BehavAuth

Human behaviour based authentication for Internet of Things

Open Call Project for TagItSmart: Grant agreement 688061

WP1: D1.2 Final system design based on feedback from early trials

Deliverable ID:	**D 1.2**
Deliverable Title:	**Final system design based on feedback from early trials**
Revision #:	**1.0.1**
Dissemination Level:	**Public**
Responsible beneficiary:	QBE
Contributing beneficiaries:	All
Contractual date of delivery:	28/02/2018
Actual submission date:	22/3/2018

Start Date of the Project: 1 September 2017 Duration: 6 Months

Contents

Section 1 - Refined use cases and requirements 7

1.1 Use cases 7
1.1.1 Digital product 7
1.1.2 Lifecycle management 8
1.1.3 Brand protection 8
1.1.4 Dynamic pricing 9
1.1.5 Home services 10

1.2 Requirement analysis 11

Section 2 - Refined system design 13

2.1 Overall system architecture 13

2.2 Mobile authentication app/library 14
2.2.1 Difference with UNIS location based authentication 14

2.3 TagItSmart polling manager app 15

2.4 Backend platform 15

Section 3 - Social interaction detection 16

3.1 Related Work 16
3.1.1 Wearable devices 16
3.1.2 Smartphones 17
3.1.3 Proposed approach 18

3.2 Proximity 18

3.3 System design 19
3.3.1 Overview 19
3.3.2 Training the detection models 20
3.3.3 Implementation 23

Section 4 - Mobile authentication app/library 24

4.1 Overall design 24

4.2 Data collection 25
4.2.1 Dataset 25
4.2.2 Pre-processing 25
4.2.3 Social interaction data 26
4.2.4 Location data 27
4.2.5 Speed data 28
4.2.6 Distance data 29

4.3 Behavioural cues extraction 30
4.3.1 Social interaction based authentication 30
4.3.2 Location-based authentication 31
4.3.3 Speed within the environment 32
4.3.4 Distance within the environment 32

4.4 Authentication inference 33

4.5 Implementation 33
4.5.1 Android permissions 33
4.5.2 Android Software Development Kit 34
4.5.3 Architecture 34
4.6 Authentication in simple words 35
Section 5 - TagItSmart polling manager 37
5.1 Mobile app usage 37
5.2 Android permissions 37
5.3 Android Software Development Kit 37
5.4 Architecture 37
5.4.1 Voting mechanism 38
5.4.2 Internal communication bus 38
5.4.3 Communication with TagItSmart platform 39
Section 6 - Design based on the functional requirements 41
Section 7 - Beyond TagItSmart - Future design 44
7.1 Overall design of the system 44
7.2 BehavAuth platform 45
7.3 BehavAuth mobile app/library 46
Section 8 - Conclusion 47
Section 9 - References 49

Document history

Revision	Date	Modification	Authors
1.0	1/03/2018	Draft ready for review	N. Palaghias
1.0.1	19/03/2018	Revision based on reviewers' comments	N. Palaghias

Introduction

The main objective of this document is to provide a detailed description of the final system design based on the use cases and the derived requirements that were extracted from the collaboration with the stakeholders (SMEs) that are interested in the BehavAuth and TagItSmart. Designing a system requires multiple inputs from the stakeholders such as psychologists, privacy experts, end-users and potential early adopters of the system. Each stakeholder will provide the appropriate input and lead to the definition of the various use cases and further to the derivation of the functional requirements of the system. Based on the functional requirements of the system, the initial design of the system was derived. Early trials and feedback from the various stakeholders enabled the refinement of the initial design and led to the final design of the BehavAuth system that is presented in this document.

Section 1 provides a description of the final use cases and the requirements defined after the early trials of the BehavAuth as well as the integration process with the TagItSmart platform. The requirements were initially defined in D1.1 [21] and are refined in this document. The section initiates with the description of the TagItSmart use cases, where the BehavAuth tool is applicable and could be utilised. Following, a requirement analysis that was redefined after the feedback from the integration process with the TagItSmart platform as well as the early trials of the tool.

Section 2 provides an overview of the BehavAuth ecosystem and the components that operate together in order to provide authentication service to the TagItSmart platform. The overview of the system architecture includes various components such as the authentication components including BehavAuth, the location based authenticator from University of Surrey and the keystroke authenticator from University of Padova. These authenticator components coordinate through the Polling Manager app that merges the authenticators results into a common authentication decision. This decision is forwarded to the TagItSmart platform and in particular to the User Repository.

Section 3 describes the social interaction detection mechanism that was developed to enable accurate detection of social interactions among people by leveraging the Bluetooth interface. An extensive data collection process was performed as well as an extensive feature extraction enabled the development of a machine-learning technique for identifying social interactions that are characterised by line-of-sight. The machine-learning model is able to run on commercial off-the-shelf mobile phones and does not induce any firmware restrictions.

In **Section 4**, the BehavAuth authentication app is analysed that leverages sensor data from Bluetooth and location interfaces in order to extract various behavioural cues and through these behavioural cues derive behavioural patterns that enable the authentication of the user. The behavioural patterns that enable the identification of a person are focused on the social interaction patterns, on location patterns, on speed and distance patterns within the environment. The machine-learning models were developed based on a 10-user dataset over the duration of a 10-week period. Lastly, the various authentication decisions are fused through

a novel voting based algorithm that concludes on the authentication result based on the accuracy of each of the authentication components.

Section 5, details the Polling Manager app that was developed in order to enable the integration of the authenticator modules that were developed through the TagItSmart and BehavAuth project. The authenticator components are including the location-based tool from the University of Surrey, the keystroke-based from the University of Padova and the BehavAuth tool. The result of the various authenticators is forwarded to the User Repository of the TagItSmart platform.

Given the use case and requirement analysis and the analysis of the authentication system, **Section 6** provides the future design of the BehavAuth system foreseen as an Authentication-as-a-Service solution. Initially, an overview of the overall design of the BehavAuth system is described with respect to third-party companies, including the workflow. Next, the BehavAuth platform is detailed, including an overall description of the main functionalities of the platform. Following, the BehavAuth mobile app/library that will perform the data collection is described including its functionalities.

Finally, this deliverable ends with the conclusion provided in **Section 7**.

Section 1 - Refined use cases and requirements

This section provides a detailed description of the concluded use cases defined by the TagItSmart and BehavAuth project, through the co-creation workshop and refined by the early trials.

1.1 Use cases

BehavAuth concluded on these use cases based on the engagement with stakeholders including the TagItSmart consortium. The images and information presented below have been taken from TagItSmart website [1] and deliverables.

1.1.1 Digital product

This use case is focused on using fast-moving consumer goods and combining them with the so-called smart-tags and the TagItSmart in order to track the product during the whole lifecycle. In this use case the product is being tracked from factory and the road to the end customer. The smart-tags are placed on the goods and are capable of tracking the environmental conditions in which the goods are exposed when they leave the factory and until they reach the end-customer. The customers are able to scan the smart-tag and identify if the goods have been exposed to certain conditions, for example above the maximum allowed temperature (See Figure 1).

Figure 1. Digital product use case.

BehavAuth contribution. BehavAuth will provide an authentication mechanism that will enable the end customer to authenticate through the TagItSmart mobile app to the TagItSmart platform. In this way, the user will be able to login to the TagItSmart mobile app and then scan the smart-tag that is placed on the products. BehavAuth will provide a secure and user-friendly authentication mechanism that will enable users to securely login to the mobile app and check the environmental conditions in which the product have been exposed during the journey from the factory to the shelf of the store.

1.1.2 Lifecycle management

This use case is focused on the managing the lifecycle of every fast-moving consumer goods or consumer packaged goods (CPG). The goal is to assist and empower companies and citizens in order to recycle their waste. TagItSmart will create a new ecosystem where the users prompted to recycle by providing to the customers information about how to recycle the product including where to recycle the product and the local regulations about the recycling process. Consumers will be able to scan the product and receive information about the recyclability of the product through the TagItSmart mobile app. A more detailed description of the process is described in the Figure 2 below.

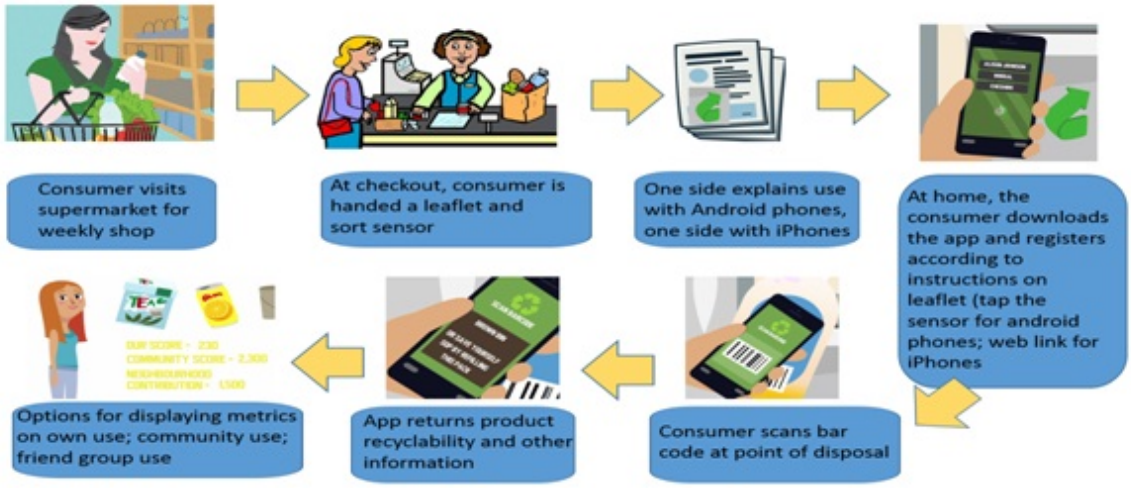
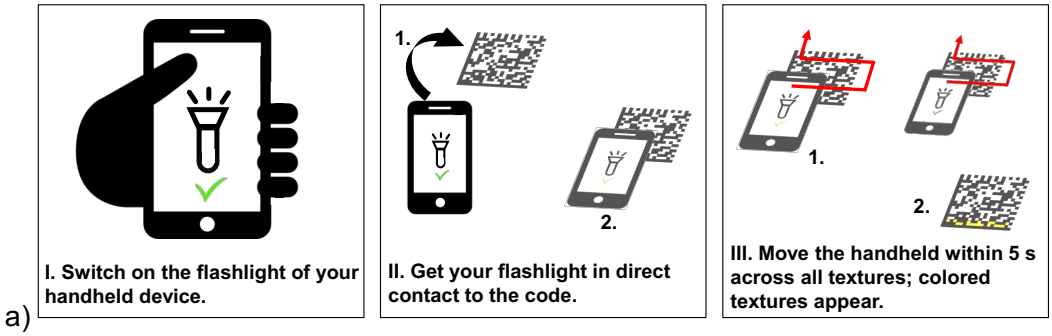


Figure 2. Lifecycle management use case.

BehavAuth contribution. BehavAuth will provide an authentication mechanism that will enable the end customer to authenticate through the TagItSmart mobile app to the TagItSmart platform. In this way, the user will be able to login to the TagItSmart mobile app and then scan the bar code in order to understand the recyclability of the product including additional information. BehavAuth will provide a secure and user-friendly authentication mechanism that will enable users to securely login to the mobile app and check the recyclability of the product as well as to get information on how to recycle the product based on contextual information.

1.1.3 Brand protection

This use case is focused on leveraging the smart-tags and the TagItSmart platform in order to product the brand of the products. The smart-tags initially will monitor the environmental conditions in which the product will be exposed such as ambient light and ambient temperature. Also the smart-tag will constitute a unique identification mechanism for the brand of the product, where the user will be able to scan the smart-tag through their mobile phone and the TagItSmart mobile app will provide information to user about the authenticity of the product. For a more detailed explanation of the process please refer to the Figure 3. This will enable the consumer to differentiate between the official and the fake products.



a)

b)

Figure 3. Brand protection use case.

BehavAuth contribution. BehavAuth will provide an authentication mechanism that will enable the end customer to authenticate through the TagItSmart mobile app to the TagItSmart platform. In this way, the user will be able to login to the TagItSmart mobile app and then scan the bar code in order to understand if the product is legitimate or not. BehavAuth will provide a secure and user-friendly authentication mechanism that will enable users to securely login to the mobile app and check the authenticity of the product while reducing counterfeiting.

