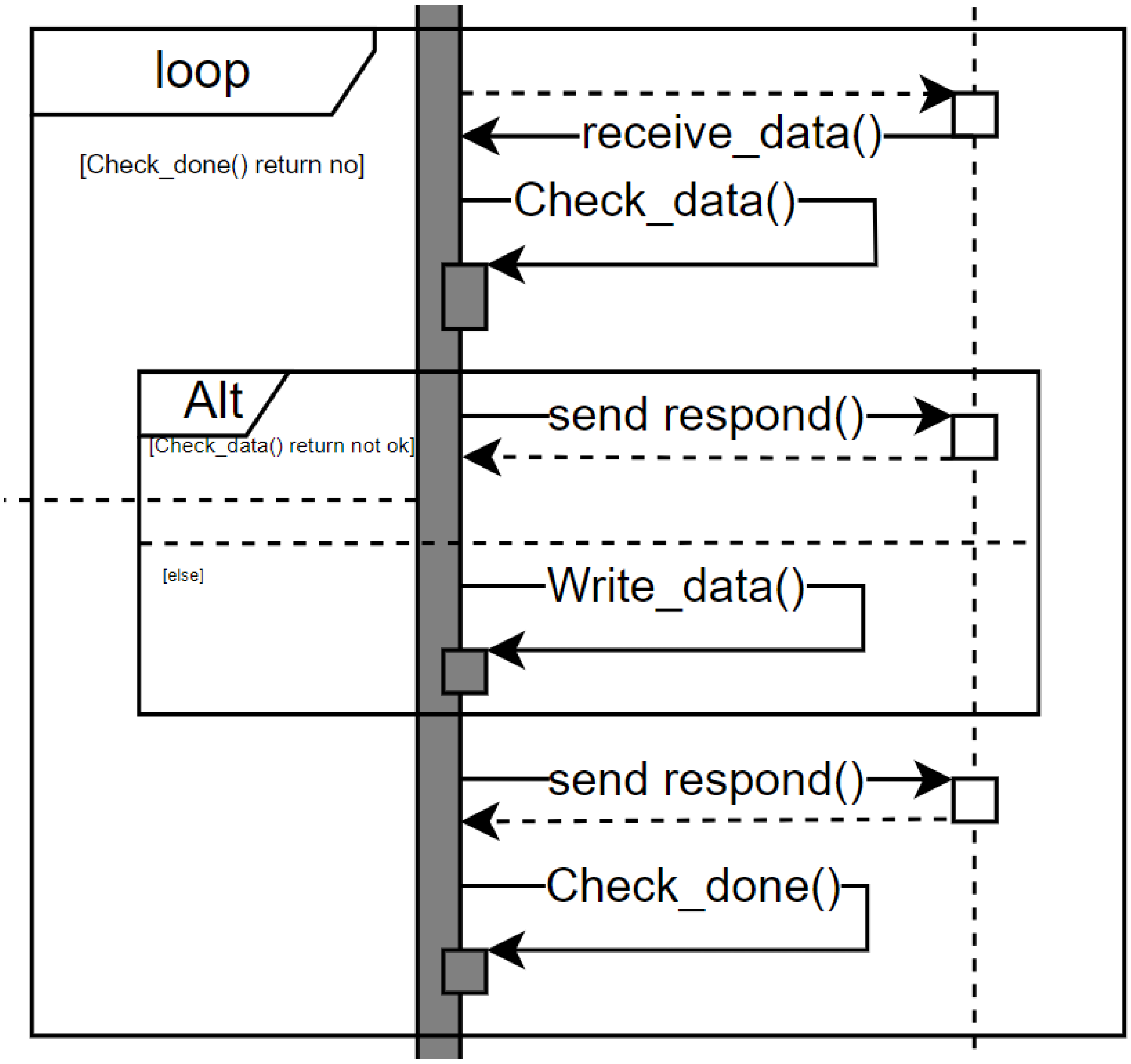
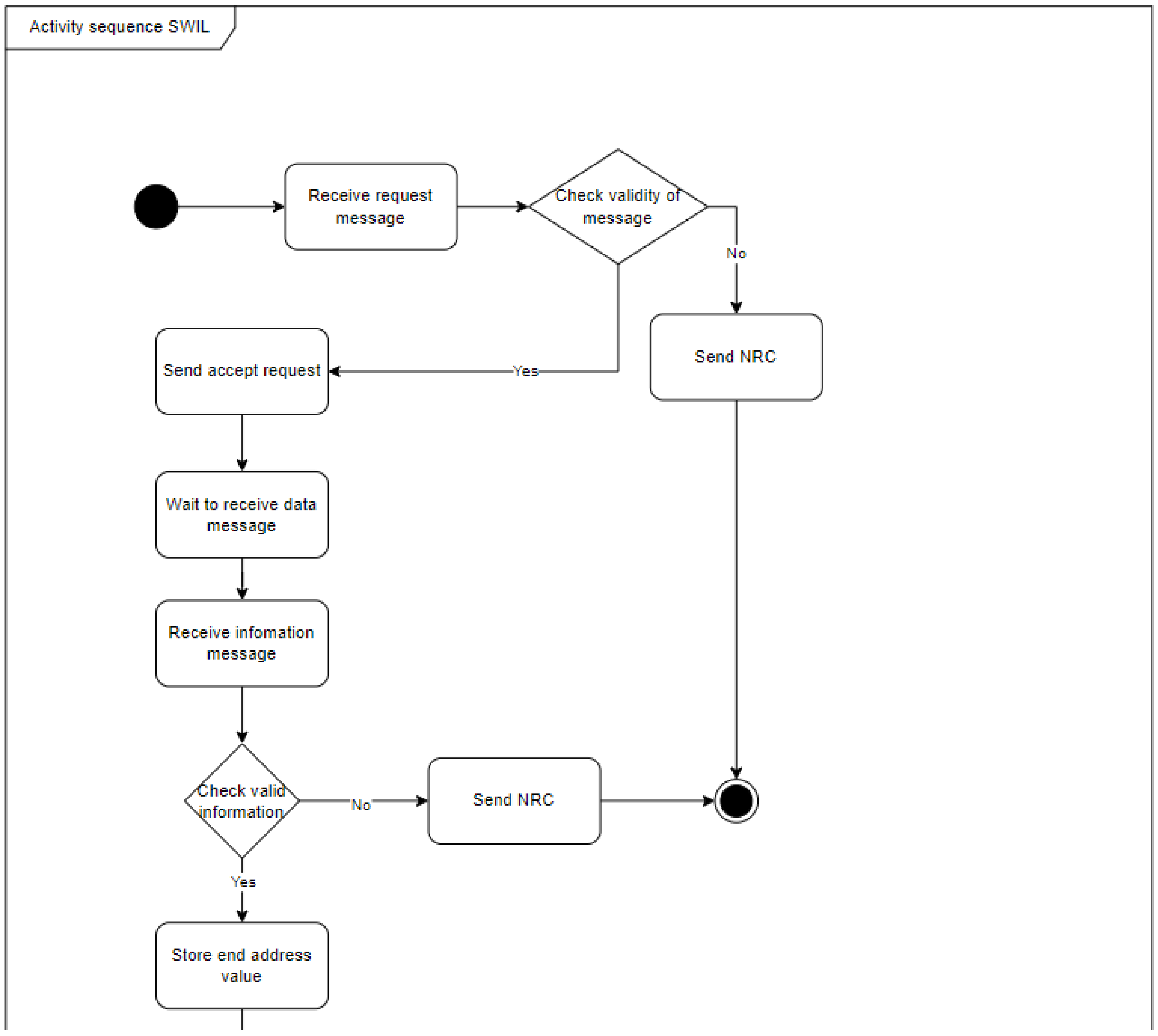
