
# Software Requirement Specification Version 3.20

Software Requirement Specification for Evsent and the Simple Event Sender

Authors: Josefin Bladh, Frida Fasth, Anders Jåfs, Felix Kimiaei, Ingrid Wendin

Tutor, Lena Buffoni

Examinator, Kristian Sandahl

# Table of contents

# Table of Versioning

| Version | Authors | Reviewed by | Changes made | Date |
|---|---|---|---|---|
| 1.1 | Ingrid Wendin | Analysts | FR2, FR5 - Will is changed to can in the line ”The client will provide a unique” | 2019-10-07 |
| 1.2 | Ingrid Wendin | Analysts | FR3 - Title: changed word "database" to "storage". Description: remove "a database of" and substitute with "access to" | 2019-10-07 |
| 1.3 | Ingrid Wendin | Analysts | FR1 - Details on the three optional input parameters was added | 2019-10-07 |
| 1.4 | Ingrid Wendin | Analysts | FR17 - Added information on SendToMessageBus | 2019-10-07 |
| 1.5 | Ingrid Wendin | Analysts | FR18 - Added "This response shall include the meta.id of the sent Eiffel Event" | 2019-10-07 |
| 1.6 | Ingrid Wendin | Analysts | FR2 - Title: changed title to "Unique UUID", Description: Content is changed to describe UUID | 2019-10-07 |
| 1.7 | Ingrid Wendin | Analysts | DC2 - Added a line that authentication session handeling does not have to be stateless | 2019-10-07 |
| 1.8 | Ingrid Wendin | Analysts | FR12 - Moved to position of FR2 | 2019-10-07 |
| 1.9 | Ingrid Wendin | Analysts | FR13 - Moved to position of FR3 | 2019-10-07 |
| 1.10 | Ingrid Wendin | Analysts | FR2 - Content moved to v.1 FR13, then deleted | 2019-10-07 |
| 1.11 | Ingrid Wendin | Analysts | FR14 - Moved to position of FR4 Description: changed meta.version to edition | 2019-10-07 |
| 2.1 | Josefin Bladh | Analysts | FR1 - Broke it up into four requirements: FR1, and the new FR19, FR20, FR21 | 2019-10-07 |
| 2.2 | Josefin Bladh | Analysts | Header 3.2.1 - Changed to Input parameters | 2019-10-07 |
| 2.3 | Josefin Bladh | Analysts | FR14 - Changed title to Meta.type & Edition | 2019-10-07 |
| 2.4 | Josefin Bladh | Analysts | Header 3.4.1 - Added header "3.4.1 Deployment" and changed old header 3.4.1 to 3.4.1.1 and header 3.4.2 and 3.4.1.2 | 2019-10-07 |

| | | | | |
|---|---|---|---|---|
| 2.5 | Josefin Bladh | Analysts | Header 3.3.1.1 - Added header "3.3.1.1 Events per seccond" | 2019-10-07 |
| 2.6 | Ingrid Wendin | Analysts | FR24 - Added FR24 describing the meta.types that shall be handled by the event sender | 2019-10-10 |
| 2.7 | Ingrid Wendin | Analysts | FR23 - Added FR25 describing the meta.time | 2019-10-10 |
| 2.8 | Josefin Bladh | Analysts | Header 3.2.1 - Changed to title " 3.2.1 JSON Object HTTP body" | 2019-10-10 |
| 2.9 | Josefin Bladh | Analysts | Header 3.2.2 - Changed to title " 3.2.2 JSON Parameter Object" | 2019-10-10 |
| 2.10 | Josefin Bladh | Analysts | FR1 - Changed description | 2019-10-10 |
| 2.11 | Josefin Bladh | Analysts | FR20 - Added statement about when Error message is needed | 2019-10-10 |
| 2.12 | Josefin Bladh | Analysts | FR21 - Added statement about when Error message is needed | 2019-10-10 |
| 2.13 | Josefin Bladh | Analysts | FR22 - Removed this requirement and added information to FR21 and FR20 as it concered those requirements. | 2019-10-10 |
| 2.14 | Ingrid Wendin | Analysts | QR2 - The HTTPS communication will not be included in the solution but happen through an external source | 2019-10-10 |
| 2.15 | Ingrid Wendin | Anders Jåfs | Scope - the same change as above: HTTPS will not be provided by the product but will be compatible through a solution | 2019-10-14 |
| 2.16 | Josefin Bladh | Anders Jåfs | FR25 - Specifying information on cliets responsibility, split from FR10 | 2019-10-15 |
| 2.17 | Josefin Bladh | Anders Jåfs | FR26 - Specifying information on cliets responsibility, split from FR10 | 2019-10-15 |
| 2.18 | Josefin Bladh | Anders Jåfs | FR12 - Changed meta to JSON data object | 2019-10-15 |
| 2.19 | Josefin Bladh | Anders Jåfs | FR10 - Crossed out part and put it into FR25 | 2019-10-15 |
| 2.20 | Josefin Bladh | Anders Jåfs | FR6 - Changed wording so that it's clear that the event database is not part of our solution. An Event database shall be able to connect to the API | 2019-10-15 |
| 2.21 | Anders Jåfs | Frida Fasth | Header 3.2.2.3 - "AllowMissingMatches" changed to "AllowMissingMatch" | 2019-10-15 |

| | | | | |
|---|---|---|---|---|
| 2.22 | Ingrid Wendin | Anders Jåfs | FR24 - Moved content on the links from FR24 to the new FR27, FR28 and FR29, made clear that lookups are to find ArtC, SCC & SCS | 2019-10-24 |
| 2.23 | Ingrid Wendin | Anders Jåfs | FR25 - Moved contet to the new FR30, explain that the client shall specify which event types that the links will link to, FR30: client will give link type | 2019-10-24 |
| 2.24 | Ingrid Wendin | Anders Jåfs | FR26 - Specified that the req. is also for several events | 2019-10-24 |
| 2.25 | Ingrid Wendin | Anders Jåfs | FR14 - Moved content on Eiffel edition to new FR31 | 2019-10-24 |
| 2.26 | Josefin Bladh | Anders Jåfs | FR23 - Specified that client can set time stamp but this will be replaced if the client does. | 2019-10-24 |
| 2.27 | Josefin Bladh | Anders Jåfs | FR13 - Removed Error statement and added to 2.5 Assumptions and dependencies that we assume UUID to be unique if provided | 2019-10-24 |
| 2.28 | Anders Jåfs | Ingrid Wendin | FR4 - Changed the requirement so it says that the latest meta.version should be found, based on the edition given by the client. | 2019-10-24 |
| 2.29 | Ingrid Wendin | Anders Jåfs | QR2 - Moved content on safe communication to the new QR3 | 2019-10-24 |
| 2.30 | Ingrid Wendin | Anders Jåfs | QR3 - New Req. Moved content on safe communication from QR2 and specified that the communication will be able to happen over revese proxy | 2019-10-24 |
| 2.31 | Josefin Bladh | Anders Jåfs | Header 3.2.4.1 - Changed word "protocol" to "edition" | 2019-10-24 |
| 2.32 | Josefin Bladh | Anders Jåfs | Header 3.2.4.2 - Changed word "protocol" to "version" | 2019-10-24 |
| 2.33 | Josefin Bladh | Anders Jåfs | FR5 - Changed word "protocol" to "edition and version" | 2019-10-24 |
| 2.34 | Ingrid Wendin | Josefin Bladh | FR15 - Changed word "protocol" to "edition" | 2019-10-24 |
| 2.35 | Ingrid Wendin | Josefin Bladh | FR16 - Changed word "protocol" to "edition" | 2019-10-24 |
| 2.36 | Ingrid Wendin | Analysts | FR25 - Moved requirement to 2.5 assumptions | 2019-10-24 |