1.1.4 Dynamic pricing

This use case is focused on providing the ability to the manufacturer to insert product information in the smart-tag and based on that allow the dynamic pricing concept to emerge. Once the product is created, the smart-tag is defined with information such as best before, ingredients, recycling, consumable before etc. This information allows the product change price depending for example on the current date, as particular thresholds of the information are added. Consumers are able to scan the smart-tags and through the mobile app they are able read information about the product such as best before, recycling, consumable before,

recipes. The mobile app communicates with the platform and the dynamic pricing calculation provides the update price of the product (See Figure 4).

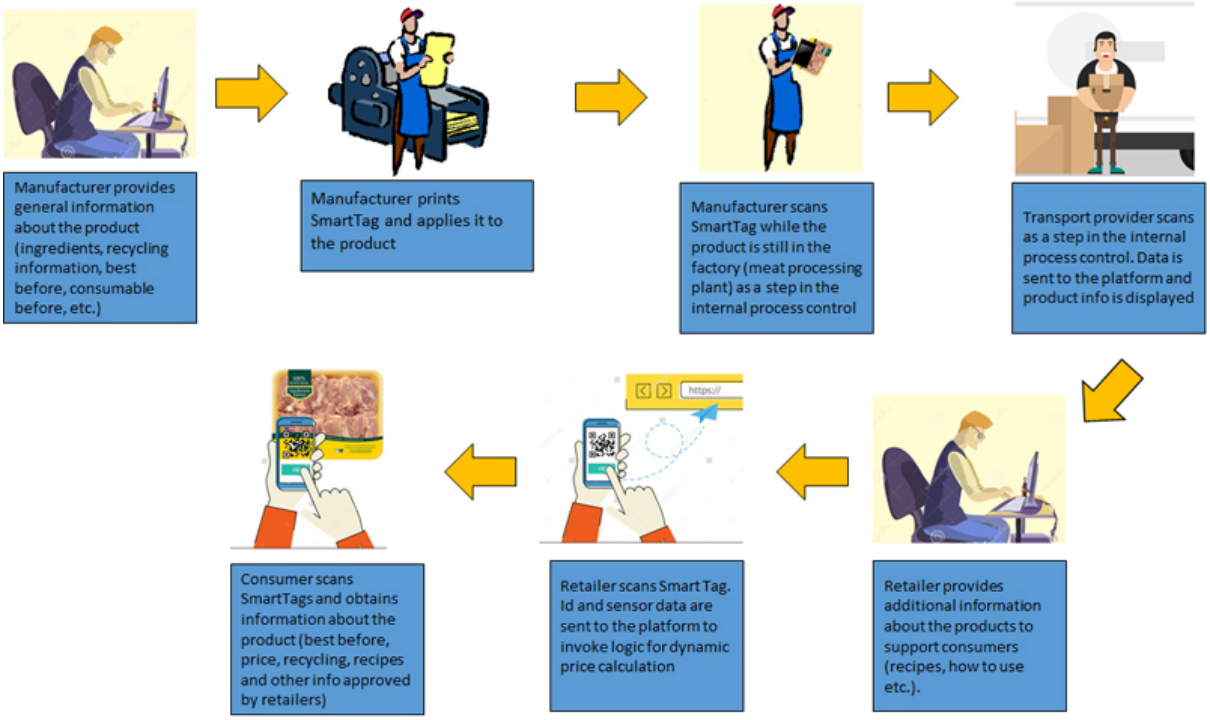


Figure 4. Dynamic pricing use case.

BehavAuth contribution. BehavAuth will provide an authentication mechanism that will enable the end customer to authenticate through the TagItSmart mobile app to the TagItSmart platform. In this way, the user will be able to login to the TagItSmart mobile app and then scan the bar code in order to read the dynamic price of the product. BehavAuth will provide a secure and user-friendly authentication mechanism that will enable users to securely login to the mobile app and check the various information of the product such as best before, consumable before, recycling, recipes as well as to calculate through secure communication with the TagItSmart platform the dynamic price of the product.

1.1.5 Home services

This use case focuses on monitoring, maintaining, repairing and replenishing of the heating equipment inside a household. Various of the TagItSmart services will be tested in real-world households, where the smart-tags will provide the appropriate environmental and contextual information regarding the heating equipment of a household. The monitored information is related to the actual heating equipment product, the environment in which it operates, the type of use performed by the owners of the product as well as additional services from the point-of-view of the product manufacturer. Figure 5 shows the overview of the use cases including the various stakeholders and the way they interact with the TagItSmart platform, the smart-tags and the different heating equipment.

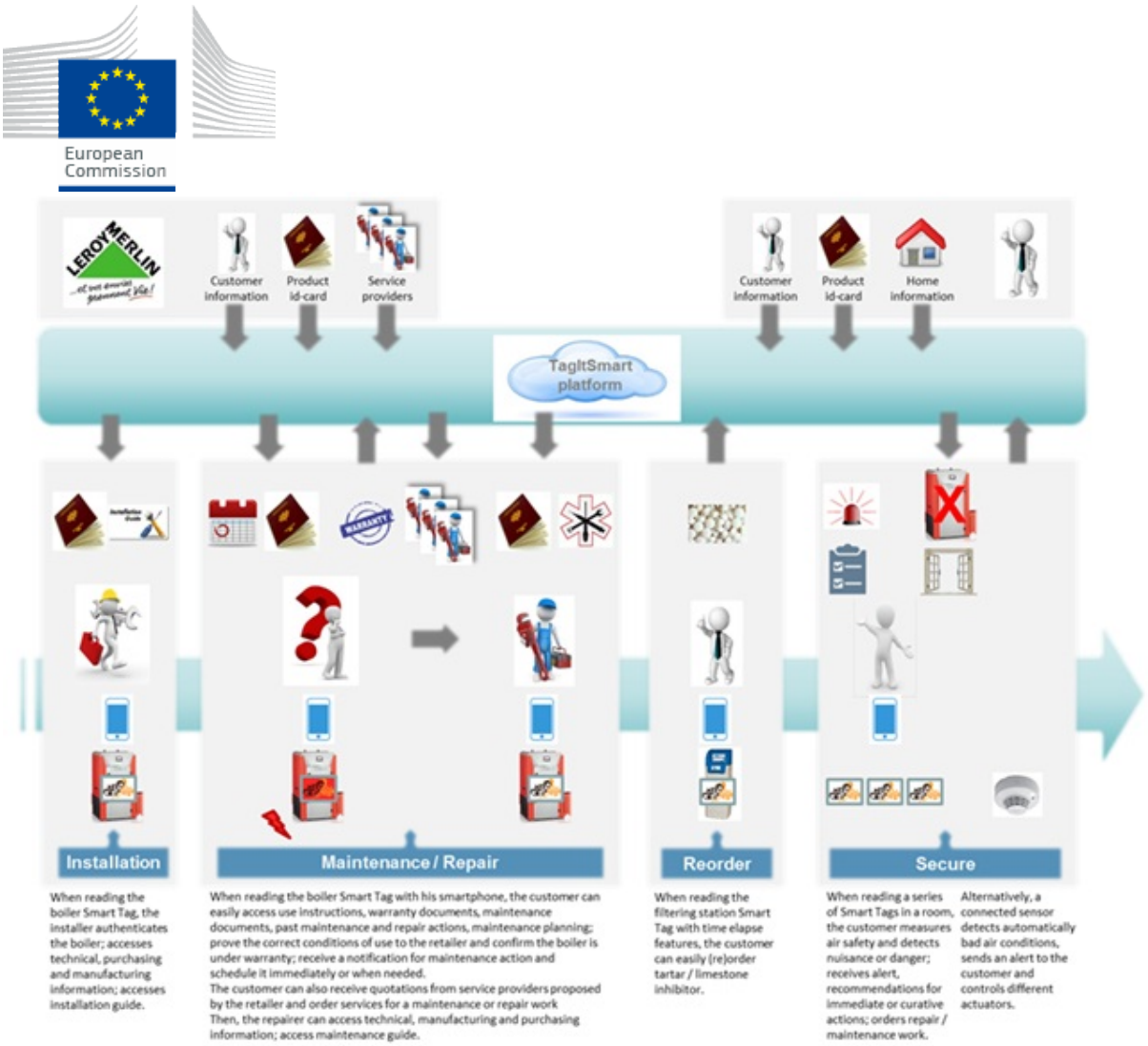


Figure 5. Home services use case.

BehavAuth contribution. BehavAuth will provide an authentication mechanism that will enable the end customer to authenticate through the TagItSmart mobile app to the TagItSmart platform. In this way, the user will be able to login to the TagItSmart mobile app and then scan the bar code in order to read the information from the different appliances for maintenance, monitoring, repair and replenishment.

1.2 Requirement analysis

This section describes the refined requirements derived initially from the use cases and the engagement of the various stakeholders and then refined through the integration process and the early stage trials. Table 1.1 provides a list of the requirements including category, requirement ID (REQ_ID) and description of each requirement.

Table 1.1. Functional requirements.

Category	REQ_ID	Description
Security	F01	The tool must provide a secure communication channel between the cloud and the mobile devices.

	F02	The platform must provide a secure communication channel between the cloud and the TagItSmart platform.
	F03	The platform must ensure the integrity of all the communications.
Privacy	F04	The platform must should store a pseudonym for each registered user.
	F05	Users' data should be processed only when explicit consent has been given by the users.
	F06	Users should have the ability to stop the data collection and processing at any time.
	F07	Only data required for the authentication mechanism should be collected.
	F08	Users should have the ability to delete the collected data and any extracted knowledge.
ID	F10	Only the pseudonym of each user will be stored, followed by the behavioural data.
	F11	The ID of each user will be anonymised.
	F12	The platform should allow the registration and deletion of users.
Authentication	F13	The authentication mechanism should user-friendly.
	F14	The authentication mechanism should allow developers to easily integrate with TagItSmart and other third-party platforms.
	F15	The behavioural patterns should accurately identify and verify the users.

Section 2 - Refined system design

This section provides the refined system design for the BehavAuth tool aligned with the TagItSmart platform. The design is split in three subsections that include the overall system design, the mobile app and the backend platform.

2.1 Overall system architecture

This section provides an overall description of the BehavAuth architecture and how it interconnects with the additional components that are introduced by the TagItSmart platform. The architecture of the system was designed with the collaboration of the involved stakeholders such as the TagItSmart consortium and in particular with the University of Surrey and University of Padova. This is because both partners of the TagItSmart project were involved in the development of authentication components for the TagItSmart platform and there was a need to create a consensus in the development process. Some of the key characteristics that were taken into account in the development process were the integration with the TagItSmart platform as well as the extendibility of the approach in order to allow other authentication components to be integrated with the overall system in an easy and seamless manner without the need of modifications in the core platform.

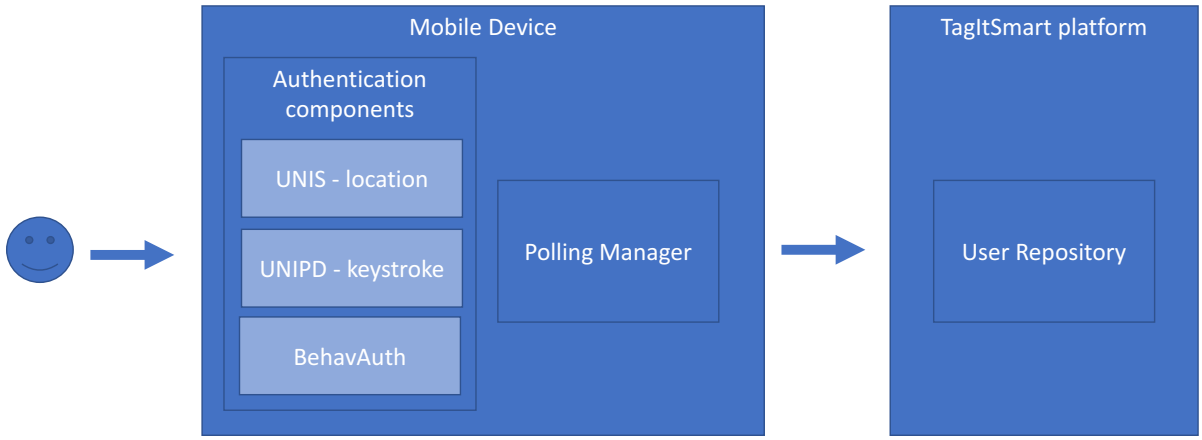


Figure 6. Overall system architecture.

Figure 6 provides an overview of the overall system architecture that was developed for the TagItSmart project, indicating the architecture and the integration process of the different components developed in the BehavAuth project as well as the TagItSmart project. The core entities of the overall system architecture are the mobile phone that the user owns and the TagItSmart platform.