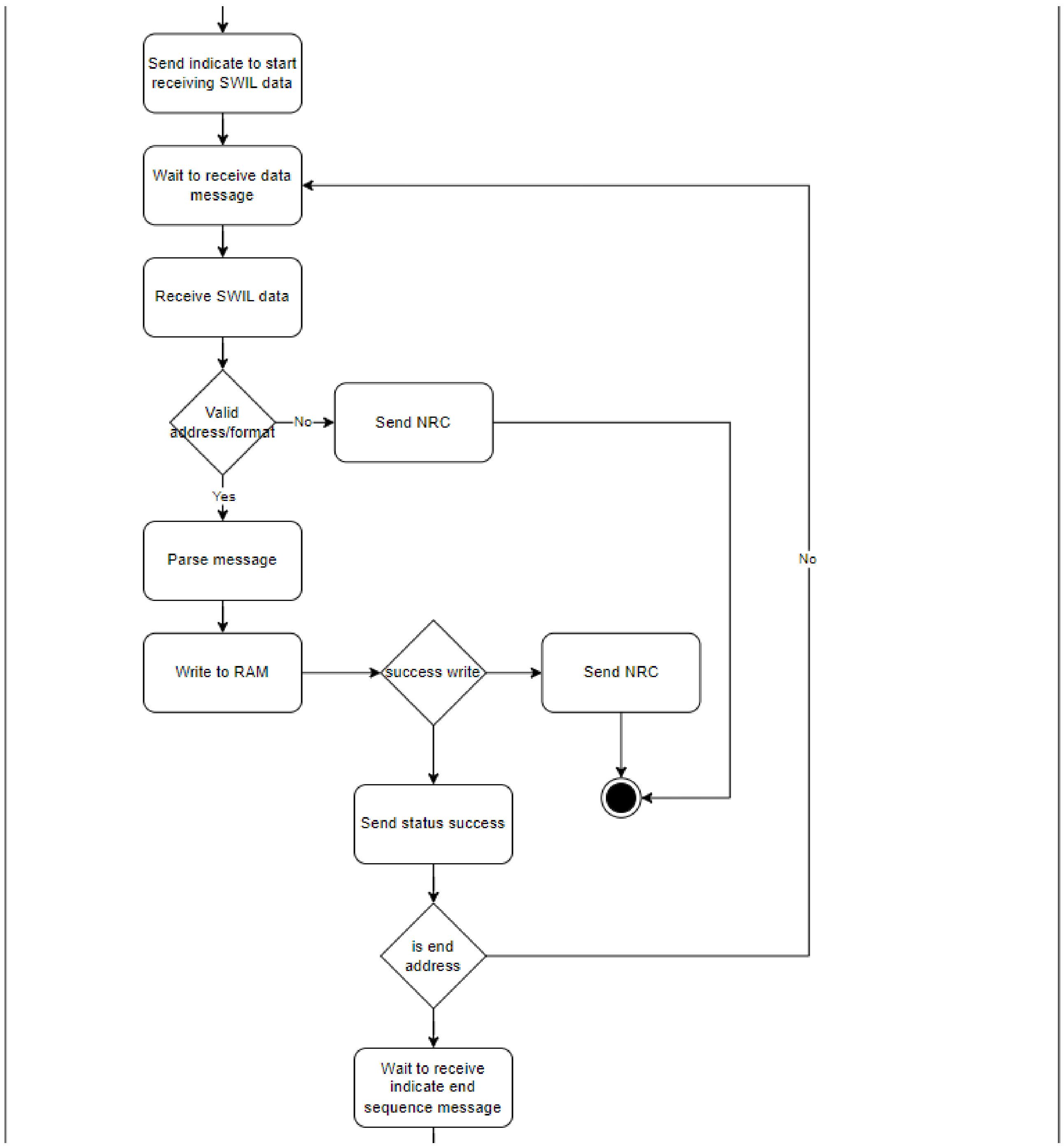