| | | | | |
|---|---|---|---|---|
| 2.37 | Ingrid Wendin | Analysts | FR30 - Moved requirement to 2.5 assumptions | 2019-10-24 |
| 2.38 | Ingrid Wendin | Analysts | FR26 - Moved requirement to 2.5 assumptions | 2019-10-24 |
| 2.39 | Anders Jåfs | Ingrid Wendin | FR31 - Moved the requirement to Assumptions | 2019-10-24 |
| 2.40 | Ingrid Wendin | Anders Jåfs | FR14 - Removed redundant requirement that was also in the wrong position | 2019-10-24 |
| 2.41 | Ingrid Wendin | Anders jåfs | 2.5 Assumptions - Added a point specifying that since the UUID generated by the event sender is unique, that there is no need to verify its uniqueness | 2019-10-24 |
| 2.42 | Ingrid Wendin | Anders Jåfs | 2.2 Product functions - Changed from protocol to edition and specified that the event sender will use that information to identify the correct version, also removed database | 2019-10-24 |
| 2.43 | Ingrid Wendin | Analysts | FR17 - Changed "data" to "file" in JSON data | 2019-10-24 |
| 3.1 | Josefin Bladh | Anders Jåfs | FR18 - Changed description to clarify that we only need to send response with meta.id if event is sucessfully sent | 2019-11-05 |
| 3.2 | Josefin Bladh | Anders Jåfs | FR19 - Added statement that full Event should be sent as respons if SendToMessageBus is false | 2019-11-05 |
| 3.3 | Josefin Bladh | Anders Jåfs | Header 3.2.4 - Added a new header "3.2.4 Look-up " | 2019-11-08 |
| 3.4 | Josefin Bladh | Canceld | FR30 - Added new requirement to clarify when look up should be performed | 2019-11-14 |
| 3.5 | Frida Fasth | Anders Jåfs | Header 2.2.1.3 - Added sequence diagram for AllowMissingMatch | 2019-11-08 |
| 3.6 | Anders Jåfs | Ingrid Wendin | Header 2.2 - Added sequence diagrams for parameters "SendToMessageBus" and "AllowSeveralMatches" as well as added new Headlines | 2019-11-11 |
| 3.7 | Anders Jåfs | Ingrid Wendin | FR31 - Added requirement about what edition is prioritized | 2019-11-11 |

| | | | | |
|---|---|---|---|---|
| 3.8 | Anders Jåfs | Ingrid Wendin | FR32 - Added requirement about what edition is prioritized | 2019-11-11 |
| 3.9 | Anders Jåfs | Ingrid Wendin | FR33 - Added requirement about what edition is prioritized | 2019-11-11 |
| 3.10 | Anders Jåfs | Ingrid Wendin | FR34 - Added requirement about what edition is prioritized | 2019-11-11 |
| 3.11 | Ingrid Wendin | Anders Jåfs | FR3 - Removed, since atomic in FR31-FR34 | 2019-11-12 |
| 3.12 | Anders Jåfs | Ingrid Wendin | FR8 - Removed, since redundant. | 2019-11-14 |
| 3.12 | Anders Jåfs | Ingrid Wendin | FR9 - Removed, since redundant. | 2019-11-14 |
| 3.12 | Anders Jåfs | Frida Fasth | FR10 - Removed, since redundant. | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | 2.2.1.2 - Modified the sequence diagrams for ASM to include information on if link protocol data Multile allowed is yes or no | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | 2.2.1.3 - Modified the sequence diagrams for AMM to include information on if link is required = yes or no | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | FR1 - Specified that parameter object only can contain one parameter and one string, added information on the Look-up Object | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | Header 3.2.3 - Created header Look-up Object where specification on what the Look-up object will contain and structure is stated | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | FR20 - Moved to under the look-up object and changed to being link specific | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | FR21 - Moved to under the look-up object and changed to being link specific and only allowing missing non-required links | 2019-11-14 |
| 3.12 | Ingrid Wendin | Frida Fasth | FR19 - Added clarification on that the SendToMessageBus is event specific | 2019-11-14 |
| 3.12 | Josefin Bladh | Frida Fasth | Header 3.2.5 - Changed from "Look up' to 'Look up function' | 2019-11-14 |
| 3.12 | Josefin Bladh | Frida Fasth | FR36 - Added requirement about when look up should be performed according to decision from look up cross functional team. | 2019-11-14 |

| | | | | |
|---|---|---|---|---|
| 3.12 | Ingrid Wendin | Frida Fasth | FR35 - Added requirement on string input for edition | 2019-11-14 |
| 3.13 | Ingrid Wendin | Anders Jåfs | FR31 - Changed edition names so that edition-agen is the one before the latest (edition-agen-1) | 2019-11-15 |
| 3.13 | Ingrid Wendin | Anders Jåfs | FR32 - Changed edition names so that edition-agen is the one before the latest (edition-agen-1) | 2019-11-15 |
| 3.13 | Ingrid Wendin | Anders Jåfs | FR35 - Changed edition default to agen-1 | 2019-11-15 |
| 3.14 | Ingrid Wendin | Josefin Bladh | 2.2.1.1 - Also return whole event to client when STMB = true | 2019-11-18 |
| 3.14 | Ingrid Wendin | Josefin Bladh | FR19 - Removed if statement on what happens if STMB = true and added response for both true and false | 2019-11-18 |
| 3.14 | Andreas Behrendtz | Josefin Bladh | Headers - Edited headings to match graphical profile (from Calibri to Garamond) | 2019-11-18 |
| 3.17 | Ingrid Wendin | Josefin Bladh | 2.5 - Added detail that the standardized format can be found in the document End-user Documentation. | 2019-11-21 |
| 3.17 | Ingrid Wendin | Josefin Bladh | 2.5 - Changed the assumption to include all required filds, not only the meta.type | 2019-11-21 |
| 3.15 | Anders Jåfs | Frida Fasth | FR18 - Changed that it's the content of the Event, and not the meta.id that is sent to confirm that an Event has been created | 2019-11-19 |
| 3.17 | Ingrid Wendin | Josefin Bladh | 2.5 - Clarified the assumption on the editions published on Eiffel | 2019-11-21 |
| 3.16 | Ingrid Wendin | Anders Jåfs | FR35 - The string EDITION is now required by our solution | 2019-11-19 |
| 3.17 | Josefin Bladh | Ingrid Wendin | 2.2.2 - Added description of Look up function to the SRS | 2019-11-21 |
| 3.18 | Linus Johansson | Anders Jåfs | - Edited first page to Official document standard | 2019-11-30 |
| 3.19 | Frida Fasth | Josefin Bladh | 2.1 - Changed picture of high level architecture | 2019-12-06 |
| 3.20 | Frida Fasth | Josefin Bladh | Figures - Added figure text for all figures | 2019-12-06 |

# 1 Introduction

## 1.1 Purpose

The purpose of this document is to give a description of the requirements for "The simple event sender" software. It will state the software design, constraints on the system and interactions with external systems. This document is meant to be proposed to the customer for its approval and as a reference for the development team during the project.

## 1.2 Scope

"The Simple Event Sender" is a product, which will generate well-formed and complete messages of different events a specific user creates. The application will be able to be accessed by robot actors, such as for instance automated programs, as well as human users, who will access the product via a Frontend-Web application with a user interface. All users can be able to receive authorization to the application over HTTPS by using external HTTPS solution.

The product will also be able to fetch event data from the Eiffel Event Database in order to look up Eiffel links connected to the Event that is being generated. An Event will then be generated and a response in form of a JSON file will be sent over AMQP to the RabbitMQ.

## 1.3 Definitions, acronyms and abbreviations

In this document requirements will be assigned a unique ID. Each ID contains of a number combined with an abbreviation that refers to the type of requirement. Abbreviation used are:

- IR – Interface Requirement
- FR – Functional Requirement
- PR – Performance Requirement
- DC – Design Constraints
- QR – Quality Requirement

Each requirement is also assigned a priority level where level 1 means must be implemented, and level 2 means could be implemented in the product.

## 1.4 References

The Event sender works together with Ericsson's Eiffel protocol. More information on what the Eiffel protocol is and what the standards are can be found here: https://github.com/eiffel-community

## 1.5 Overview

The second chapter of this SRS handles the overall description regarding the product whereas the third chapter is to give the reader an understanding of the specific requirements the product shall uphold.