At the mobile device level, the end user interacts with the different authentication components that are the University of Surrey location based authentication, the University of Padova keystroke based authentication components and the BehavAuth authentication app. All these three components provide some authentication estimation about the users, if they are the actual owners of the devices or not. The results of the different authentication components are forwarded to the Polling Manager that is located also in the mobile device. The Polling Manager is responsible for listening on a common communication channel for the authentication results from the different authentication components. In this way, multiple authentication apps could

be added to the mobile device without the need of modifying the initial core structure. The Polling Manager, once it receives authentication results from the different authentication components located on the same device, merges the different authentication results and propagates the outcome of the authentication process to the TagItSmart platform and in particular to the User Repository. The User Repository is a core component of the TagItSmart platform which manages and stores the users information. In this way, the TagItSmart platform is able to keep updated the authentication process, by continuously authenticating its users through these authentication components.

Following, there is a brief description of each of the core components that were showcased in Figure 6 and discussed above.

2.2 Mobile authentication app/library

This mobile authentication app/library is the core component of the BehavAuth project. It is an app that is installed at the users' smartphone and learns the behavioural patterns of the user and based on them authenticates the user to TagItSmart platform. The app collects Bluetooth and location data. The collected data are pre-processed through certain techniques and are forwarded to the behaviour extraction components. The first step in the behaviour inference process is the extraction of the behavioural cues i.e. social interactions, location, speed and distance within the environment. Each of these behavioural cues are extracted and are stored temporary in order to forward them to the different behavioural based authentication components. Each of the behavioural based authentication components infers the authentication result with respect to a particular behavioural cue. Then, the results of the authentication components are fused into the final authentication result through the authentication inference engine. The final result is propagated to the Polling Manager app and further to the TagItSmart platform.

Table 2.1. Sensor data

Sensor data	
Bluetooth data	Location data

More details about the authentication app/library are provided in Section 3 - and Section 4 - .

2.2.1 Difference with UNIS location based authentication

As mentioned above, the BehavAuth authentication app is focused on detecting various behavioural cues such as social interaction, location, speed and distance patterns. In parallel, the University of Surrey has already developed a location-based authentication component. Both location-based authenticators that have been developed are different and operate as complementary location based authentication tools. The UNIS location based authentication is understanding the mobility patterns of the user, considering the path the user leveraged to reach a particular location. The BehavAuth is focused on understanding which locations the user has visited at a certain time during the day and also considering the travel state (moving, standing) of the user. The UNIS authenticator is learning the mobility patterns of the user based on the path used to reach a location and the BehavAuth is learning the location patterns at which locations the user goes at certain times and days of the week. In a more tangible example, the UNIS authenticator will validate the user based on the path followed, from location A->B->C. The BehavAuth authenticator will validate the user based on the individual

patterns of location and time (A, 9:00am), (B, 3:00pm) and (C, 8:00pm). This indicates that the two location estimators operate complementary and do not overlap in terms of functionality.

2.3 TagItSmart polling manager app

The Polling manager app constitutes the core component that integrates the various authentication apps with the TagItSmart platform and in particular with the User Repository. This operates as an individual app that starts at the boot of the smartphone device. The sole purpose of this mobile app is to listen for the different authentication components of the TagItSmart platform, merge any authentication results that have been forwarded to the Polling Manager and then propagate the result of the merging of the authentication process to the TagItSmart platform and in particular to the User Repository. More details about the Polling Manager app are provided in Section 5 -

2.4 Backend platform

The backend platform of the current implementation of the BehavAuth system constitutes the TagItSmart platform. In particular, the BehavAuth authentication app performs the whole authentication process on the device. Then the Polling Manager app forwards the result of the different authentication components such as the BehavAuth, the location authenticator of University of Surrey and the keystroke authenticator of University of Padova, to the TagItSmart platform. The TagItSmart platform constitutes of many components, however the Polling Manager communicates only with the User Repository. The User Repository is a data store which contains information about the users involved in the TagItSmart ecosystem such as the users of the platform including personal information, preferences and optional information. In this way, the Polling Manager communicates through a REST channel with the User Repository to inform about the authentication result. More details about the User Repository component of the TagItSmart platform are provided in D1.2 [38] and D4.2 [39].

Section 3 - Social interaction detection

This section provides the description of the social interaction detection mechanism [2] that was developed in order to understand the existing social interactions and feed them to the authentication mechanism.

3.1 Related Work

Various approaches have been proposed in order to detect social interactions in an automatic fashion. Depending on the selected underlying technology, each of these approaches comes with its advantages and disadvantages. In the following paragraphs, we briefly survey the related work and while doing so we highlight how the proposed social interaction detection mechanism differs from them.

3.1.1 Wearable devices

The main idea in wearable approaches is to detect people in proximity of each user that are having an almost opposite facing direction. Although this may result in many false positive errors, the accuracy and effectiveness of the approach in real-world environments has been emphasised in many of the state-of-the-art solutions. The F2F orientation and proximity of the users has been realised through different approaches such as utilising the conic range produced by Infrared (IR) transceiver or the semi-circle radius generated by RFID-tag at the facing direction of the user.

One of the first research in the field was conducted by Choudhury [4], where they developed a sociometric badge; a custom wearable device combining multiple modalities such as IR transceiver for F2F interaction detection and microphone for speech detection. For social interaction detection they trained an HMM over the noisy IR data and a 2-stage HMM for estimating when two people are having a conversation.

In Groh et al [5] they proposed a system for quantifying social interactions comprising the distance among the users and their relative orientation at a specific time. The system was based on IR technology, requiring the deployment of eight IR cameras in the experiment room; four placed on the floor and four attached to the ceiling. Each participant was associated with two IR beacons, one around the throat to measure the relative distance of body centres and one attached to the shoulder to estimate the orientation of the torsos.

Another recent work emerged from the SocioPatterns project [6], where small RFID tags are utilised. These tags continuously broadcast small data packets that are received by a number of base stations and relayed through a local network to a server. By adjusting the signal resolution of the RFID tag to 1-2 meters and considering the absorption of the human body, the detection is confined to the plane that the torsos is facing.

The major disadvantage of these approaches is the intrusiveness of these methods as they are not part of users' daily lives. Further, the user is required to wear the device at a specific on-body position which introduces obtrusiveness. In essence, forcing the user to wear an unknown for him device at a certain position will affect his spontaneity and thus his social behaviour. Furthermore, some of these systems collect the data and forward them to a centralised server, which requires special infrastructure (RFID Reader) that must be deployed around the environment where the measurements will be performed. The demand for external infrastructure limits the spectrum and mobility of the deployment but also increases the intrusiveness of the system. However, the proposed social interaction detection mechanism is based on mobile phones, pervasive and ubiquitous devices included in user's daily life, and

does not require any additional infrastructure (even Internet connectivity); thus, preserves user's privacy and does not induce any mobility limitations.

3.1.2 Smartphones

In smartphone based sensing realm the majority of the work has been focused only on proximity detection. In these approaches authors focused on estimating only which users are in proximity and did not take into consideration the directionality of the participants. The existing trend for detecting social interactions through proximity on mobile phone based sensing platforms has been based on data collection (simple discovery) achieves resolution proportional to the utilised technology mean, via Bluetooth, GSM signals or Wi-Fi access points such as in [7], CenceMe [8], PeopleTones [9], Serendipity [10] and SoundSense [11]. As a result, there would be numerous occasions when being in proximity does not guarantee the existence of a F2F interaction, leading to a considerable amount of false positive error in social interaction detection.

In order to improve the simple discovery approach, fine-grain distance estimation was proposed in multiple works to improve the granularity. Among them is Comm2Sense [12], a client-server implementation performing proximity detection by analysing WiFi RSSI. For the detection of nearby devices, each device must operate as a hotspot and then switch to normal WiFi. However, in order to be able to enable and disable the hotspot mode and set the transmission power to 0dBm, rooting of the Android device is required. By applying machine learning techniques on 20-sample RSSI window for distance estimation, they claim of achieving a median error of 0.5m in indoor and outdoor environments.

PhoneMonitor [13] is a monitoring application that logs various sensors of a mobile phone for the detection of F2F interactions based on proximity estimation. A number of experiments are performed in order to increase the accuracy of estimating the distance among the devices. A probabilistic model for the Bluetooth RSSI is proposed to classify if the users' interpersonal distance is suitable to perform F2F interaction. Although PhoneMonitor, estimates if the users are in proximity or not while they are required to hold the device in their hands, authors introduce privacy issues by performing off-line analysis.

MAUC [14] is a contact detection method for mobile phones that based on user's movements, detects start and ending point of a contact. The authors infer about the existence of social interaction through simple threshold on Bluetooth RSSI measurements, which is erroneous due to large fluctuation in RSSI values. However, MAUC is not able to detect social interactions when the users are moving.

A significant part of proximity detection has focused on leveraging different path loss models. Among them, the free space path loss model (Free Space PLM) which refers to theoretical conditions and the path loss model [15] specific for indoor office environments (Office PLM). A more complicated approach is BlueEye [16], a system targeting proximity detection in crowded environments. The authors induct a modified path loss model which considers the relative orientation between the devices and Bluetooth RSSI in order to estimate the interpersonal distance of the users. The system initially performs calibration to estimate the environmental constants and then executes the modified path loss model followed by the appliance of K-means clustering which leads to the detection of the distance between the mobile devices. Further, BlueEye requires pre-training for each environment in which the approach is applied in order to detect social interactions.

Virtual Compass [17] is a system for detecting social interactions among people. It utilises the RSSI values from Bluetooth and WiFi interfaces to build regression models based on pre-

acquired training set, for estimating distance between two devices. In their feature extraction process, they extracted the mean RSSI and the uncertainty for each distance. Also, they propose an algorithm to estimate the spatial arrangement of nearby devices. However, because they utilise both Bluetooth and WiFi interfaces in the inference process, energy consumption constitutes an important issue that prohibits the usage in real-world applications. Also, until this moment WiFi on off-the-shelf mobile phones do not allow ad-hoc communication without specific firmware modifications.

A recent attempt towards F2F interaction detection on mobile phones is presented in [18] however it comes short in several aspects. The approach combines proximity detection, calculation of relative orientation and speech activity detection through external accelerometer attached to the chest of the user. The inference of F2F interaction was executed off-line and the mobile phone device was placed at a predefined on-body position. For estimating the interpersonal distance they utilised a propagation model with an environmental parameter, that was acquired through a calibration processes, measuring the WiFi RSSI at 1-meter distance for a few minutes. However, the device demands firmware modification in order to operate in ad-hoc discoverable mode. This functionality is achieved by a time-consuming process of turning on/off the hotspot/scan mode of WiFi interface, which limits the granularity of the detection time to 1 minute.

Existing techniques although successful in their domain are facing these several challenges that limit their applicability in real-world environment. In the next subsection, a social interaction detection mechanism is introduced as an initial attempt to resolve these problems and provide a non-intrusive and robust tool for long-term monitoring of F2F interactions.

3.1.3 Proposed approach

The proposed system performs social interaction detection by leveraging the Bluetooth RSSI from nearby Bluetooth devices. The proposed system is based only on commercial off the shelf smartphones and does not require any centralised coordinator to perform the inference. The Bluetooth RSSI is fed into machine-learning models that perform the social interaction detection. As the inference is performed on the smartphone device, the data are not transmitted to any third party, thus the privacy of the users' is preserved. The logged data are available only at the user's disposal. Regarding the inference that is performed, it calculates the interpersonal distance with each user and finally performs classification, depending on the selected target class (proximity or interaction zone), about the occurrence of F2F interaction.

One of the major shortcomings of the state-of-the-art solutions is the dependency on the large number of RSSI samples for estimating interpersonal distance. Due to the Bluetooth architecture limitation, acquiring this number of samples imposes significant delay between each estimation of the proximity. In addition, we propose a novel machine learning based technique which detects the user's proximity and interaction zones with significant less number of samples i.e. 6 samples.

3.2 Proximity

Proxemics have preoccupied the field of psychology due to importance of interpersonal distance among people in order to characterise their relation. In [19], Hall inspired from spacing mechanisms among animals maps the interpersonal space among people into four distinct zones:

- Public Zone. The interpersonal distance is over 3.5m and people do not usually interact among each other.
- Social Zone. People interact in this zone but do not have any special bonding with the interpersonal distance being between 1.5m and 3.5m.
- Personal Zone. Interaction in this zone indicates a personal relationship and interpersonal distance 0.5m to 1.5m among these people.
- Intimate Zone. Interactions encountered in this zone, usually involves physical contact and the interpersonal distance corresponds to 0.5m or less.