Figure 19. Loop sequence

6.2.3 Handling SWIL activity diagram


Activity sequence SWIL
Receive request message
Check validity of message
No
Send NRC
Yes
Send accept request
Wait to receive data message
Receive infomation message
Check valid information
No
Send NRC
Yes
Store end address value


Send indicate to start receiving SWIL data
Wait to receive data message
Receive SWIL data
Valid address/format
No
Send NRC
Yes
Parse message
Write to RAM
success write
Send NRC
Send status success
is end address
No
Wait to receive indicate end sequence message

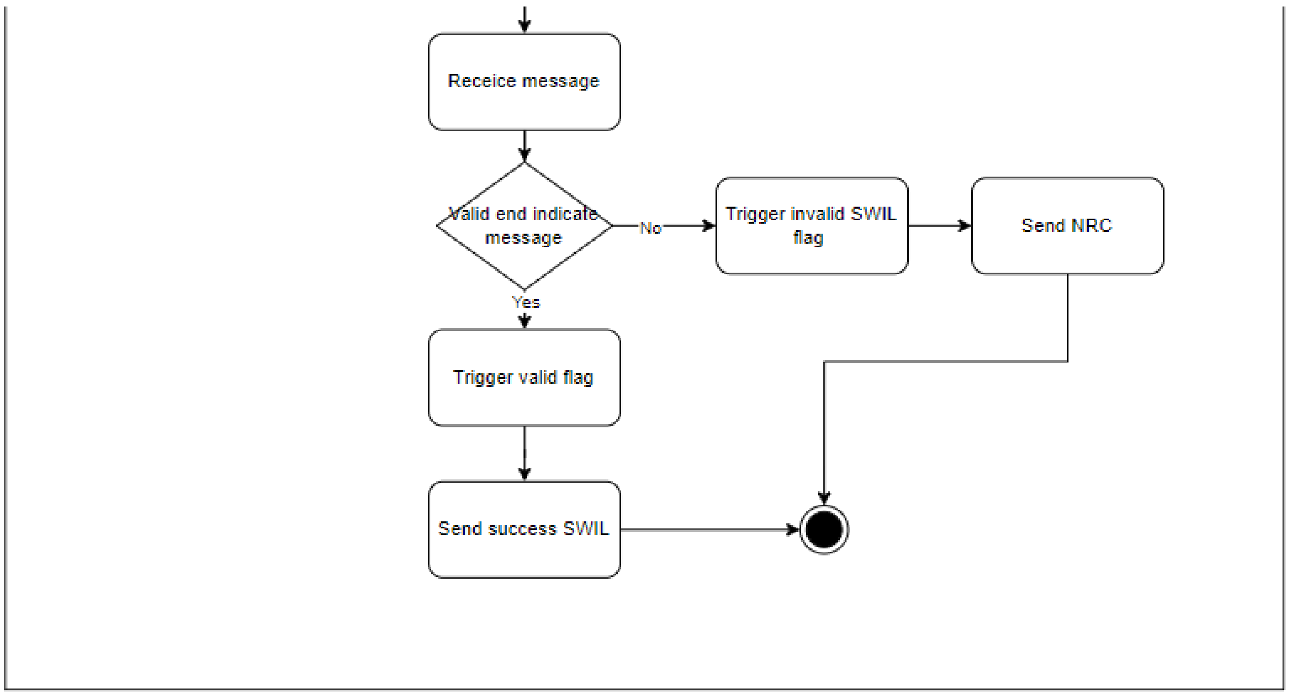


Figure 20. SWIL handle sequence

6.3 Erase handle

6.3.2 Erase handle statechart diagram