# 2 Overall description

This section will give a complete overview of the system. How the system communicates with external parties as well as basic functionality will be described.

## 2.1 Product perspective

Today, Ericsson use the Eiffel protocol event system to enable a stateless environment of continuous integration, making the artefact pipeline as efficient as possible. The key issue is that there is no current centralised way of validating the events, making sure they match the protocol standard.

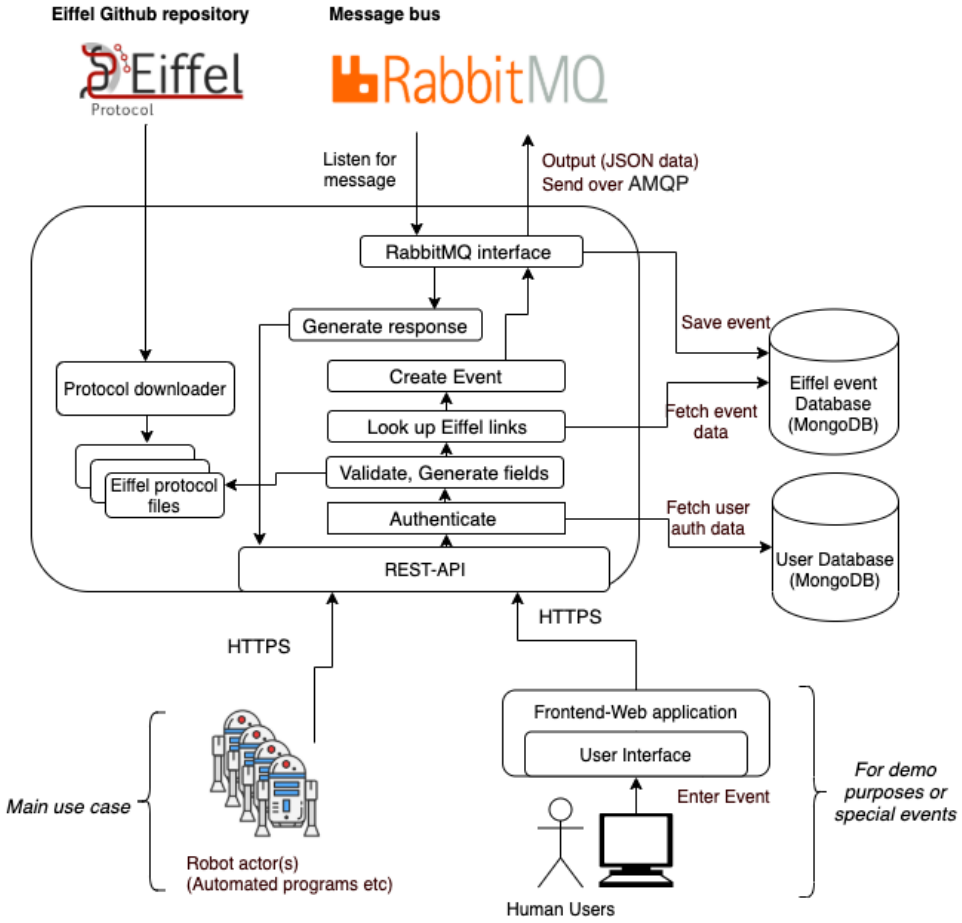


*Figure 1*. High-level architecture of the Simple event sender

As a result of the aforementioned problem, formatting and validation of any events sent into the message bus is the client's responsibility. This means that the client must also retrieve all missing information which it is not able to self-generate.

The purpose of the proposed system is taking this responsibility from the client, making sure that all events that enter the message bus are according to protocol standard. Enabling the use of an external actor to complete and validate all event protocols will remove a significant workload from the clients along with making sure that no incorrect event protocols enter the message bus. As a result, the documentation and continuous integration process will be more reliant, efficient and streamlined.

In a performance perspective, the current goal is for the system to be able to handle 10 events/second and have a great scalability. This is on order to eventually meet the performance demands of Ericsson's 2500000 daily Eiffel Events, as well as keep up with the latest versions of the protocol.

## 2.2 Product functions

The user of this system will create an Eiffel event. The user needs to input data and choose which edition the event should be written by. The system will collect the latest version of the event, specified in the given edition and identify if there is required data to create the event that the client has not provided. There will be lookup functions to that database with event and then collect the links that are missing. When the links have been collected, an Eiffel Event will be created and be validated. This to prevent invalid messages to be created. If a correct event has been made the event will be sent to a message bus and the client will get a message that a correct event has been made.
The simple event sender will also listen to the RabbitMQ message bus and update the Eiffel Event database with any Eiffel Events that are created and do not pass through the Simple Event Sender.