These zones are not strict and can vary depending on the origin of the people, their personality and environmental factors.

Proximity detection is a field in which various research have been conducted. Most of these works intended to detect ongoing social or to limited extend F2F interactions in different situations. Depending on the range of the communication interface and the techniques they apply, different accuracy results have been achieved. Based on the accuracy, it is feasible to infer if a person is in proximity with another person or not.

In this study, proximity boundary is set to 3.0m. The boundary was decided due to the fact that F2F interactions are characterized by immediacy and tend to occur in closer interpersonal distances because of the intimacy conveyed from the specific spatial formation [20]. We utilise the same interaction zones and corresponding boundaries from the above presented psychology literature for proximity detection. Still, depending on the origin of the participants, it is possible to retrain our approach on the boundaries of the specific culture.

3.3 System design

Realisation of the aforementioned concept on off-the-shelf smartphones is not as straight forward as it may seems. In case of interpersonal distance estimation, a novel technique is required to address the shortcomings of exiting solutions in terms of accuracy and reliability. This section elaborates our efforts on resolving these challenges.

Section 3.1.2 presented various approaches for distance estimation between mobile phones via different short-range communication means. The most common ways to estimate the distance between two or more nearby devices are Time of Arrival (ToA), Time Difference of Arrival (TDoA), Angle of Arrival (AoA) and RSSI. The first two methods calculate the time duration from transmitting to receiving a packet, which requires specific time synchronisation. In the described approach, the RSSI was selected in order to estimate the interpersonal distance due to lack of detailed time synchronisation information among the devices. The main disadvantage of this approach is the unreliability of RSSI measurements primarily due to device orientation, device variability, human body absorption and environmental factors.

In contrast to these methods, the proposed social interaction provides relative distance estimation at both coarse and fine-grained resolutions corresponding to proximity detection and recognition of interaction zones.

3.3.1 Overview

Empirical evaluation results from replicating the prior arts solutions, indicated that path loss based models fail to provide an adequate accuracy and are very sensitive to the environmental

conditions. Conventional machine learning techniques showed better performance in coarse-grain proximity recognition but still come short in recognition of the interaction zones.

Learned from these observations we developed a new machine learning based solution. The key idea is to perform the classification of the interaction zones in a hierarchical fashion. Our Hierarchical classifier consists of 2 layers of machine learning models. Classifiers at the first layer are particularly trained for a given region i.e. social, personal or public zone and in this regard from a set of domain expert classifiers. Classification confidence values from this layer provide a fuzzy membership for each window of RSSI values. The second layer classifier then strives to find the optimum thresholds for mapping the membership values into the correct interaction zone. It is worth noting that all classifiers that we evaluated faced significant difficulties in discerning between personal and intimate interaction zone with adequate performance. We therefore merged intimate zone and focused our classification on personal, social and public zones.

In addition to that we develop a single classifier to detect the coarse-grained distances i.e. proximity. To mitigate the problems associated with previous similar works, particular consideration is taken in training the model and selecting the input feature sets.

3.3.2 Training the detection models

Our roadmap for developing the interpersonal distance classifiers is to first generate a generic training set and then produce a bank of features from this training set. Next, we perform feature selection to select the most informative and less redundant feature set and finally we evaluate different classifiers at each level of hierarchy to find the most appropriate selection. These steps are detailed as follows.

In order to construct the training set for our interaction classifier, we conducted an extensive data collection campaign in an indoor office environment using HTC One S smartphones. During each experiment the Bluetooth interface of one devices was set to discoverable mode, while the other device was performing a Bluetooth discovery. The experiment acquired RSSI readings between two smartphones for different distances and device orientations. The experiments included eight distinct distances from 0.5m to 4.0m, every 0.5m. For each of the different distances, three device orientations were scrutinized: a) Screen-to-screen b) Screen-to-Back c) Back-to-Back. For reasons of statistical significance, a large number of measurements (2,000 samples) was acquired for each distance/orientation combination. Overall the experiment produced 48,000 RSSI measurements. Due to the extremely lengthy data collection process, we replaced humans with water-filled cylinder to which the devices where attached. The devices were always in vertical orientation and were placed at a height of 0.8m from the floor.

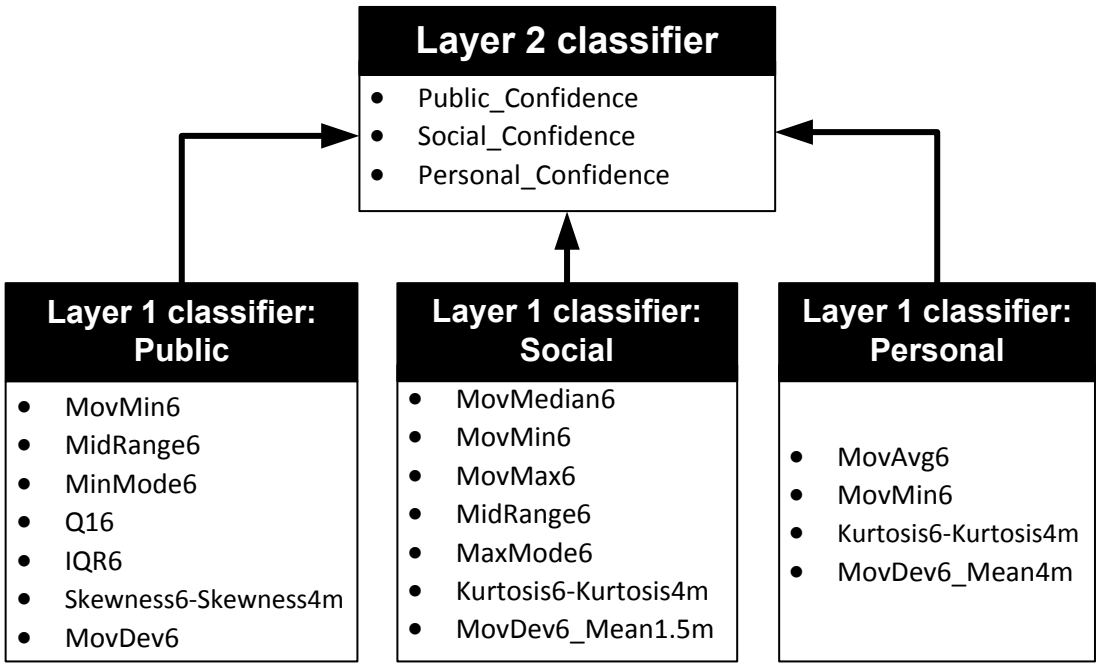


Figure 7. Features for 2-layer hierarchical interaction zone classifier [2].

Table 3.1, Table 3.2 and Table 3.3 provide an overview of the features considered for feature extraction. Assuming that the maximum number of samples required for performing one inference is 6 samples, then the window sizes 2, 3, 4, 5, 6 will be used and will produce 5 discrete features for each basic feature in Table 3.1. Basic Features. This window size is particularly selected through several empirical experiments, as it requires an acceptable delay for data acquisition and at the same time is long enough to provide informative features. In particular, each device requires around 1 minute for the first 6-sample window and then each new sample is added to the window; in case of lack of RSSI samples after a certain amount of time, the window is cleared and the process restarts to keep the values updated.

$$Relative_{Basic} = Basic_{window} - Basic_{class}$$

Table 3.1. Basic Features

	Name	Description
1	Mean	The mean value of specific window of samples
2	Median	The median of specific window of samples
3	Min	The minimum value of specific window of samples
4	Max	The maximum value of specific window of samples
5	MidRange	The mean value of minimum and maximum of specific window of samples
6	MinMode	The most frequent value of specific window of samples. If multiple, it returns the minimum of them.
7	MaxMode	The most frequent value of specific window of samples. If multiple, it returns the maximum of them.
8	Percentile	25th, 75th and 90th percentile of specific window of samples.
9	IQR	The inter-quartile range of specific window of samples.
10	MAD	The median absolute deviation of specific window of samples.

11	STD	The standard deviation of specific window of samples.
12	Kurtosis	The kurtosis of specific window of samples.
13	Skewness	The skewness of specific window of samples.

Table 3.2. Relative features.

1	Interaction Zone	The difference between the basic feature of a window of samples and each of the interaction zones (Public, Social, Personal).
2	Distance	The difference between the basic feature of a window of samples and each of the distances (0.5m, 1m, 1.5m, …, 4m).
3	Orientation	The difference between the basic feature of a window of samples and each of the distances and orientations (0.5m Screen-to-Screen, 0.5m Back-to-Screen, 0.5m Back-to-Back, etc.).

Table 3.3. Combined features.

1	MovDev	The deviation of a specific window of samples with respect to the mean of a class (interaction zone, distance, distance \& orientation).
2	ZScore	The z-score of a specific window of samples with respect to the mean and standard deviation of a target class (interaction zone, distance, distance \& orientation).

Likewise, the distance of each of the produced basic features from the corresponding value of the class was computed, resulting in 2625 relative features, see Table 3.2. Moreover, from Table 3.3 the moving deviation and z-score for each window of samples regarding each class was calculated, concluding that interaction zone detection with a window of 6 samples will generate overall 3050 distinct features.

Due to the large amount of features produced, further feature reduction techniques were applied. Initially a subset of features is selected by considering the level of consistency [22] of each distinct feature with respect to the target class. After obtaining this subset of features, a wrapper subset evaluation [25] was performed to acquire an optimised feature set for the selected classifiers.

Table 3.4. Single classifier features

MovMedian6	MovMax3	STD6	MovDev6-Mean3m
MovDev4-Mean4mB2F	Kurtosis6 - Kurtosis4mB2F	MovDev6-Mean4mF2F	MovDev6-Mean1mF2F

Table 3.4 shows the optimised feature set for single classifier, while the Figure 7 shows the optimum feature sets for the hierarchical approach. The number at end of each feature corresponds to the number of samples used from the 6-sample window. For feature naming, except for the number of samples, the corresponding distance is used e.g. 1m and the orientation of the devices (Screen-to-screen = F2F, Back-to- screen = B2F, Back-to-back = B2B). Furthermore, the four selected feature sets correspond to four distinct classifiers.

Our evaluations determined that MultiBoostAB [23] with decision tree J48 [24] as the best classifier for our interaction zones. We initially trained single classifier for all three zones based

on several algorithms and concluded on MultiBoostAB which achieved the highest accuracy in a robustly manner due to its native capability for variance and bias reduction. In order to boost its performance further, we explored the use of a hierarchical classifier as shown in Figure 7 The previously achieved accuracy and inference robustness of MultiBoostAB led as also to train the models of each layer of the hierarchical classifier based on the same algorithm, which succeeded in an even higher accuracy than the single classifier. Because the Hierarchical classifier discriminates the public zone from the other zones, there was no need to train a new Proximity classifier, instead the personal and social interaction zones of the Hierarchical classifier were classified as proximity.

3.3.3 Implementation

The architecture is fully distributed which makes it more reliable with respect to the conventional centralised orchestration that is prone to node failure. The architecture is particularly favoured in pervasive applications as it does not rely on any infrastructure. Off-the-self smartphones are the only hardware devices required for the setup and deployment of the system.

Our initial implementation is based on the Android mobile operating system due to its openness to develop advanced sensing applications. More specifically, the app is executed as a background service without interrupting or affecting user's experience. It has a multi-threaded design, in which each module is executed in a separate thread to distribute locally the workload and to eliminate the slightest effect on user experience.

Android 4 based versions, allow Bluetooth interfaces to be set always to discoverable mode without any time limitation, as opposed to previous versions in which super user permissions were required. The system makes use of this feature by setting Bluetooth discoverability without time limit during initialisation.

In order to detect nearby devices, a standard Bluetooth discovery is triggered periodically. The device discovery process requires about 12 seconds according to the Android API, which is confirmed by our empirical measurements.

Section 4 - Mobile authentication app/library

This section provides a detailed analysis of the mobile authentication app and the authentication components developed in order to validate the identity of the user through machine-learning techniques by understanding and learning various behavioural cues and patterns of the users.

4.1 Overall design

This section provides an overview of the design of the mobile app/library that performs authentication of the user by leveraging sensor data generated by the integrated sensors of the smartphone device. The mobile app/library collects sensor data, extracts behavioural cues that will allow the derivation of behavioural patterns of the user and further will enable the validation of the identity of the user. The sensor data are initially pre-processed in order to convert them in the appropriate format. Following, the normalised sensor data are forwarded to the behavioural cues extraction components. Each of the behavioural cues component processes the sensor data that are related to the particular module. The data stream is fed in the behaviour extraction module that learns and authenticates the user. The results from the learning and authentication process are forwarded to the authentication inference component that fuses the results from the different authentication components. Once the authentication inference component fuses the different authentication results, an overall authentication decision is generated and propagated to the interface that the BehavAuth library exposes to third party apps.

