Requirement definition

Access Control



Neutorstraße 13  
5020 Salzburg

AUSTRIA

Tel: +43 (662) 276198-11

Fax: +43 (662) 276198-98

Mail: [office@breanos.com](mailto:office@breanos.com)

|  |  |
| --- | --- |
| File | RD\_AccessControl\_V1.docx |
| Date | 25.04.2018 |

Table of contents

[1 Change history 4](#_Toc511334798)

[2 Requirement definition 5](#_Toc511334799)

[2.1 Introduction 5](#_Toc511334800)

[2.2 Scope 5](#_Toc511334801)

[2.2.1 General 5](#_Toc511334802)

[2.2.2 Permission Management 5](#_Toc511334803)

[2.2.3 Data Structure 6](#_Toc511334804)

[2.2.4 Exclusion from scope 6](#_Toc511334805)

[2.3 Action plan 6](#_Toc511334806)

[2.4 Effort estimation 6](#_Toc511334807)

[2.4.1 Annotations 6](#_Toc511334808)

[2.5 Risk assessment 7](#_Toc511334809)

[2.5.1 Risk short description 7](#_Toc511334810)

[2.5.1.1 Description 7](#_Toc511334811)

[2.5.1.2 Consequences 7](#_Toc511334812)

[2.5.1.3 Avoidance 7](#_Toc511334813)

[2.5.1.4 Probability 7](#_Toc511334814)

[2.6 Prerequisites 7](#_Toc511334815)

[2.7 Acceptance 7](#_Toc511334816)

# Change history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Description |
| 14.03.2018 | 1 | Hutterer | Initial version |
| 24.04.2018 |  | Hutterer | Finalize the initial version |

# Requirement definition

## Introduction

An access control system (AC) for the DAIPAN framework, have to be realized. This concerns the Service Fabric for Applications (SFA) (Shared Access Signatures – SAS), as well as the general communication (messages within the system have to be authenticated) and the GUI client access to the KPUs and modification of their configuration.

## Scope

### General

In the first step the technology SAS in combination with Service Bus (SB) and SFA have to be inspected closely to do the detail planning. Following questions has to be clarified:

* How can we ensure that corrupted messages (from a foreign KPU, Client …) does not affect the system? (SB-and SFA-Authentication)
* How will the authentication in the system be done?
* How to hand over user information from one application to another? Who is sending the message? And how can this be ensured?
* How can we create, delete and manage users and permissions in SAS(SB/SFA)?

The AC is a microservice running SFA. The personal assistant (ADAM) needs a reference to the running instance, to handle the access control, because every permission request (client, message from the blackboard) has to be checked by ADAM, to ensure that no unauthenticated request is handled.

### Permission Management

The permission management has to be fully flexible to fit the requirements, that on the one hand any number and any type of KPU can run in the system and on the other hand each type of KPU can have its own internal permission structure. Therefor an interface has to be defined, where each KPU can register itself, as well as its permissions. To handle this each KPU needs a unique name and several KPUs have to be grouped to functional units called Workers, this Workers can again be grouped to so called Worker Groups. So, the structure of the permissions corresponds to a tree structure.

Example:

* Worker Group: Plant1
  + Worker: LineSGM
    - KPU: OEE-Analysis
      * Permission A
        + SubPermission A-1

SubSubPermission A-1-1

* + - * + SubPermission A-2
      * Permission B

An example of a bit similar implementation can be found in *Breanos.Security*.

### Data Structure

This permission structure has to be assigned to the users within the system. So, on one side there are the users and on the other side are the registered permissions. For easier administration, it would be better to assign the permissions via association table to User Groups and assign one or more groups to a user.

A bit simpler structure can be found in *Breanos.Security* respectively on the SQL-Server (BRE-SQL01), where each user is assigned to one User Group (see Figure 1).

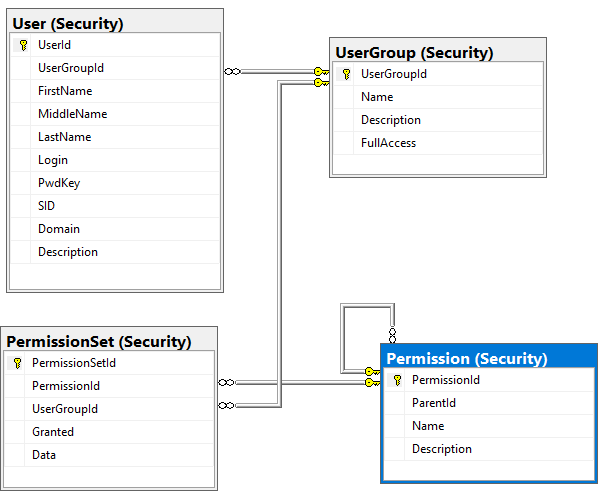


Figure 1

Database stored users and permissions are only one possibility, all of this can be stored in other structures like Active Directory or something else as well. Hench an interface for this data has to be defined to abstract the concrete data layer. For the first implementation a databased structure is adequate enough.

### Client

There are two different requirements for the client (based on Xamarin). First the client has to request a list of all services, which can be accessed by the current user, from the assistant. And visualize them, in a first step a simple visualization should be fine.

Secondary, master data views have to be designed to display and modify *Users* and *User Groups* and the relation to the assigned *Permissions*.

Overall a host application has to be designed which can host each type of visualization from any service. In a XML configuration each type of running KPU has to be assigned to a visualization template to be visualized in the client.

For further details please see the [requirement definitions for the GUI-client](../../UI%20Client/Requirements%20Definition).

## Exclusion from scope

User session and access to a KPU is part of ADAM.

GUI-Client is defined in another RD.

## Action plan

|  |  |
| --- | --- |
| Date | Milestone |
| 20.04.2018 | UML design finished |
| 27.04.2018 | Code implementation finished |
| 04.05.2018 | Testing finished |

## Effort estimation

The implementation effort is estimated as follows.

|  |  |
| --- | --- |
| Task | Hours |
| UML design | 75 |
| Implementation | 75 |
| Testing | 30 |
| Total | **180** |

### Annotations

## Risk assessment

### Technology limitations

#### Description

Due to the usage of MS .NET Core 2.0, there might be feature cuts of needed elements, available in MS .NET 4.6.2.

#### Consequences

Delays due to increased research efforts.

#### Avoidance

Detailed definition of the needed framework elements, previous research of the Core 2.0 implementation, effort calculation, pros and contras of using MS .NET Core 2.0.

#### Probability

Middle to high.

## Prerequisites

Running test environment (SFA cluster) and database server.

## Acceptance

* The service is hosted in SFA.
* Each public method is called either by the assistant or a test application.
  + Register a permission.
  + Request a permission of a user (granted/not granted).
  + User login (success/no success).
  + Modify and delete user-group-associations.
  + Modify and delete permission-group-associations.
  + A test protocol has to be made, including the input parameter as well as the database and output results.