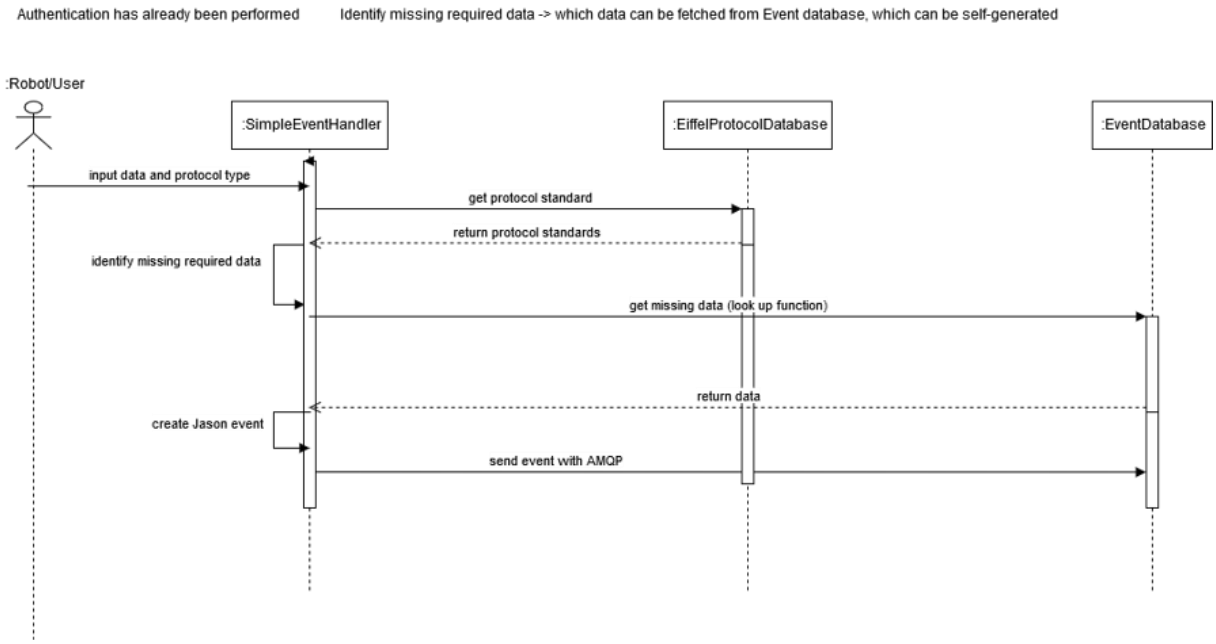


*Figure 2.* Sequence diagram for the Simple event sender

## 2.2.1 Parameter-Object functions

### 2.2.1.1 *SendToMessageBus*

SendToMessageBus= true

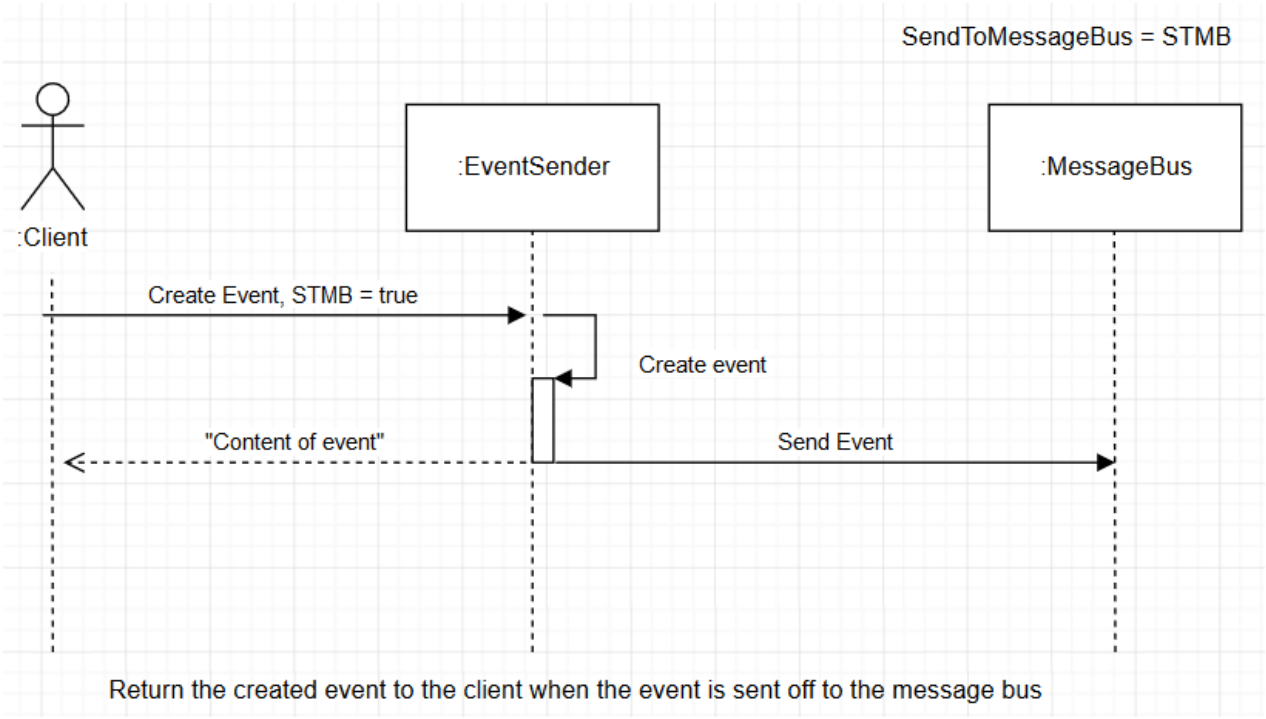


*Figure 3.* Sequence diagram for the parameter object SendToMessageBus (true)

SendToMessageBus= false

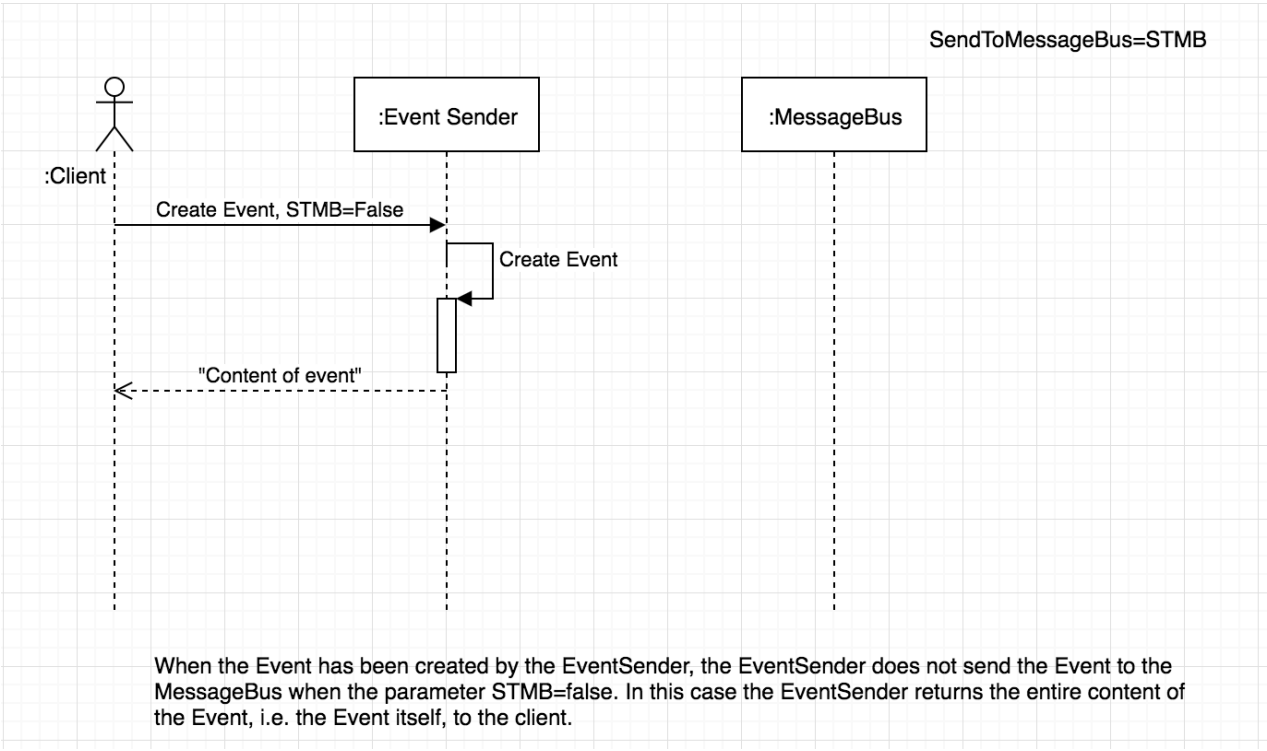


*Figure 4.* Sequence diagram for the parameter object SendToMessageBus (false)

#### 2.2.1.2 *AllowSeveralMatches*

Protocol standard for link: Multiple allowed = yes

Parameter AllowSeveralMatches = false

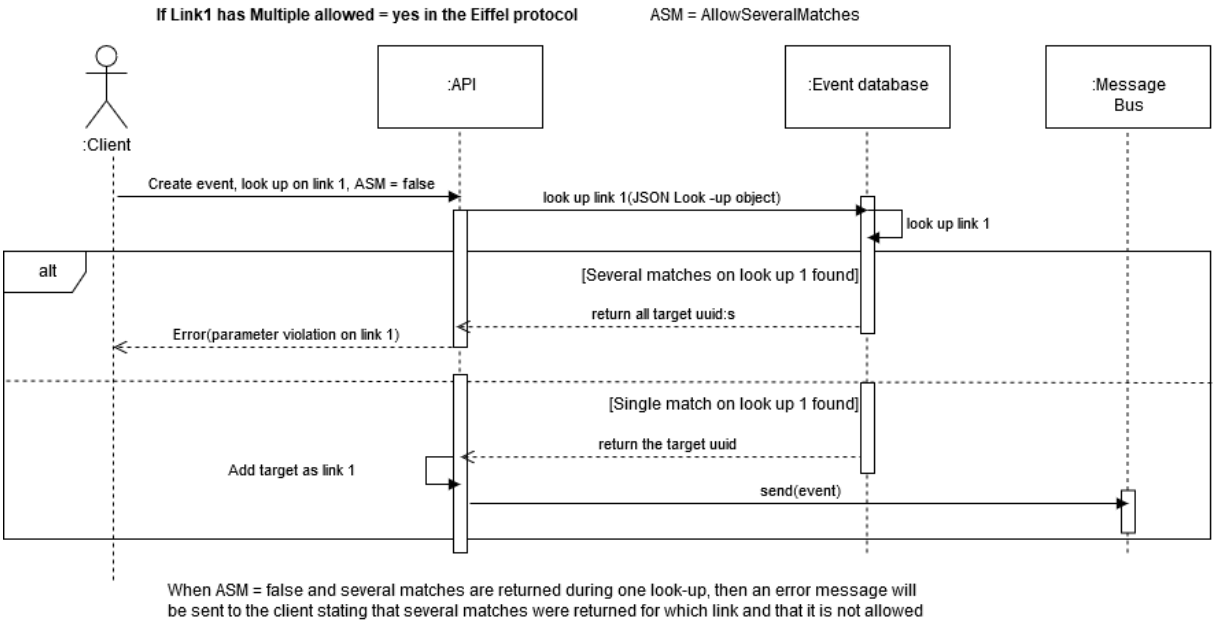


*Figure 5.* Sequence diagram for the parameter object AllowSeveralMatches (false)