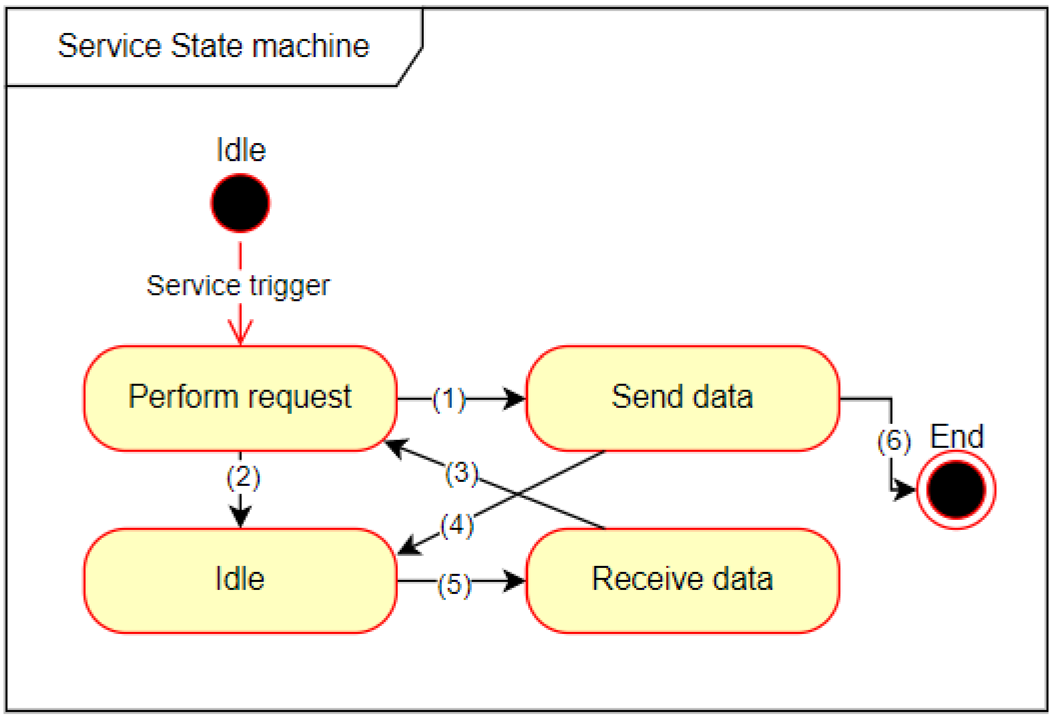


Figure 21. Erase handle statechart

6.3.2 Erase handle sequence diagram

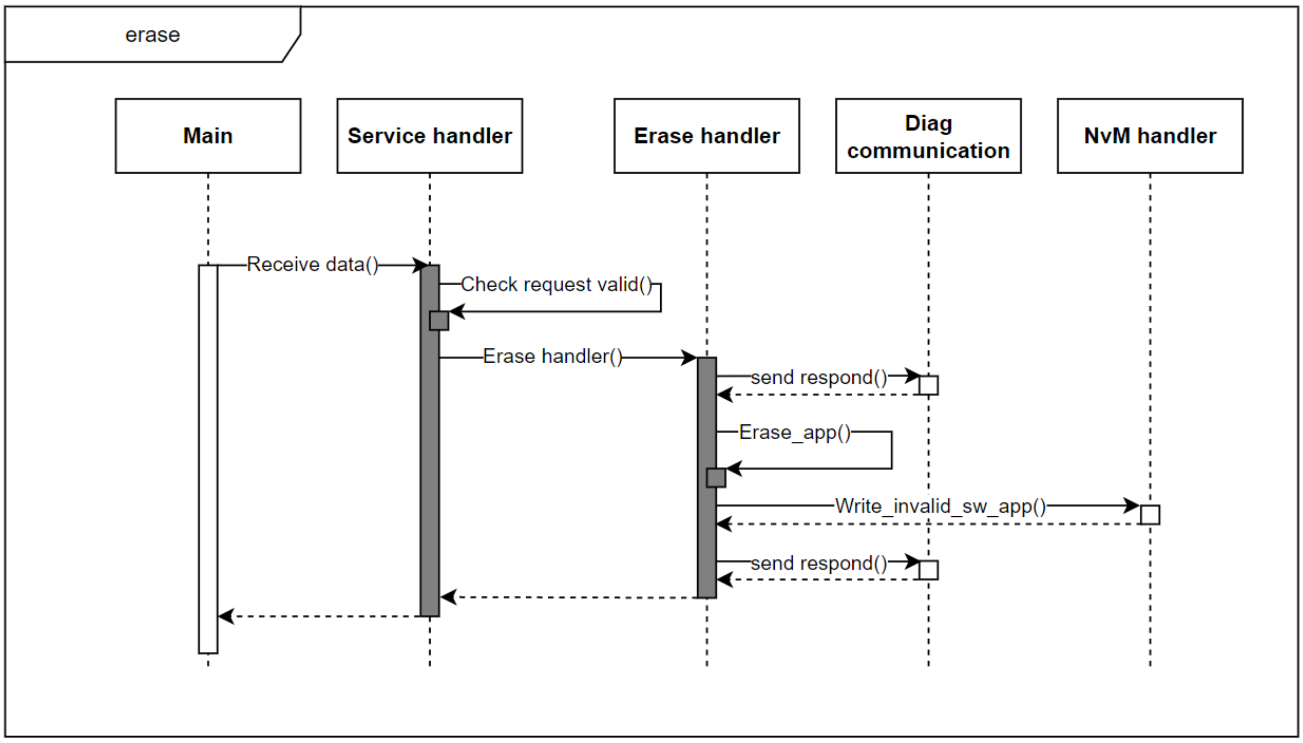


Figure 22. Erase handle sequence

6.3.3 Erase handle activity diagram

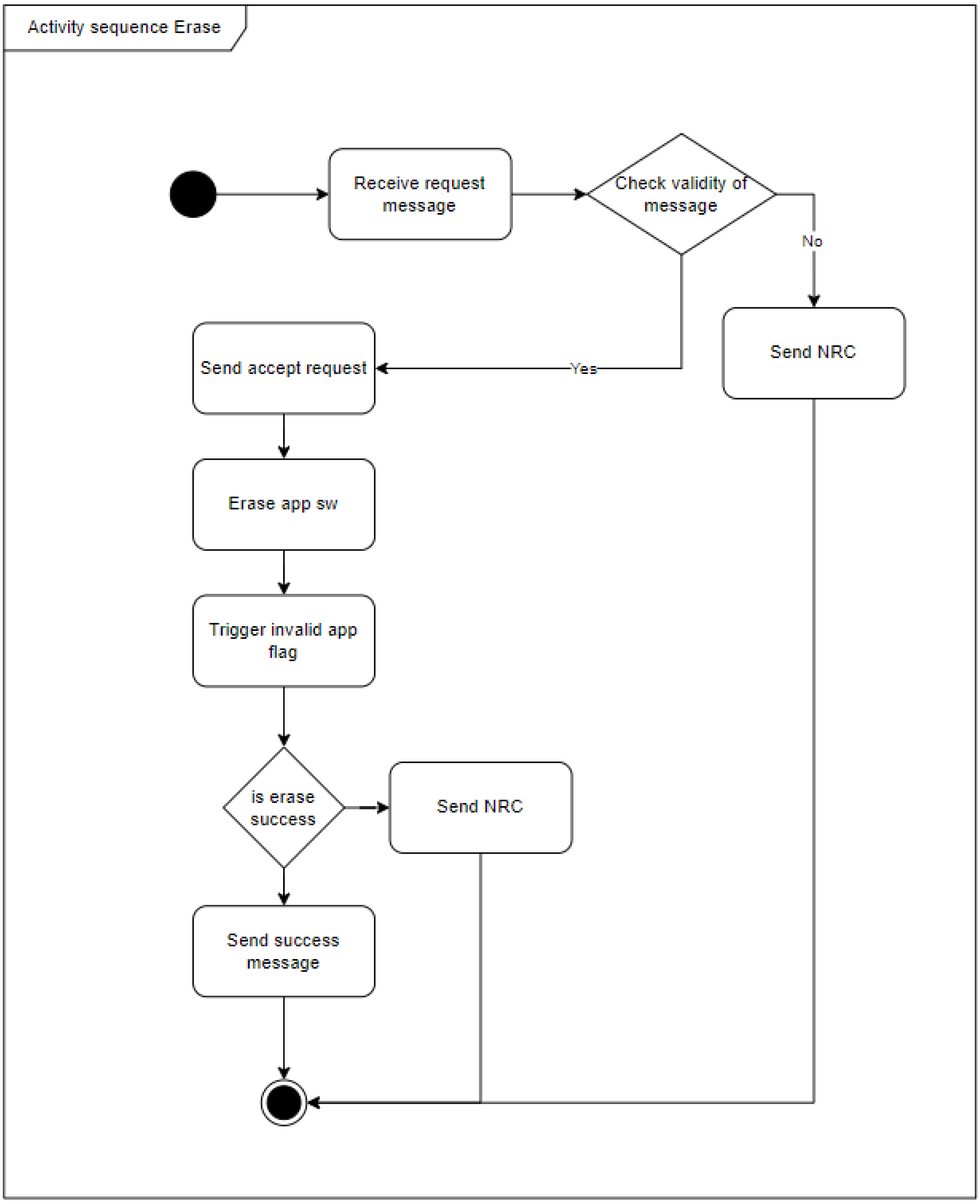


Figure 23. Erase handle sequence

6.4 Flash handle