The design of the library/app is event based in order to minimise the energy consumption of the smartphone. The library is coordinated by a background service that operates as a manager responsible for the interconnection of the different components of the authentication library. The data collection process that is taking place is event based, so when each sensor generates a sample, then the sample is forwarded to the behavioural cue and the behaviour extraction modules. The behavioural cues and behavioural pattern extraction modules are executed on separate threads, in order to off-load the background service that performs the coordination of the library and to enable constant availability of the service without interruption.

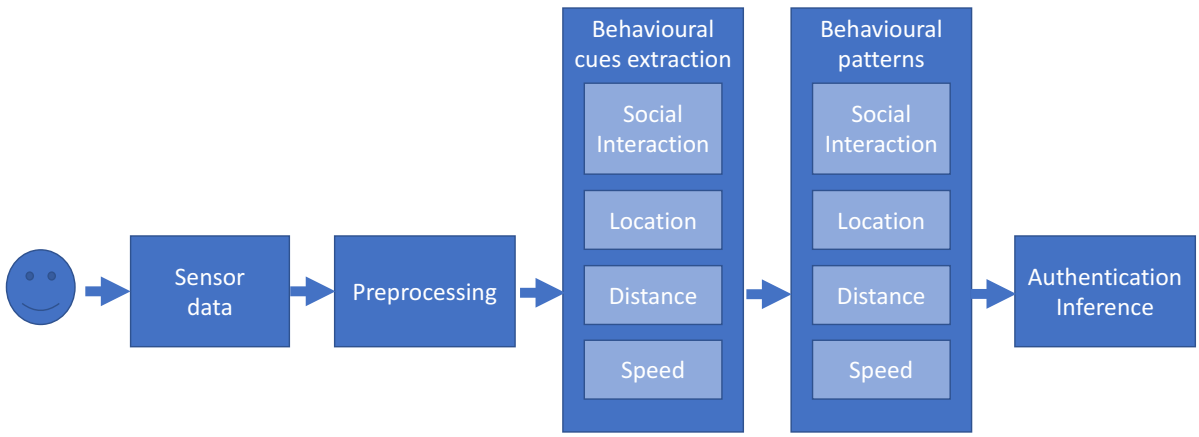


Figure 8. Authentication pipeline

The architecture followed in this mobile app/library is derived from modern software design patterns for mobile devices. The initial key concept is the fact that the mobile library is event based, to remove any need for polling the sensors for new values and increasing the energy consumption of the device. The observer pattern is applied in the architecture to enable each of the authentication components to wait for new sensor data generated and once they are generated to initiate any processing and inference. As the inference process requires multiple stages of processing and knowledge extraction, another software design pattern that is used is the pipeline. Once new sensor data are generated, the pipeline architecture is applied and data are fed in the pipeline and then follow a certain process to extract the appropriate authentication knowledge. In the next sections, each of the stages of the pipeline will be analysed in more detail for the reader to receive a deeper knowledge of the processing and knowledge extraction technologies used in each stage of the pipeline.

4.2 Data collection

This section describes the data collection process performed for each of the authentication components developed, considering the various major parameters that provide the information in order to perform the inference about the authenticity of the user.

4.2.1 Dataset

For the evaluation of the authentication components a state-of-the-art dataset was used. This dataset was selected for a number of reasons including that it provided the required number of users (10 users), the duration of the experiment that lasted 10 weeks and also the fact that the dataset included all the required data for the BehavAuth system. Among the students who completed the study, are undergraduates and graduate students. In terms of gender, the dataset included both males and females. The participants that were involved in the study were enrolled in the CS65 Smartphone Programming class, a computer science programming class at Dartmouth College that is offered to both undergraduate and graduate students during Spring term in 2013. All students were offered an unlocked Android Nexus 4s to complete assignments and class projects. Each mobile phone had deployed a data collection app that was monitoring the selected sensors and without any interaction with the user, the mobile phones were uploading the data to the backend server. The data off-loading was performed only when the device was charging and only through the WiFi interface. The uploaded data were monitored through scripts to understand if the users are using the phones and participating in data collection.

4.2.2 Pre-processing

The data collected from the sensors of the device are pre-processed in order to reach the appropriate format that is required to perform the inference pipeline. In this section, the pre-processing steps that are described, are applied in the mobile app, in addition to the pre-processing steps that were applied to the evaluation dataset. The steps related to the data labelling and the merging of the datasets are not applied in the mobile app and were only for the evaluation dataset.

The StudentLife dataset [26] provided by Dartmouth University is a quite large and extensive dataset that included various types of sensor and questionnaire data. In order to be able to leverage the collected data in the BehavAuth project, there was a need to perform some pre-processing of the data, before applying the machine-learning models required for the authentication process.

The data pre-processing procedure required the following steps:

1. **Dataset**. Initially there was a need to extract only the sensor data related to the BehavAuth system. This extraction refers to the Bluetooth and Location measurements for the first 10 users.

2. **Labelling the data**. As the data for the Bluetooth and the Location measurements for each user are provided in individual file, there was a need to label the data with the ID of the user. In particular, another column was added to the dataset describing the ground truth which is the ID of the user, to whom the data belong. In this way, we will be able to analyse the results provided by the machine-learning algorithms in order to understand the performance of the system by comparing the outcome of the algorithms with the actual value.

3. **Additional features**. The StudentLife dataset includes various sensor and questionnaire data in a digital form. However, it does not include some specific features such as the speed and distance within the environment that were mentioned in D1.1 [21]. The location dataset includes a field "speed" which refers to the instantaneous speed of the user the moment the location measurement is collected. As there are many points in the dataset where the speed is 0, so there is a need to calculate a more indicative measurement of the speed of the user in the environment. For that reason, the location data combined with the timestamps of these locations were used in order to compute the users' speed and distance within the environment.

    a. **Distance within the environment**. The distance that the user travels in the environment is not provided as a sensor at the commercial-off-the-shelf mobile phones. For that reason, there is a need to compute the distance the users has travelled through the data that have already been collected from the smartphone. The distance data are computed through the knowledge of the locations that the users has visited and the time when the user visited these particular locations. Through the knowledge of the above facts, the system computes the average distance that the user travels within the environment.
    b. **Speed within the environment**. The speed of the users' movement within the environment is provided by the mobile operating system API. However, the speed that is provided is the instantaneous speed calculated at the moment when the location measurement is taken. As in many cases the instantaneous speed of the user is zero, there was a need to compute the average speed of the user within the environment among the different location points. The average movement speed constituted a more representative measurement of users' speed within the environment.

4. **Merging the datasets**. Having extracted the appropriate data, labelled them with the ground truth to be able to calculate various performance metrics and added the additional features required to extract the appropriate information, the various small datasets were merged into one concrete data for social interactions and one for location/speed/distance. These two datasets were not further merged as the number of samples of each of dataset is different, given the different sampling frequency of each of the sensors (Bluetooth and Location/GPS/WiFi/Cellular).

4.2.3 Social interaction data

The social interaction data are derived mainly from the Bluetooth interface of commercial-off-the-shelf mobile phones. The Bluetooth interface constitutes a short-range communication mean to exchange information among devices without the need of additional infrastructure.

The Bluetooth interface has the ability to discover devices in close vicinity (for indoor environment devices that are in less than 35m range). For that reason, the Bluetooth interface constitutes ideal sensor to detect devices and users that are in vicinity to interact.

For the extraction of the social interaction data, the Bluetooth traces from the StudentLife dataset were used. The dataset includes the following data:

- **Timestamp**. The timestamp of the measurement
- **MAC address**. The MAC address of the Bluetooth interface of the detected device that was in vicinity.
- **Class id**. The class id of the Bluetooth interface of the detected device that was in vicinity.
- **Level**. The received signal strength of the detected device that was in vicinity.
- **Class.** The ground truth of the user, meaning the user to whom the data belong.

The features were defined based on the information provided by the Bluetooth sensor. The 'Time' factor was important to be considered in the inference process as the social interaction patterns are time dependent. For example, a person may interact with a certain number of people at a certain time of the day, which can be more informative for the algorithm in contrast to just consider the social interaction. The MAC address operates as an identifier of the person with whom the users socially interact. As the MAC address is a unique identifier of the device interface it constitutes an ideal identification method. The class ID is the Bluetooth class and is specific to the device, this operates as an additional device verification method. The Level feature is the received signal strength and is used to understand the interpersonal distance among the users as well as the existence of a social interaction.

4.2.4 Location data

The location data are collected from the virtual location sensor that is provided by the Android mobile operating system. The majority of today's smartphones have integrated a GPS location sensor. However, as the GPS constitutes a power-hungry sensor there was a need for the Android operating system to develop some virtual sensor that integrates various location estimators to provide a less power hungry solution. For that reason, the Android operating system provides a location sensor that fuses the location estimations from the GPS sensors, from WiFi interface as well as from the cellular network. About the WiFi interface and the cellular localisation techniques, multiple approaches have been proposed by the literature and the analysis of these techniques falls out of the scope of this document. The information required is only related to the fact that the location data are extracted from a virtual sensor that fuses the location estimations of the GPS, WiFi and cellular interfaces.

For the extraction of the location data, the location traces from the StudentLife dataset were used. The dataset includes the following data:

- **Timestamp**. The timestamp of the measurement
- **Provider**. The provider is the location provider, meaning which sensor was used from the sensor fusion mechanism. This can be either network or GPS.
- **Network type**. This field provides additional information about the above field, when the provider is defined as network then the network type could be either cellular or Wi-Fi.
- **Accuracy**. The accuracy of the estimated location.
- **Latitude.** The latitude of the location in degrees.
- **Longitude.** The longitude of the location in degrees.
- **Altitude.** The altitude of the location.
- **Bearing.** The bearing of the location.

- **Speed.** The instantaneous speed of the user when the location estimation was received.
- **Travelstate.** The instantaneous movement type of the user (moving, standing).
- **Class.** The ground truth of the user, meaning the user to whom the data belong.

This is the list of the features used for the creation of the machine-learning model in order to authenticate the users based on the location patterns of the user. The location patterns are dependent on the time, as users may be at certain locations during a particular period of time. The features called Provider and Network type are related to the type of sensor that was used in order to provide the location estimation; this is included in the knowledge extraction process in order to take into account the type of sensor used to provide the data. The accuracy is an estimation of the accuracy of the location measurement. The features latitude, longitude, altitude and bearing are parameters to define the location of the user with respect to the earth's coordinates. The speed is an estimation of the users' instantaneous speed at the moment that the location measurement was retrieved from the sensor. Similarly, the travel state provides information about the user, in case the user is moving or standing. These all features provide information about either the mobility of the user or the accuracy of the location estimation.

4.2.5 Speed data

The Android location data provides multiple types of information that accompany the actual location estimation as shown in the previous subsection. One of this information is also the instantaneous speed of the user at the moment the location estimation was generated from the fused location sensor. As observed in the StudentLife dataset the speed field of the location sensor provides often zero speed for the user. This is happening most probably because the user is not moving at the point when the location estimation is generated. However, this does not allow the BehavAuth system to leverage the speed of the user in the environment in order to learn the speed patterns. For that reason, the average speed of the user in the environment was calculated. The average speed was calculated through the knowledge of the current and previous location of the user and the time required to move from one point to the other. This provided a more representative movement estimation of the user. More details about the calculation process is provided in the next paragraph where the behavioural cues are calculated.

In the previous case of the location-based pattern recognition, the smartphone has a dedicated sensor that provides location measurements. However, there is no sensor that provides a measurement of the users' speed. So, there is a need to derive the users speed through another sensor. The majority of the smartphones today have integrated an accelerometer that provides instantaneous measurement of 3-dimensional acceleration, from which the speed could be derived. However, in order to derive the speed from the accelerometer there is a need to compute the integral of the acceleration, which over time produces a drift due to the accumulated error. Another way that could provide an average estimation of the users' speed over the time is to take into account the distance the user moved over a period of time.

The average speed is provided by a general form velocity = distance / time. It was assumed that each user started to use the mobile data collection app from stance, where velocity was 0. So, the initial speed sample for each user is 0. Then, for each sequential pair of locations and timestamps, the distance and time were calculated. The distance was calculated similarly with the (a) point in kilometres. The time was calculated by subtracting the newest timestamp minus the previous timestamp – both timestamps were represented in Unix time – which resulted in the time in seconds and was converted into hours. Using the initially mentioned formula the speed was calculated and provided an average speed of the user in the environment.