Protocol standard for link: Multiple allowed = yes

Parameter AllowSeveralMatches = true

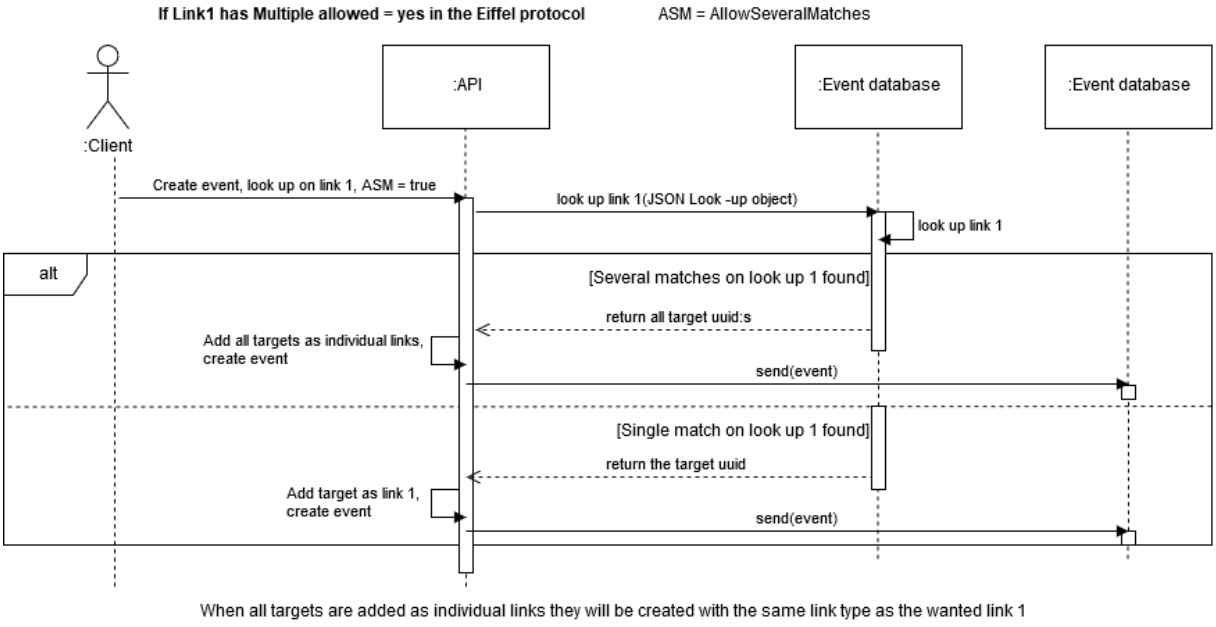


*Figure 6*. Sequence diagram for the parameter object AllowSeveralMatches (true)

Protocol standard for link: Multiple allowed = no

Parameter AllowSeveralMatches = false

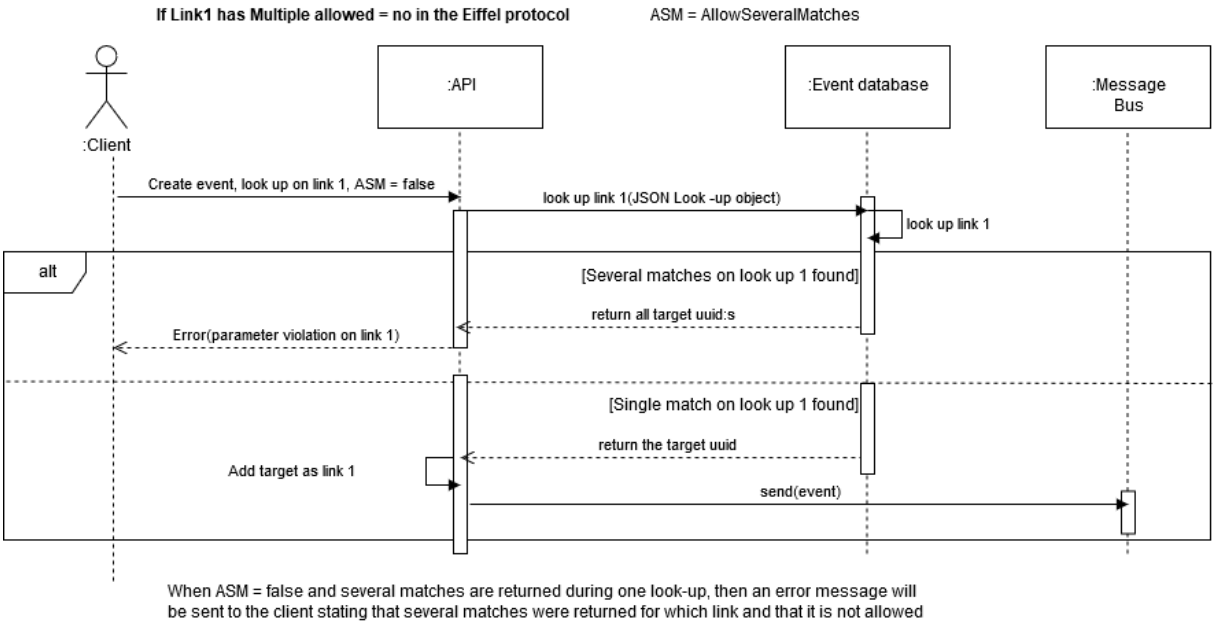


*Figure 7*. Sequence diagram for the parameter object AllowSeveralMatches (false)

Protocol standard for link: Multiple allowed = no

Parameter AllowSeveralMatches = true

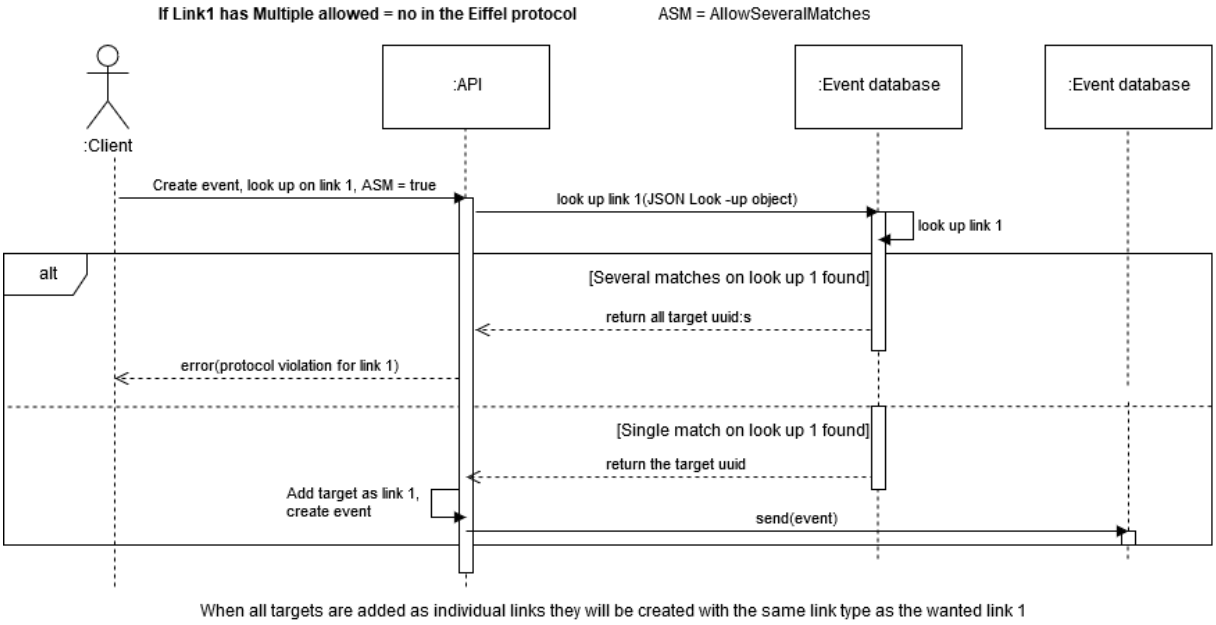


*Figure 8*. Sequence diagram for the parameter object AllowSeveralMatches (true)

#### 2.2.1.3 *AllowMissingMatch*

If AllowMissingMatch = false

Link is or is not Required (both will have the same response) but note the different error messages.