6.4.1 Flash handle statechart diagram

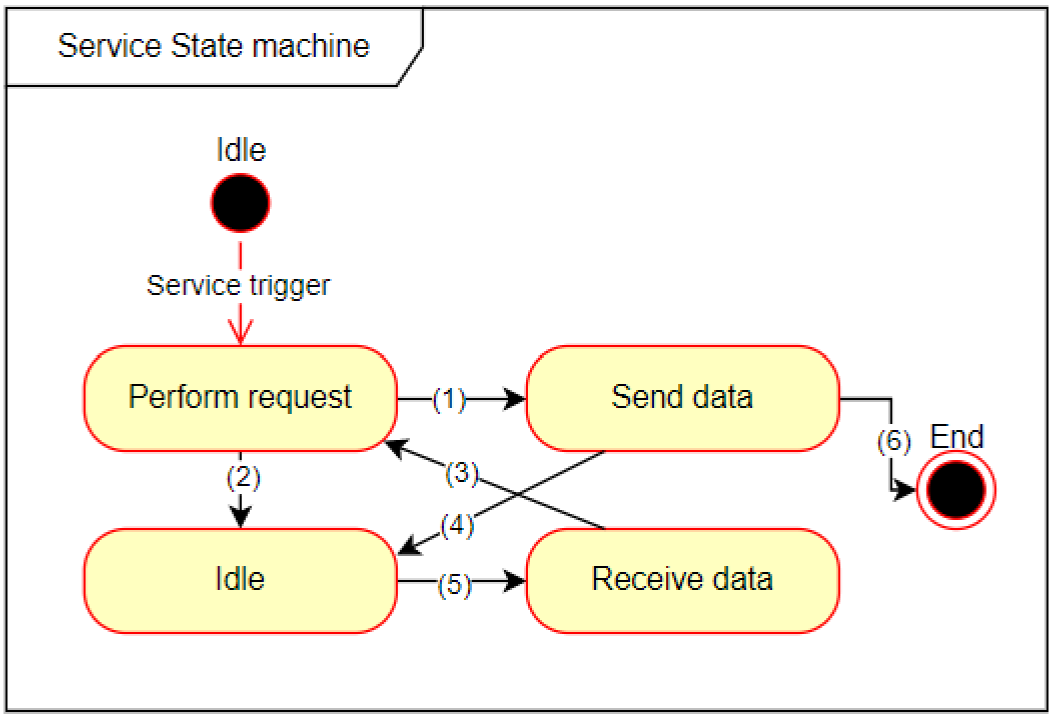


Figure 24. Flash handle statechart

6.4.2 Flash handle sequence diagram

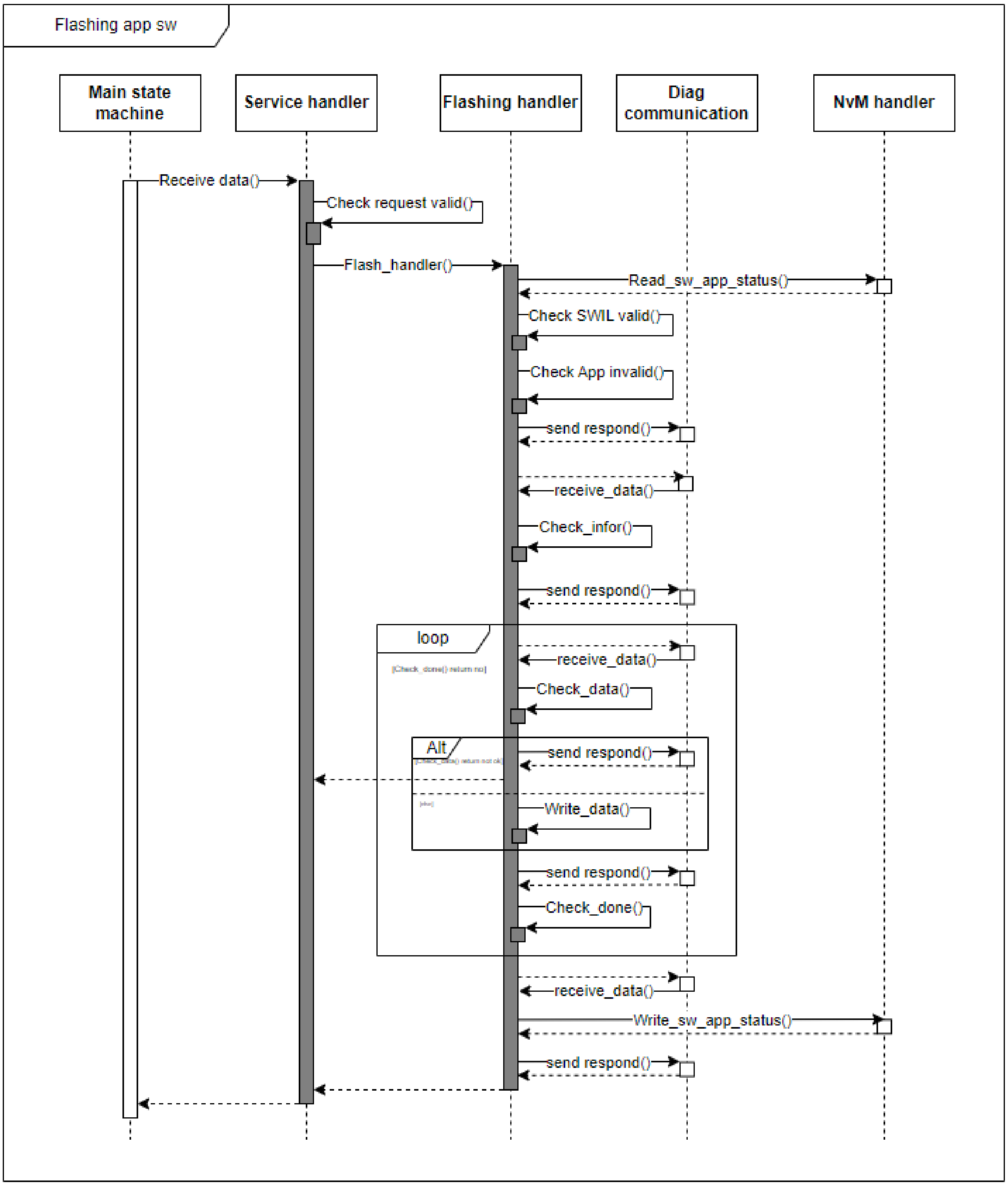


Figure 25. Flash handle sequence

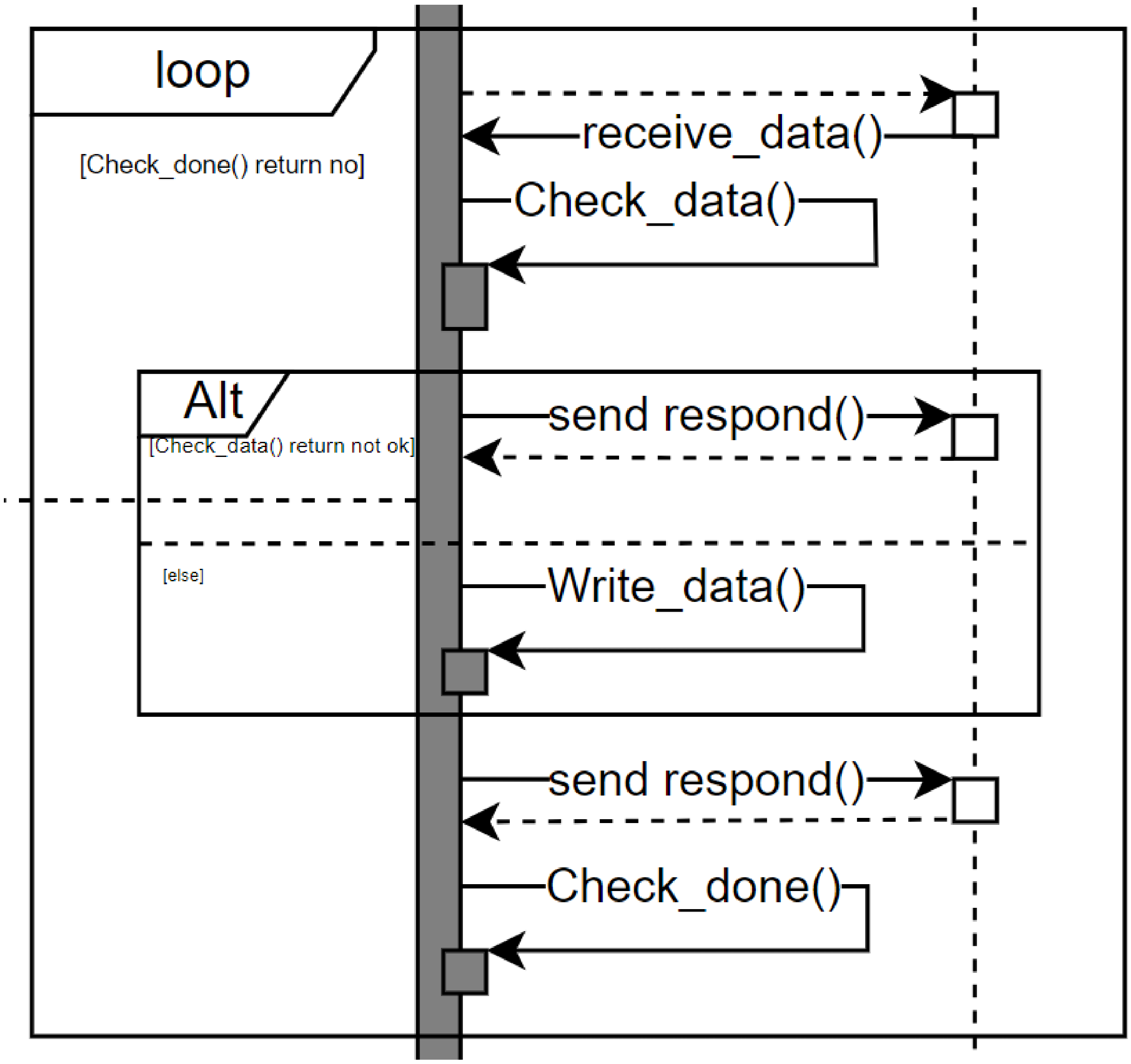
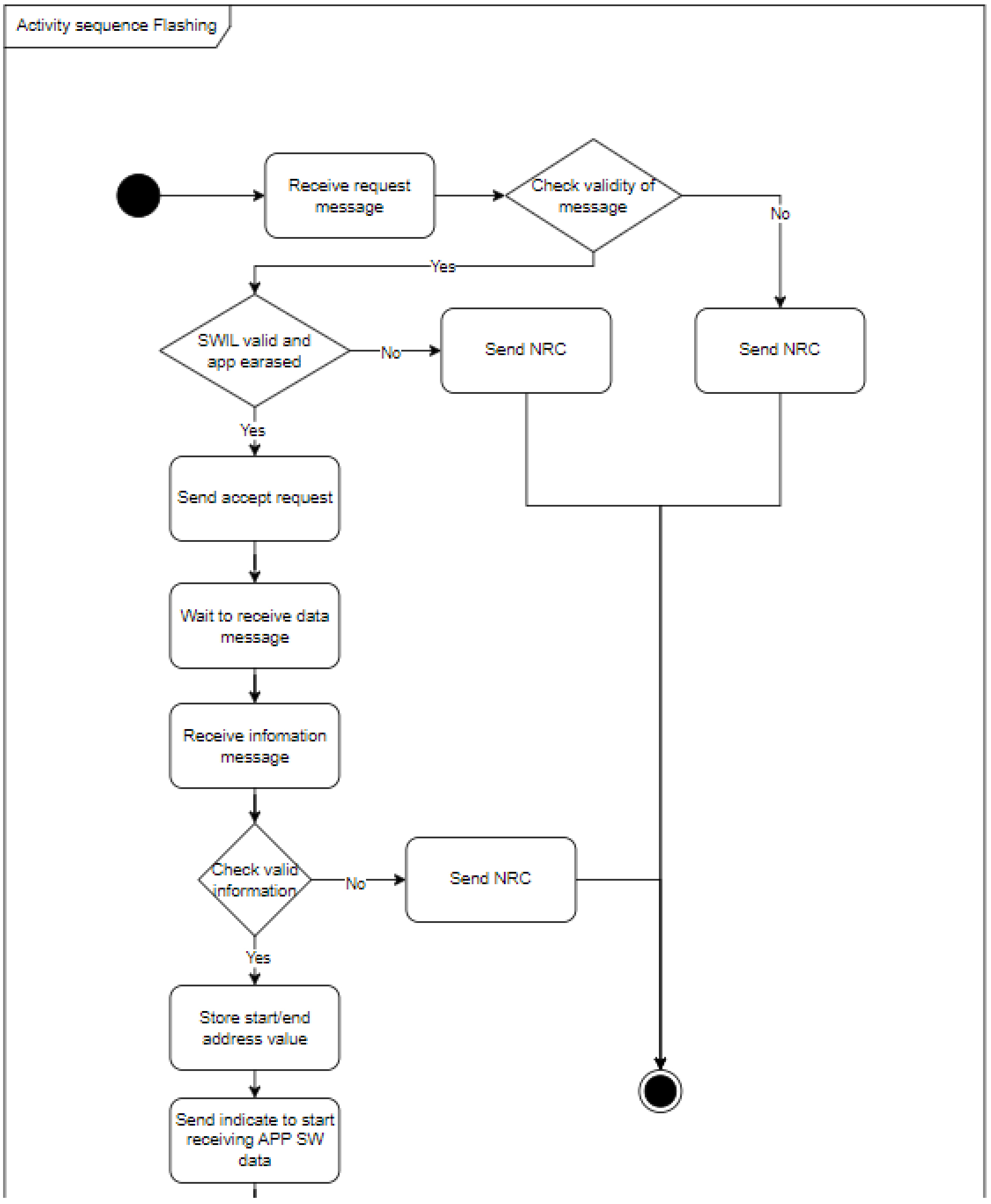
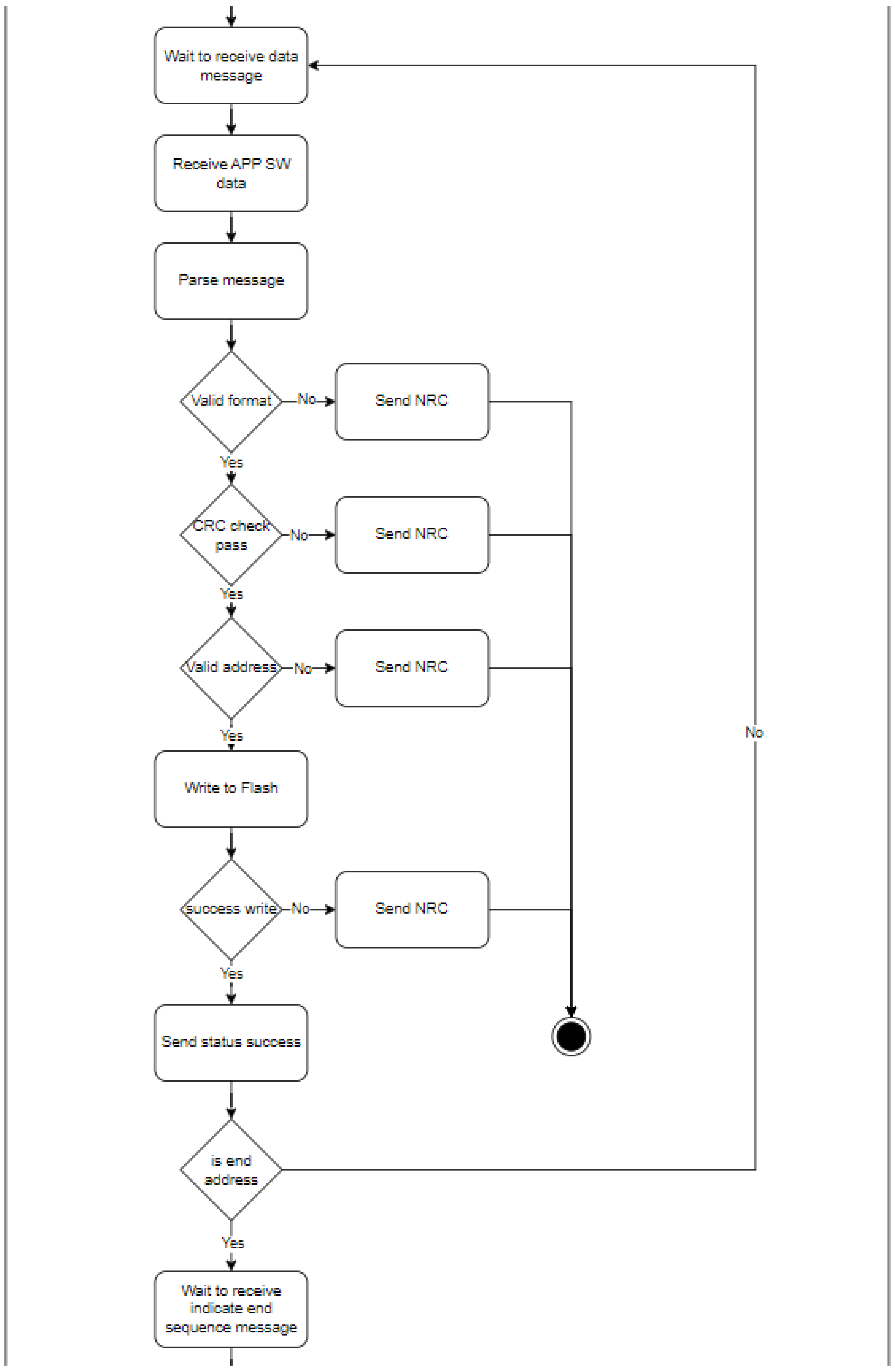