For the extraction of the speed data, the speed traces from the StudentLife dataset were used. The dataset includes the following data:

- **Latitude.** The latitude of the location in degrees.
- **Longitude.** The longitude of the location in degrees.
- **Speed.** The average speed of the user when the location estimation was received.
- **Travelstate.** The instantaneous movement type of the user (moving, stationary).
- **Class.** The ground truth of the user, meaning the user to whom the data belong.

The list above presents the features used for the development of the authentication models using machine-learning techniques. The feature set combines the location at which the user is with the average speed within the environment as calculated above and the travel state of the user. The latitude and longitude provide the location of the user with respect to the earth's coordinates in degrees. The speed the user is travelling within the environment refers to the average speed among two location estimations and is represented in km/h. The travel state provides information about the users regarding their movement state. All these features were combined in order to train a machine-learning model for authenticating the user based on the average speed of the user within the environment.

4.2.6 Distance data

The distance that the user has moved in the environment during a certain period could constitute a significant parameter that would allow the differentiation of the users. Android smartphone devices do not provide a sensor that generates the distance that the user has moved. The distance that a user has moved can be generated implicitly through the location data, where a method in the Android location API allows the developers to generate the distance moved during a certain period. The distance estimation can be calculated through the knowledge of the last two location estimations. Particular techniques are required to calculate the distance that the user has moved as the location estimations are based on earth coordinates and are represented in the degrees. The distance calculation technique is further analysed in the behavioural cue extraction section.

The commercial off-the-shelf mobile phones do not provide a sensor for the distance that the user has moved within the environment. For that reason, there is a need to compute the distance the user has moved within the environment. Various methods have been proposed in order to compute the distance within the environment using sensors such as GPS or by using the pedometers in order to compute the distance that the user has walked. The use of location points from the fused location sensor was selected, from which the average travelled distance will be calculated in the next paragraph.

The distance within the environment refers to the average distance the user moves within the environment every period of the location sampling frequency. The distance was calculated using the current and the previous location measurements represented in latitude and longitude. As latitude and longitude are in degrees there is a need to apply special techniques for calculating the distance. The initial approach would be to use Euclidean distance; however, this does not take into account that the measurements are in degrees and also that the earth is a sphere. Another solution is Haversine formula [27], which considers that the earth is a sphere but according to literature [28] results in 0.5% error. For the above reasons, BehavAuth focused on using Vincenty [29] which provides more accurate ellipsoidal models. The distance is represented in kilometres and refers to the average distance between two sequential location measurements.

For the extraction of the speed data, the speed traces from the StudentLife dataset were used. The dataset includes the following data:

- **Latitude.** The latitude of the location in degrees.
- **Longitude.** The longitude of the location in degrees.
- **Distance.** The average distance of the user when the location estimation was received.
- **Travelstate.** The instantaneous movement type of the user (moving, stationary).
- **Class.** The ground truth of the user, meaning the user to whom the data belong.

An overview of the features is provided above, which were used in the authentication process that leverages the average distance travelled within the environment. The first pair of features are related to the location of the user and are represented in degrees in terms of latitude and longitude with respect to the earth coordinates. The location is used to combine the distance the user has travelled with the particular environment in which the user is placed in. Following the next feature is the average distance the user is travelling within the environment. The last feature is the travel state of the user, meaning if the user is stationary or is moving. These features were all combined through a machine-learning model in order to estimate if the identity of the user.

4.3 Behavioural cues extraction

This section described the extraction of the different behavioural cues and the behaviour inference with respect to user authentication. As indicated in the Description of Work, the BehavAuth will leverage multiple behavioural cues in order to learn the behavioural patterns of the users and based on these patterns to identify them to the TagItSmart platform. The results of the inference of the different behavioural patterns are fused into a coherent authentication result that is propagated to the TagItSmart platform.

For the development of the machine-learning models that learn the behavioural patterns of the users and then authenticate them based on these patterns, the same development process was followed. The data collected from the sensors were converted in the suitable format to be able to train the behavioural model and then to be able to predict the authentication of the user. The system was evaluated on the StudentLife data and the evaluation process is described in detail in the D2.2 [51] deliverable. The BehavAuth library has been developed in a sense to monitor the behaviour of the user for one week and then initiate the authentication process based on the behavioural patterns that have been learned during this period. Each of the models developed were evaluated against multiple machine-learning techniques in order to understand the accuracy and robustness of each of the machine-learning approaches. The machine-learning technique that achieved the highest accuracy was considered as the final technique that will be used in the BehavAuth app.

In the following subsections, each of the authentication components that were developed are analysed in detail.

4.3.1 Social interaction based authentication

This first authentication component that was developed according to the Description of Work was the social interaction detection. In the previous Section, a social interaction detection mechanism was developed in order to allow the detection of social interactions using the Bluetooth received signal strength. The Bluetooth received signal strength was leveraged in order to estimate the interpersonal distance of the user and further to understand if people are in close proximity to interact or not.

Having understood the social interactions that the users participate in, the next step would be to create an authentication model based on which the user is authenticated by the social interaction patterns. The initial step towards the creation of the authentication model based on the social interaction patterns was to define and extract the features for the machine-learning model. The features that were considered for the machine-learning model are outlined in Table 4.1.

Table 4.1. Features used for the social interaction detection.

Features			
Time	MAC address	Class ID	Level

These features were fed into different machine-learning techniques as shown in the D2.2 [40]. The different machine-learning techniques were evaluated based on the StudentLife dataset for a 10-week period. The Decision Tree Classifier showed the highest accuracy and was selected as the machine-learning technique to train the authentication model. The outcome of the trained model is the inference if the user is the actual owner of the device or if it is an imposter. The result of the model is forwarded to the Authentication Inference component to perform the final layer of the decision making for the BehavAuth app.

4.3.2 Location-based authentication

This subsection provides a detailed analysis of the authentication module that was developed for the BehavAuth project, which leverages the location the user visits in order to identify the user. Users visit different location and have certain mobility patterns during their daily lives. This means that location by itself can provide valuable information about the identity of the user and could enable the secure authentication process.

The majority of today's smartphone devices have integrated a location sensor such as GPS. Modern mobile operating systems provide for free intelligent mechanisms that estimate the location of the user not only by using the power-hungry GPS sensor but also through using the WiFi and cellular interfaces. Mobile operating systems vendors have developed intelligent location fusion techniques that leverage all three interfaces to provide accuracy and less power consuming location estimations. These virtual sensors, in addition to the location in earth coordinates, provide multiple types of information such as the bearing, altitude but also the instantaneous speed of the user at the moment of the measurement. Considering the data provided by the Android API for the location sensor, the feature definition and extraction process was initiated.

Table 4.2. Features for the location-based authentication component.

Features	
Time	Provider
Network type	Accuracy
Latitude	Longitude
Bearing	Altitude
Speed	TravelState

Considering the above defined features, multiple machine-learning models were trained as defined in D2.2 in order to understand the accuracy of different machine-learning techniques. Evaluation of the different machine-learning models provided an indication of the performance of the different approaches. The evaluation was performed based on the StudentLife data, where 10 users were monitored over 10-week period. Following the evaluation, the Decision Tree Classifier performed best among the other machine-learning techniques. Thus, the

Decision Tree Classifier was selected as the machine-learning technique to train the location patterns model. The result of the machine-learning model for the user authentication based on the location patterns was forwarded to the Authentication Inference in order to perform the final decision about the user authentication.

4.3.3 Speed within the environment

The next authentication component that was developed was focused on the speed of the user within the environment and the patterns observed in order to authenticate the user. People may visit similar locations in their daily routine. However, they move in the environment with different speed, for example one person is more active than another person or even has different speed during the different movements. The speed patterns of the users could provide valuable information to the authentication process.

Table 4.3. Features for the speed-based authentication component.

Features	
Latitude	Longitude
Speed	TravelState

In order to conclude on the machine-learning technique to be used, different machine-learning models were developed and evaluated based on the StudentlLife dataset. The authentication models that were developed during the BehavAuth project, were evaluated in order to understand the performance of the different models, and select the one with the highest performance. Following the evaluation in the D2.2, the Decision Tree Classifier was selected as it achieved the highest performance. The result of the authentication model is then forwarded to the Authentication Inference component that performs the last decision making regarding the authentication process of the BehavAuth tool.

4.3.4 Distance within the environment

This section describes the authentication model developed for the BehavAuth project that leverages the average distance the user is moving in the environment in order to authenticate the users. People in their daily lives move in a different way and also move for different distances. Some people are more active than others and some people are less active than others. For that reason, the distance that a user is moving within the environment could enable the authentication of the users based on their movement patterns.

Table 4.4. Features for the distance-based authentication component.

Features	
Latitude	Longitude
Distance	TravelState

For the training and development of the machine-learning model multiple approaches were followed. In order to validate the performance of different machine-learning approaches, various machine-learning models were trained. Each of the models was evaluated through the StudentLife dataset, using 10 users over a period of 10-weeks. The Decision Tree Classifier was selected as it outperformed the rest of the machine-learning models and it achieved the highest accuracy. After training the machine-learning model with the Decision Tree Classifier, the result of the model is forwarded to the Authentication Inference.

4.4 Authentication inference

The authentication inference component is focused on fusing the authentication results of each of the authentication components and providing a unique decision about the authenticity of the user.

The process of authentication inference is performed by leveraging a multifactor and seamless voting-based decision making mechanism that fuses the authentication results of the different authentication mechanism that were described above. The decision making fusion mechanism takes into account the accuracy of each of the authentication components i.e. social interaction detection, location, speed and distance within the environment, pattern recognition components and based on the result each module provided and the accuracy of the estimation that was provided through the evaluation of each of the components (See D2.2 [40]), estimates the result of the authentication.

Each of the authentication components is characterised by a unique weight that ensures each module has different effect in the authentication inference process. This weight is calculated through the accuracy of each of the authentication components. The accuracy of the authentication components was extracted from the trials that were performed during a 10-week period and using 10 participants. The results of this evaluation are detailed in D2.2. By providing a weighted mechanism and combining this with a voting system, the robustness of the authentication mechanism is increasing by considering multiple factors in the authentication process as well as taking into account the accuracy of each of the authentication components that are included in the inference process.

4.5 Implementation

The system components that were described in the above subsections were implemented into a mobile app, that performs data collection of the required information from the sensors of the off-shelf mobile phones and then infers about the authenticity of the users based on their behavioural cues. The result of the authentication process is forwarded to the backend server to inform the TagItSmart platform if the user is the device owner or if someone else is using the device.

The mobile app was implemented through the Android software development kit and focuses on the Android platform. This is because, Android constitutes the most widely used mobile phone platform and given its open source nature provide great abilities to the developer to implement sophisticated mobile apps and to perform highly computational and complex procedure in an easy manner.

4.5.1 Android permissions

The Android permissions required by the mobile app to collect the data from various sensors of the device mentioned in the above sections are defined below:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

A brief explanation of the android permissions required is defined below. For more detailed analysis the reader is referred to the D2.4 Ethical review [42], where the required permissions are described in detail.

- **ACCESS_FINE_LOCATION**. This is required for the app to be able to access the location data from sensors such as the GPS.
- **ACCESS_COARSE_LOCATION**. This is required for the app to be able to access the location data from sensors such as the WiFi and cellular.
- **BLUETOOTH**. This is required for the app to be able to access the Bluetooth data from the particular interface.
- **BLUETOOTH_ADMIN**. This is required for the app to be able to access the Bluetooth data from the particular interface.

4.5.2 Android Software Development Kit

The mobile app is targeting particular Android Software Development Kit and provides support to a certain range of operating system versions which can be viewed in the Table below:

Table 4.5. Android Software Development Kit versions.

Minimum SDK	21
Maximum SDK	Latest (27)
Targeted SDK	27
Build tools	25.0.3

4.5.3 Architecture

The mobile app is designed using modern software design pattern focused on mobile devices. In particular, a Model-View-Controller architecture was selected for the implementation of the mobile app, in order to enable the maintenance and extendibility of the mobile app.

The app has been designed as a library in order to allow third-party applications to incorporate the BehavAuth app in their own applications and use it as an authentication module in their apps. Together with the library, a sample code dedicated to the TagItSmart project was developed in order to enable the integration of the mobile app with the TagItSmart ecosystem. The accompany app that hosts the BehavAuth library receives the authentication result from the BehavAuth mobile app and forwards this to the Polling Manager app that is described in the next section. The Polling Manager app is responsible to collect the authentication results from the various authentication components and to forward the outcome to the TagItSmart backend server.