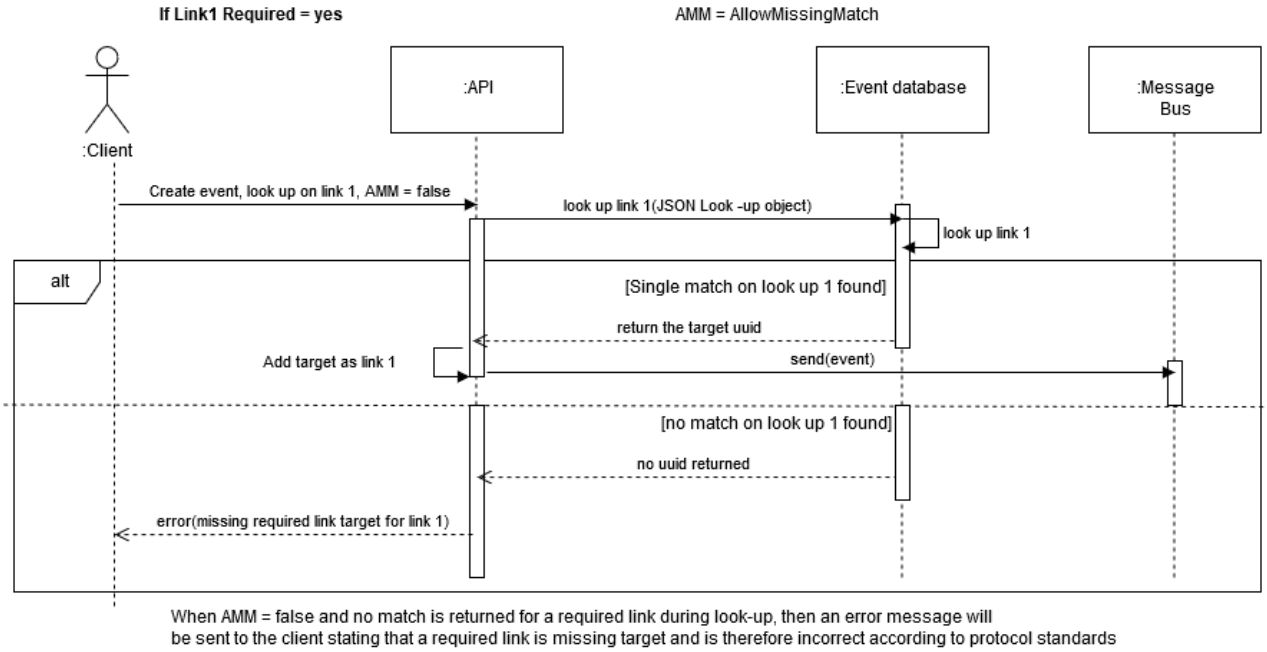


*Figure 9.* Sequence diagram for the parameter object AllowMissingMatch (false)

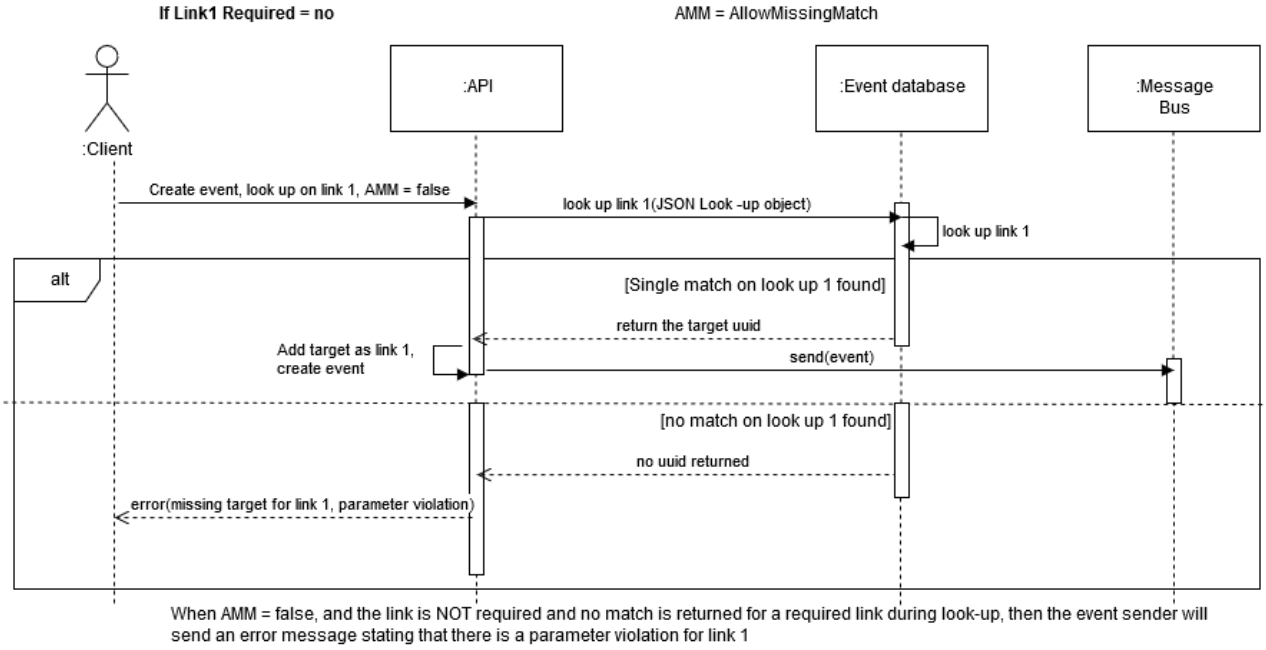


*Figure 10.* Sequence diagram for the parameter object AllowMissingMatch (false)