Figure 26. Loop sequence

6.4.3 Flash handle activity diagram


Activity sequence Flashing
Receive request message
Check validity of message
No
Yes
Send NRC
SWIL valid and app earased
No
Send NRC
Yes
Send accept request
Wait to receive data message
Receive infomation message
Check valid information
No
Send NRC
Yes
Store start/end address value
Send indicate to start receiving APP SW data


Wait to receive data message
Receive APP SW data
Parse message
Valid format
No
Send NRC
Yes
CRC check pass
No
Send NRC
Yes
Valid address
No
Send NRC
Yes
Write to Flash
success write
No
Send NRC
Yes
Send status success
is end address
No
Yes
Wait to receive indicate end sequence message

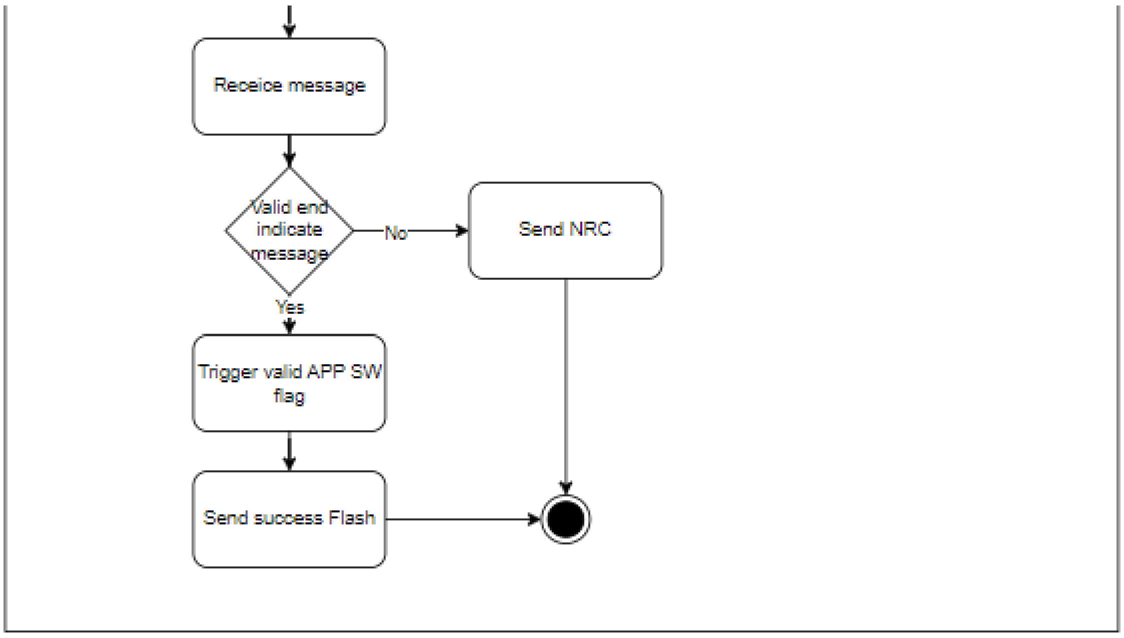


Figure 27. Flash handle sequence

7. Deployment view

7.1 Static view

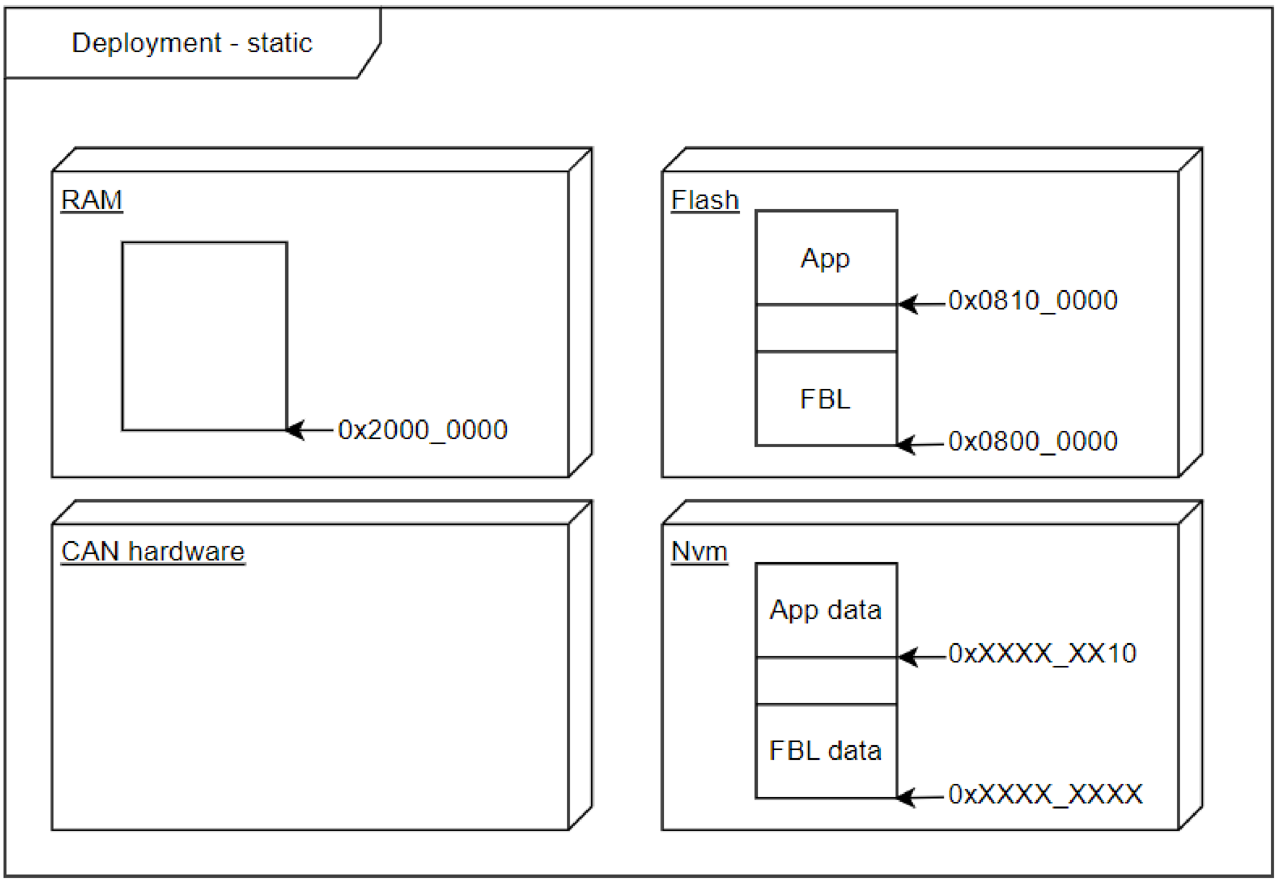


Figure 28. Static view

7.2 Dynamic view

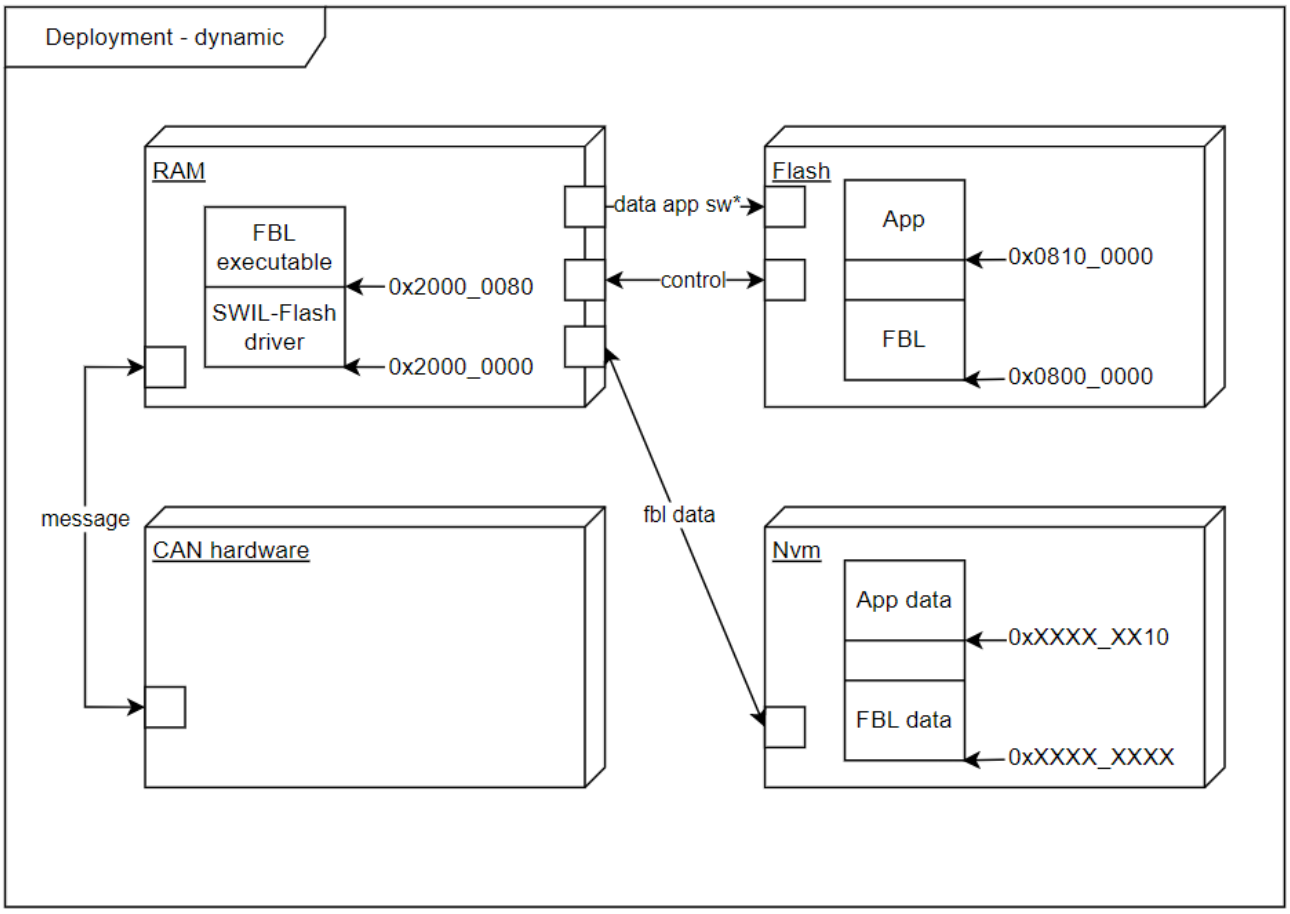


Figure 29. Dynamic view

8. Crosscutting concepts

8.1 Diag message

- Motivation:
  - Each service need to communication with Tool to send and receive status in service process
  - These message need follow one specific format (diagnostic communicaition)
- Solution:
  - Diag com module will define standard format for message before send/receive to CAN driver
  - Diag com will be included into all service module to perform transmit formal data

8.2 Nvm interaction

- Motivation:
  - Some service need to check/write Nvm data to perform next step
- Solution:

- Nvm handler module will provide API for other module to read and write data. Other module will not access to NVM driver directly. Handling Nvm driver will be handle by Nvm handler
- Nvm handler will be included into all service module to perform process Nvm data

9. Architecture decisions

Architecture decisions	Link
Software structure design	ADR Software structrure

10. Quality requirements

10.1 Quality Requirements

10.2 Quality Scenarios

11. Risks and Technical Debt

Risk	Desdription	Priority	Impact
Hardware without Nvm	SW can not store validity of APP SW	1	FBL can accept/auto jump to APP althougth APP SW invalid

12. Glossary

Term	Definition
DDD	Design decision document
SDD	Solution design document
SWIL	Software interlock
CAN	controller area network
Nvm	Non-volotile memory

13. Appendix

This is a link to the Asciidoctor Documentation.
Arc42 template reference Arc42 Documentation.

Contact information

Email (our adress is spam-protected) LinkedIn Github issue tracker