The library is based on an Android Service that coordinates the data collection and inference process. The data collection process is event based, so when a new measurement from the required data is received by the device, this measurement is forwarded to the Service, removing any requirement for constantly polling the sensors and increasing the computational and power consumption burden. The inference process is then activated once a new measurement is received and it is taking place on a separate thread. This enables the Service that performs the coordination of the data collection and inference, not to block and to be able to serve multiple events simultaneously.

When the inference component provides a result, the result is forwarded to the MainActivity of the TagItSmart accompany app and then further to the Polling Manager app.

4.6 Authentication in simple words

This subsection focuses on providing a brief description of the process followed during the training and authentication, in order for an end-user to get a better understanding of the overall authentication process. First an overall description of the process followed with the different components of the BehavAuth and TagItSmart systems will be provided. Then the BehavAuth authentication process will be summarises in order to provide a deeper explanation of the authentication process.

The three main components of the overall system are a) the TagItSmart platform (including the User Repository), b) the Polling Manager app and the c) the BehavAuth mobile app. The TagItSmart is the overall third-party cloud platform that includes the User Repository; a user management system. The Polling Manager is a mobile app that is co-hosted with the BehavAuth app on the same mobile phone and is responsible to merge the authentication results from the different components and forward it to the User Repository at the TagItSmart platform. The BehavAuth app estimates the authentication result and forwards it to the Polling Manager. This constitutes the overall information flow among the three aforementioned components.

Next, the BehavAuth app will be described in order for the reader to understand the usage of the mobile authentication app. First the end-user installs the BehavAuth app by accessing the publicly available repository https://bitbucket.org/quadible/behavauth. Next, when the user launches the app, a wizard will come up describing to the user what type of personal data will be collected, for which reason and also will ask the permission of the user to collected and process the data. After providing/revoking the appropriate permissions, the user will be directed to the main screen. On the upper right corner, the user is able to provide/revoke any data collection and processing permission as well as to delete any collected data (training data and machine-learning models).

In order to perform training and authentication, the app requires to turn on the Bluetooth and Location interfaces to enable the data collection process. Also, it requires that the Write to storage permission is enabled in order to store the training data and the machine-learning models at the SD card.

For the social interaction detection, the users just need to install the BehavAuth app. Then no further action is required as the app opportunistically collects the Bluetooth information and calculates the interpersonal distance with other Bluetooth devices. The only requirement is that the other users have enabled the Bluetooth interface and enable the discoverability. This is because, if the Bluetooth interface is not enabled at the devices in vicinity and set to discoverable, the smartphone with the BehavAuth app is not able to discover the other devices in vicinity. Lastly, the BehavAuth does not require any special devices to perform social interaction detection, just there is a need to enable the Bluetooth interface and set it to discoverable.

The next step to be followed is to train the machine-learning models, in order to recognise the user. The main screen provides the appropriate interface to initiate/end the training process for the actual user and the imposter. Ideally the machine-learning models should be trained for at least one week, in order to be able to consider the weekly patterns of the users. By clicking the “Start” button for training the user’s machine-learning models, given that the appropriate permission for data collection and processing are provided, the app starts to collect Bluetooth and Location data. From the Bluetooth interface, it collects the timestamp, the MAC address,

the RSSI and the class level of any detected nearby devices. From the Location interface it collects timestamp, provider, network type, accuracy, latitude, longitude, altitude, bearing, speed and travelstate. The data are labelled (user) and stored in a CSV file. Then, the data collection for the training set is finished and stopped by the user. The mobile app provides the ability to the user to train also imposters machine-learning models with similar process. When the data collection process is finished, a button “Train models” allows the user to train the models.

Until the user hasn’t trained the machine-learning models, the authentication button at the lower part of the screen are disabled. Once the user trains the machine-learning models, then the authentication buttons are active that allow the user to start/stop the authentication process. It should be noted that the user can collect more data if required at a later stage and can re-train the machine-learning models. Once the start authentication button is pressed, the app initiates the authentication process. The authentication process waits until it receives any social interaction instance (Bluetooth event) and then based on the last known location it performs a classification/authentication. The result of the authentication is sent to the Polling Manager. For every new Bluetooth event, the BehavAuth performs an authentication. A discovery for new Bluetooth device is performed every 10 seconds and also, the location updates are taking place every 10 seconds.

Section 5 - TagItSmart polling manager

This section describes the TagItSmart Polling Manager app, which is an accompany app that enables the integration of the different authentication components such as BehavAuth, location based authentication (provided by the University of Surrey) and keystroke based authentication (provided by the University of Padova).

5.1 Mobile app usage

The Polling Manager operates mainly in an opportunistic manner. This means that the users just need to install the mobile app. After the installation process, the mobile app operates on its own without the needs of the users input. From the point that the app will be installed, it will monitor the communication bus about any incoming messages and in case it receives any messages it will forward them to the authentication voting mechanism and the result of it will be propagated to the User Repository of the TagItSmart platform.

5.2 Android permissions

The Android permissions are required by the mobile app to collect the data from various sensors of the device mentioned in the above sections are defined below:

```
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.INTERNET" />
```

A brief explanation of the android permission required is defined below.

- **RECEIVE_BOOT_COMPLETED**. This is required for the app to be able to understand when the device has booted (started), in order to start to monitor the communication bus of other authentication components for the TagItSmart system.
- **INTERNET**. This is required for the app to be able to communicate with the TagItSmart platform and in particular with the User Repository component.

5.3 Android Software Development Kit

The mobile app is targeting particular Android Software Development Kit and provides support to a certain range of operating system versions which can be viewed in the Table below:

Table 5.1. Android Software Development Kit versions.

Minimum SDK	21
Maximum SDK	Latest (27)
Targeted SDK	27
Build tools	25.0.3

5.4 Architecture

The mobile app is operating basically as a fusion mechanism for the different authentication streams that are provided in the TagItSmart project and also as the communication mean

between the mobile device and the TagItSmart backend platform. The Polling Manager is a mobile app that is initiated once the mobile device is turned on. A broadcast receiver gets a message that the device has been booted and then a background service is started that performs the listening for new messages process and the redirection of information.

Each of the authentication components of the TagItSmart platform performs inferences at different periods of time and publishes the result through a common communication bus internally on the mobile device. The Polling Manager is listening on the particular communication bus. Once an authentication result is received on the communication bus, the Polling Manager initiates a voting mechanism in order to decide based on the messages received, which is the result of the authentication process. Once the Polling Manager completes the voting mechanism the result is forwarded to the TagItSmart User Repository component.

5.4.1 Voting mechanism

The voting mechanism constitutes a component of the Polling Manager with the sole purpose to merge the outcomes of the different authentication components of the TagItSmart platform. The different components of the TagItSmart platform are BehavAuth, the location authentication from the University of Surrey and the keystroke authentication component from the University of Padova. Each of these components are part of the TagItSmart platform and provide individual estimation about the authentication of the user. However, there is a need to merge these technological components into a coherent authentication tool, that will provide the authentication decision to the TagItSmart platform in order to allow the mobile app to communicate with the backend server.

State-of-the-art techniques have proposed many different ways in order to combine the same type of information from various streams into one coherent stream and decision. One of the most commonly used approaches is to leverage a voting system. Each of the components provides a vote depending on its “opinion/decision” about the authentication of the user. These votes are combined together and the most popular decision is provided as outcome of the Polling Manager. The outcome is then forwarded to the User Repository of the TagItSmart platform. The voting mechanism constitutes a democratic and more robust solution as it considers the “voice” of most of the authentication mechanisms in order to decide about the result of the authentication process. As the result is binary and the votes are three (as the number of the authentication systems), then there will always be a decision about which is the most popular decision among the authentication modules.

5.4.2 Internal communication bus

The internal communication bus is shaped using broadcast intents internally in the Android application space. This process is performed through the Android Intents, which are broadcasted to the application space of the device. Every installed mobile app is able to listen to the broadcasted intent as long as they know the ACTION ID that is used to broadcast the message. The Polling Manager is registering to the ACTION ID that is defined in advantage and will be used by all TagItSmart authentication components including the Polling Manager itself. Once the mobile device is started the Polling Manager is registering to the ACTION ID.

```
<action android:name="eu.tagitsmart.auth.pollingmanager" />
```

Then, each of the authentication components such as BehavAuth, location based authentication from University of Surrey and keystroke authentication from University of

Padova, will publish their authentication results to the particular ACTION ID. Once this is performed the Polling manager will receive a BroadcastMessage that will contain the information required. The information will be forwarded to the voting mechanism and further the result will be propagated to the TagItSmart platform through the REST communication.

5.4.3 Communication with TagItSmart platform

The Polling Manager, after receiving the various results of the authentication of the different authentication components that are included in the TagItSmart platform and having fused them through the voting mechanism described above, it communicates the result to the TagItSmart platform. The communication of the result of the TagItSmart platform is based on the 3G/Wifi interface and is performed through Internet. For the communication, a particular component was developed in the Polling Manager app that performs HTTP POST to the User Repository of the TagItSmart platform.

An example of the query performed to the TagItSmart User Repository is provided below, using the curl tool. This example performs an authorization update to bind the FCScannerService to an “UnAuthorized” user. The correct credentials need to be sent to be able to access the User Repository, along with a matching deviceID for the user related to the credentials

```
curl -v -k -H 'Content-Type: application/json' https://XXX.XXX.XXX.XXX:XXXXX/updateauth
-d'{"email":"xxxxx@xxxxx.xxx", "password":"XXXXXXX", "deviceID":
"XX:XX:XX:XX:XX:XX", "updateUser": "UnAuthorized"}'
*   Trying 127.0.0.1...
* Connected to localhost (XXX.XXX.XXX.XXX) port XXXXX (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 592 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_RSA_AES_128_GCM_SHA256
*   server certificate verification SKIPPED
*   server certificate status verification SKIPPED
* error fetching CN from cert:The requested data were not available.
*   common name:  (does not match 'localhost')
*   server certificate expiration date OK
*   server certificate activation date OK
*   certificate public key: RSA
*   certificate version: #1
*   subject: C=GB,ST=England,L=Guildford,O=University of Surrey
*   start date: Fri, 07 Jul 2017 14:08:00 GMT
*   expire date: Mon, 21 Nov 2044 14:08:00 GMT
*   issuer: C=GB,ST=England,L=Guildford,O=University of Surrey
*   compression: NULL
* ALPN, server accepted to use http/1.1
> POST /updateauth HTTP/1.1
> Host: localhost:4322
> User-Agent: curl/7.47.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 118
>
* upload completely sent off: 118 out of 118 bytes
< HTTP/1.1 200 OK
```

```
< X-Powered-By: Express
< Content-Type: application/json; charset=utf-8
< Content-Length: 57
< ETag: W/"39-tI/EzcH7tx58jb8UCl5+4Hy0a/E"
< Date: Tue, 27 Feb 2018 09:48:42 GMT
< Connection: keep-alive
<
* Connection #0 to host localhost left intact
{"userUpdated":"UnAuthorized","updateTime":1519724922399}
```

Section 6 - Design based on the functional requirements

This section describes how the functional requirements defined at the initial sections of this document were matched with the overall design and architecture of the BehavAuth system. The functional requirements were derived from the dissemination activities that were initiated from the beginning of the project. The dissemination activities included engagement of the potential stakeholders such as psychologists, privacy experts, end-users and potential early adopters of the platform. Following, Table 6.1 provides a mapping among the functional requirements defined in initial sections of this document with the design of the BehavAuth tool.

Table 6.1. Description of how the BehavAuth design matches the functional requirements.