If AllowMissingMatch = true

Link is Required

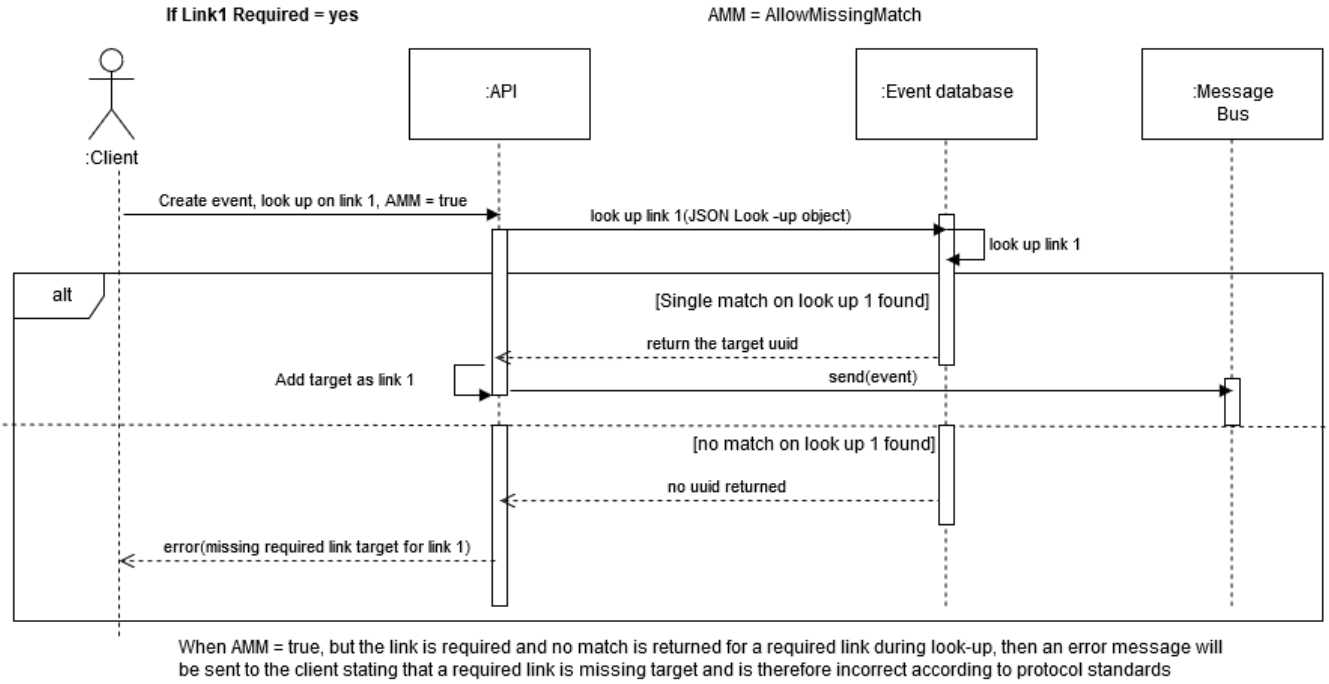


*Figure 11.* Sequence diagram for the parameter object AllowMissingMatch (true)

If AllowMissingMatch = true

Link is NOT Required

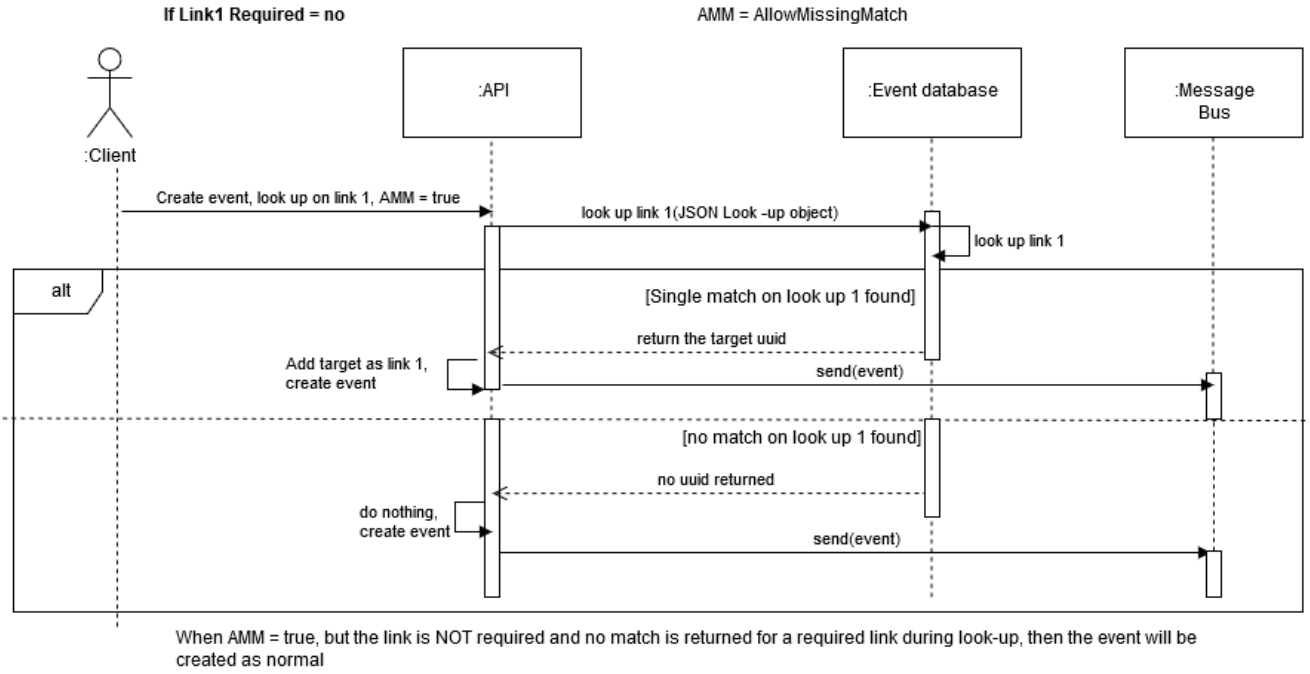


*Figure 12.* Sequence diagram for the parameter object AllowMissingMatch (true)

### 2.2.2 Look up function

If client wants to link to another Eiffel event but does not know the link target, the client shall provide a separate look up object. The link target is the UUID of the linking event. This object shall contain all information needed for the Event Sender to perform a query in the event database and find the wanted link target.

The look up object shall for each link contain link type and can contain the optional parameters AllowSeveralMatches and AllowMissingMatch. If the parameters are not provided the default values will be used. Data used for query shall also be provided from the client. In the example below data.identity is given for the first look up and the two optional parameters AllowSeveralMatches and AllowMissingMatch are given. For the second look up the query data is data.gitCommitID and no parameters are given. The query can consist of one to several data inputs.

```
    links: [
        { linkType: "SUBJECT",
          allowSeveralMatches: true,
          allowMissingMatch: true,
          query: {
              "data.identity" :"gitID:250203"
          }
        },
        {
            linkType: "CAUSE",
            query: {
                "data.gitCommitID" : "1250812587125"
            }
        }

    ]
}
```

*Figure 13.* Example of the Look up function

## 2.3 User characteristics

There are two kinds of actors that can use this system. The first one is an Ericsson staff and the second is an automatic robot running tests in the Eiffel event message bus. The Ericsson staff member will access the Simple event sender through a graphical interface. After logging in an event can be created by filling in the required data from a keyboard and pressing the create-button.

The main accesses will happen with robots and they will connect directly to the API. The robots will go through the same authentication process as the Ericsson user before it can provide the API with data to create an Eiffel event.

## 2.4 General constraints

The product is constrained by the time it may take to fetch the authorized user data from the user database as well as the time it takes to look up mentioned event data from the Eiffel event database and create links. The messages that are being created are constrained in the way that they must follow Eiffel standard. The outputs must be in JSON files and sent over the AMQP.

## 2.5 Assumptions and dependencies

This Software Requirement Specification is made under some assumptions as follows:

- Although client's input will vary, all clients will provide data in a standardized format described by the document End-user Documentation.
- To create a complete event, the client will provide all required data that cannot be automatically generated by the Event sender, either direct or, in the case of incomplete links, by providing information that can be used to identify link targets.
- No changes will be made to the editions already published on GitHub, thus enabling the Event Sender to be up to date with the current published editions.
- The Event Sender will not have any responsibility for the distribution of Eiffel Events further than getting the Event in the message bus.
- There will be a stable connection between the client and the Event Sender
- Most requests will be from non-human clients.
- There will only be two kind of actors, the robot actor and the Ericsson Staff Member.
- If the client provides a UUID to the meta.id-field this will be unique.
- The UUID that is created by the event sender will be unique, hence its uniqueness does no need to be verified against the content of the event database.
- The client is responsible for specifying what link types it wants to create.
- The client is responsible for specifying what event type it want each requested link target to be.

- For each link target, if the meta.id is not provided by the client, then the client is responsible for providing information that can be used by the API to find the meta.id. This information shall be given to the API in the JSON look up object.
- The client shall provide the Eiffel edition that the event shall be created according to.

## 2.6 Lower ambition levels

If the project gets delayed, then the frontend interface towards Ericsson staff will be left to implement in the next possible iteration.

# 3 Specific requirements

## 3.1 Interface requirements

### 3.1.1 User interfaces

#### 3.1.1.1 *Human user interface*

ID: IR1

Description: The product will have a user interface for human users.

Priority level: 2

### 3.1.2 Software interfaces

#### 3.1.2.1 *Standardized, well documented interface*

ID: IR2

Description: The product shall provide a standardized, well-documented interface to its users.

Priority level: 2

#### 3.1.2.2 *Open source license*

ID: IR3

Description: The product will be licensed under Apache License 2.0.

Priority level: 1

#### 3.1.2.3 *Rest API*

ID: IR4

Description: The product will have a Rest API.

Priority level: 1

## 3.2 Functional requirements

### 3.2.1 JSON Object HTTP body

#### 3.2.1.1 *Receive requests*

ID: FR1

Description: The product shall be able to receive requests from clients containing a JSON object containing data to the event, The JSON data Object; an optional JSON

object with one optional parameter and one optional string, the JSON parameter Object; and one optional JSON object containing look up requests, the JSON Look-up Object.

Priority: Level 1

### 3.2.2 JSON Parameter Object

#### 3.2.2.1 *SendToMessageBus*

ID: FR19

Description: The SendToMessageBus parameter controls if the created event is sent to the message bus and stored in the database or not. It has default true. It is event specific. Upon generate event success, the API shall respond to the client with the created event.

Priority: Level 1

#### 3.2.2.2 *Edition String*

ID: FR35

Description: The string EDITION controls what edition the event will be created in. The String is required.

Priority: Level 1

### 3.2.3 JSON Look-up Object

#### 3.2.3.1 *AllowSeveralMatches*

ID: FR20

Description: The AllowSeveralMatches parameter controls if several matches in a look up for one link shall be accepted and all added to the created event as links. It has default false. It is link specific.

- If the look up is not accepted an Error message will be created.

Priority: Level 1

#### 3.2.3.2 *AllowMissingMatch*

ID: FR21

Description: The AllowMissingMatch parameter controls if a missing match during look up for one non-required link shall be accepted or not. It has default false. It is link specific.

- If the missing match is not accepted an Error message will be created.

Priority level: 1

### 3.2.4 JSON data object

#### 3.2.4.1 *JSON data object inputs*

ID: FR12

Description: The client will be able to provide all parameters in the JSON data object apart from meta.time, which shall be appointed by the Simple Event Sender.

Priority level: 1

#### 3.2.4.2 *Meta.id*

ID: FR13

Description: If the client does not provide a UUID for the meta.id-field, the Simple Event Sender shall generate a unique UUID for the event.

Priority level: 1

#### 3.2.4.3 *Meta.time*

ID: FR23

Description: The event sender shall always generate the time stamp for meta.time.

Priority level: 1

#### 3.2.4.4 *Handle Meta.type*

ID: FR24

Description: The event sender shall be able to create the following event types.

1. CLM
2. SCS
3. CD

Priority level: 1

### 3.2.5 Look-up function

#### 3.2.5.1 *Perform look-up*

ID: FR36

Description: The event sender will perform a look-up when request is given in the look-up object. The look-up will return a correct link to the data object with link.type and UUID as link.target.

Priority level: 1

#### 3.2.5.2 *Look-up for CLM*

ID: FR27

Description: The event sender shall be able to perform a look-up for CLM events to link to ArtC events.

Priority level: 1

#### 3.2.5.3 *Look-up for SCS*

ID: FR28

Description: The event sender shall be able to perform a look-up for SCS events to link to SCC events.

Priority level: 1.

#### 3.2.5.4 *Look-up for CD*

ID: FR29

Description: The event sender shall be able to perform a look-up for CD events to link to ArtC events and to SCS events.

Priority level: 1.

### 3.2.6 Handle multiple Eiffel protocols

#### 3.2.6.1 *Handle latest Eiffel edition*

ID: FR31

Description: The product shall be able to create events according to the edition-agen-1 edition.

Priority 1

#### 3.2.6.2 *Handle edition-agen*

ID: FR32

Description: The product shall be able to create events according to the agen edition.

Priority: 2

#### 3.2.6.3 *Handle edition-toulouse*

ID: FR33

Description: The product shall be able to create events according to edition-toulouse edition.

Priority: 2

#### 3.2.6.4 *Handle edition-bordeaux*

ID: FR34

Description: The product shall be able to create events according to the edition-bordeaux edition.

Priority: 2

#### 3.2.6.5 *Use correct version*

ID: FR4

Description: The product will for every event have a way to match and retrieve the latest meta.version based on the edition name received from the client.

- If the correct Eiffel Event cannot be found (either by faulty connection to database or simply failing to match) the product will output a detailed error message describing what went wrong.

Priority level: 1

#### 3.2.6.6 *Validate in-data*

ID: FR5

Description: The product will validate the input-data from the client according to the requested Eiffel edition and meta.version.

- If there is a mismatch in types, formats etc, the product will notify the client that there is a problem.

Priority level: 1

### 3.2.7 Gather data

#### 3.2.7.1 *Eiffel Event database*

ID: FR6

Description: The product will be able to connect to an Eiffel Event database.

Priority level: 1

#### 3.2.7.2 *RabbitMQ interface*

ID: FR7

Description: The product shall contain a RabbitMQ interface that contains two functions. The first sends out messages to the RabbitMQ message bus. The second listens to the message bus and update the Eiffel Event database with new events from it.

Priority level: 1

#### 3.2.7.3 *Retrieve meta.id from Eiffel event database*

ID: FR11

Description: To produce a link, the product will have a way to retrieve the meta.id from the correct Eiffel Event in the Eiffel Event database.

Priority level: 1

### 3.2.8 Create Eiffel Event message

#### 3.2.8.1 *Compile into JSON file*

ID: FR15

Description: The product shall be able to compile all necessary information into a correctly formatted JSON file, according to the previously selected Eiffel edition, creating a correct Eiffel Event Message.

Priority level: 1

#### 3.2.8.2 *Validate complete JSON file*

ID: FR16

Description: The product shall validate the complete JSON file with the selected Eiffel edition before outputting the Eiffel Event.

- If the JSON file is invalid (missing non-optional fields etc), there will be an error message output describing what fields are problematic and the file will not be output.

Priority level: 1

#### 3.2.8.3 *Output data*

ID: FR17

Description: The product shall output the JSON file over AMQP sent onto RabbitMQ after validation.

- If a correct output couldn't be created, the product shall generate a detailed Error message, describing what, where and how something got wrong.
- If the client has inputted SendToMessageBus = false into the API, then the Eiffel Event message will not be outputted into the message bus nor will it be directly inserted into the Event database.

Priority level: 1

#### 3.2.8.4 *Generate response*

ID: FR18

Description: When the event has successfully been created the content of the Event shall be sent to the client confirming that the event has been created.

Priority level: 1

## 3.3 Performance requirements

### 3.3.1 Performance

#### 3.3.1.1 *Events per second*

ID: PR1

Description: The product will be able to create ten events every second.

Priority level: 1

## 3.4 Design constraints

### 3.4.1 Deployment

#### 3.4.1.1 *Deploy in cloud*

ID: DC1

Description: The product will be deployed in cloud environments using Docker.

Priority level: 1

#### 3.4.1.2 *Stateless*

ID: DC2

Description: Apart from authentication session handling, if implemented, the product shall be stateless.

Priority level: 1

## 3.5 Software system attributes

### 3.5.1 Availability

#### 3.5.1.1 *Horizontally scalable*

ID: QR1

Description: The product shall be horizontally scalable using a cloud orchestrator.

Priority level: 1

### 3.5.2 Security – Basic authentication

#### 3.5.2.1 *Token-based authentication*

ID: QR2

Description: The product shall have a token-based authentication solution.

- If the client is unauthorized the product shall send an error message to the client, stating that the login combination is not valid.

Priority level: 1

#### 3.5.2.2 *Communication method*

ID: QR3

Description: The product will be able to communicate over HTTPS through a reverse proxy.

Priority: 1

# 4 Supporting information

## 4.1 Index

## 4.2 Appendix

Here is a link to the Eiffel Git Community where we have found details on how Eiffel works