Category	REQ_ID	Description	BehavAuth design
Security	F01	The tool must provide a secure communication channel between the cloud and the mobile devices.	The TagItSmart platform provides a secure and encrypted HTTPS channel that enables the communication between the Polling Manager and the TagItSmart platform. The BehavAuth app communicates with the Polling Manager internally on the device and then the Polling Manager through the secure transmission channel it propagates the information to the User Repository on the cloud.
	F02	The platform must provide a secure communication channel between the cloud and the TagItSmart platform.	The cloud and the TagItSmart platform are co-hosted in same sandboxed environment thus there is no potential of breach, unless the whole TagItSmart platform environment is compromised.
	F03	The platform must ensure the integrity of all the communications.	At the moment, in order to communicate with the TagItSmart platform, the BehavAuth app needs to send some personal data including the email, password, username and device id. These are the data that are required by the Polling Manager and further by the TagItSmart platform.
Privacy	F04	The platform must should store a pseudonym for each registered user.	The users' username is considered as the pseudonym of each of the registered users to meet the requirements of the TagItSmart platform. In the future design, a pseudonym will be generated.
	F05	Users' data should be processed only when explicit	The first time the user installs the BehavAuth mobile, a specific wizard was implemented that takes the user step by step by each type of data that

		consent has been given by the users.	is required to be collected and the reason why that type of data is being collected. Then, the user can access the collected data as they are stored locally on the device. On the upper right corner of the screen an options menu has been developed, were the user is able to withdraw the consent at any point and stop any data collection or data processing.
	F06	Users should have the ability to stop the data collection and processing at any time.	On the upper right corner of the main screen of the BehavAuth app, an options menu has been developed. There the user is able to revoke the consent and stop any data collection and processing at any time.
	F07	Only data required for the authentication mechanism should be collected.	The BehavAuth app collects and stores Bluetooth and location data. In order to be able to perform authentication at the User Repository of the TagItSmart platform, there is a need to store also the username, email, password and device id.
	F08	Users should have the ability to delete the collected data and any extracted knowledge.	On the upper right corner of the main screen of the BehavAuth app, an options menu has been developed. There the user, by clicking on the "Delete all data" button, he/she will delete all the collected data and any extracted knowledge i.e. machine-learning models.
ID	F10	Only the pseudonym of each user will be stored, followed by the behavioural data.	The TagItSmart platform currently does not offer the ability to anonymise any personal data. Also, the behavioural data are not stored after the authentication process. In the future design, described in the following section this feature will be implemented.
	F11	The ID of each user will be anonymised.	Currently the TagItSmart platform does not offer the ability of data anonymization. Thus, in order for the BehavAuth app to be able to authenticate at the TagItSmart platform, the users' data are stored as is. In the future design, described in the following section this feature will be implemented.
	F12	The platform should allow the	The registration of new users and deletion of current users is performed

		registration and deletion of users.	through the User Repository of the TagItSmart platform.
Authentication	F13	The authentication mechanism should user-friendly.	The authentication process was designed in a user-friendly manner. This means that from the moment the user enables the authentication process, there is no need for user interaction. The user just continues with daily life routines and does not need to interact at all with the app, as all the data collection and processing is taking place on the background.
	F14	The authentication mechanism should allow developers to easily integrate with TagItSmart and other third-party platforms.	The BehavAuth app was designed in an easy and developer-friendly manner to enable the integration with third party platforms. The initial integration was particularly created for the TagItSmart use case. As shown in the future design, the platform will be developed further to ensure easiness of integration with other third-party platforms.
	F15	The behavioural patterns should accurately identify and verify the users.	The trials and validation deliverable showed that the different machine-learning models were able to accurately authenticate the users by learning various behavioural patterns and achieving even over 98% accuracy.

Section 7 - Beyond TagItSmart - Future design

This section provides the future design of the BehavAuth system by taking into consideration the uses cases and requirements detailed in the D1.1 [21]. The BehavAuth architecture and system design provided in the above sections was created to meet the requirements of the TagItSmart platform. As defined in the DoW, Quadible plans to further exploit the BehavAuth system by commercialising it and providing the solution as a Software-as-a-Service solution. In order to fit that need, there is a need to refine and expand the aforementioned design. The future design of the system is described in the subsections below. The section is structured as follows; the first subsection provides an overview of the design of the system followed by a more detailed description of each of the core components of the system.

7.1 Overall design of the system

This section provides an overview of the future design of the BehavAuth system. BehavAuth focuses on authenticating users for third-party platforms. For the design of the authentication system, state-of-the-art architecture design patterns were considered [51] from IEEE Centre for Secure Design. The authentication process is based on understanding and learning the behavioural patterns of the users by leveraging data extracted from smartphones, such as sensor and device usage data. The data are collected from a smartphone app and are forwarded to the third-party platform. Then the third-party platform forwards the collected data, including a pseudo-ID, to the BehavAuth platform. The BehavAuth platform will, as a first step, create a record of the pseudo-ID and initiate the learning process for the authentication models. Once the behavioural models have learned the patterns of the user, the same path will be followed for the biometric data. However, in this case, the authentication service will be also executed. Initially, it will perform authentication followed by learning, in order to adapt to the contextual changes of the person.

The BehavAuth constitutes of:

- **The platform**, where the processing and the authentication procedure takes place.
- **The mobile app/library**, where the data are being collected in order to send them to the back-end platform.

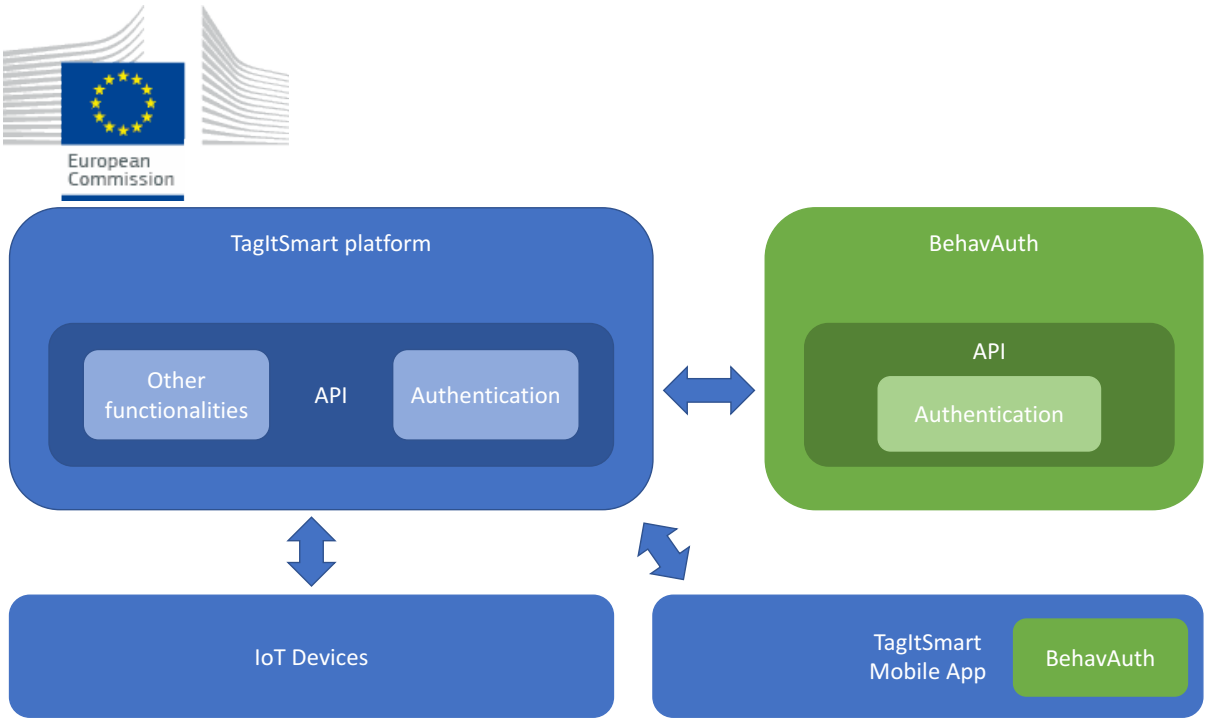


Figure 9. BehavAuth overall architecture.

Figure 9 shows the overall design of the BehavAuth system including the scenario, where the third-party platform is the system that required an authentication mechanism based on behavioural patterns. The overall system consists of the third-party and the BehavAuth platforms, the IoT devices that are coordinated by the third-party platform and mobile app (which includes a BehavAuth component). The third-party mobile app will integrate the BehavAuth library. The latter will constitute the component which collects the behavioural data from the smartphone, encrypts them and then the third-party, through the secure communication channel it has established, it will propagate the behavioural data to the third-party platform and further to the BehavAuth platform, in order to perform authentication and inform the third-party platform that this is the actual user and not some imposter.

7.2 BehavAuth platform

The first core component of the BehavAuth system is the cloud platform. This component of the system implements the machine-learning and artificial intelligence models using sensor and device usage data from users' smartphones. The models are trained initially and then adapt based on the contextual information in which the user is. To ensure the validity of the user at the initialisation process, the BehavAuth platform communicates only with the third-party platform, which has previously established a secure communication with the mobile app. Authentication tokens are used between the third-party platform and mobile app, to ensure the secure communication and validation among the two components. Similar secure communication is established between the two platforms.

The main functionalities that the BehavAuth platform will provide are:

- **User registration**. Users are registered at the BehavAuth platform through the third-party platform. The latter uses a pseudonym, an application key and the behavioural data to register a user at the BehavAuth platform. During this registration, an initialisation process is performed, the appropriate entities are created, and the training process of the various machine-learning models is initiated.

- **User authentication**. Given that a user has been registered at the BehavAuth platform, an authentication service will be exposed, that will allow users to authenticate to the third-party platform. The various machine-learning models will receive the behavioural data of the user and will perform validation. A response will be sent to the third-party platform indicating the result i.e. if the user is an imposter or not. In case this is the actual user, the biometric data will be fed to the machine-learning models in order to implement the adaptation based on the contextual data.

7.3 BehavAuth mobile app/library

The second core component of the BehavAuth system is the mobile app/library. BehavAuth performs authentication by collecting data from smartphone's sensors, as well as, the device usage in order to identify behavioural patterns which allow the unique authentication of the device owner.

The main functionalities of the BehavAuth library will be:

- **Data collection**. The library will implement the data collection process regarding all the predefined biometrics. It should be noted that the data that will be collected are based on the user's permission. At any point, the user will be able to suspend the data collection process and remove any collected data at both device level and back-end platform.

- **Data encryption**. The library will include a public encryption key that will allow the app to encrypt inside the library all the collected data and then forward them to the third-party app and further to the third-party platform.

Section 8 - Conclusion

This deliverable described the final design of the BehavAuth tool based on the use cases and the requirement analysis extracted and based on the needs of the TagItSmart platform. At the end of the document, the future envisioned design of the system is provided given the foreseen continuation of the project.

At the beginning of the document, the use case and requirement analysis is presented to the reader. The use case and requirement analysis is refined since the D1.1. Following the requirements that were refined from the engagement with the potential stakeholders.

Next, the refined system design is presented to the reader. The system design was introduced in the D1.1 and given the interactions with the TagItSmart consortium, the design was refined. The refinement occurred in order to enable the BehavAuth to integrate with the TagItSmart platform. The system design is split into various core components including the different authentication components of the TagItSmart platform, the BehavAuth authentication app, the Polling Manager app and the backend platform. The different authentication components refer to the location-based authentication of the University of Surrey, the keystroke-based authentication of the University of Padova and the BehavAuth.

Following, the social interaction detection mechanism was introduced. State-of-the-art techniques for social interaction detection were analysed for wearables and smartphones. Then a novel approach was proposed that leveraged the Bluetooth received signal strength to transform the signal to interpersonal distance and further to social interaction existence. The approach was based on an extensive data collection and feature extraction process that allowed the development of a novel machine-learning based model for social interaction detection.

Next, the mobile authentication app/library is explained in detail. Initially the overall design of the authenticator app was analysed, explaining the interactions with the different components of the overall system as well as the interactions among the internal components of the system. The data collection as well as the data processing were analysed in order to provide the appropriate information regarding the dataset used to develop the machine-learning models. In addition, the data used at the dataset were outlined. The behavioural cues that were extracted were described as well as the different authentication components. The different authentication components that were introduced were based on social interaction patterns, on location patterns, speed and distance patterns within the environment. Lastly, the authentication inference component that combines the authentication components into a coherent authentication decision.

Another core component of the TagItSmart authentication process that was analysed was the Polling Manager. This mobile app is responsible for combining the authentication decision of the various authentication components that operate together in the TagItSmart ecosystem. Among these different authentication components are location-based authentication of the University of Surrey, the keystroke-based authentication of the University of Padova and the BehavAuth.

Finally, the future design of the BehavAuth tool architecture was provided, taking into account the requirements extracted from the early adopters of the platform. BehavAuth collaborators concluded on the particular design based on the requirements derived from existing state-of-the-art authentication systems and the stakeholder engagement. This constitutes the future design of the BehavAuth architecture as Quadible plans to continue the development of the

BehavAuth project and provide an Authentication-as-a-Service solution towards third-party companies.

Section 9 - References

1. TagItSmart EU project – www.tagitsmart.eu
2. N. Palaghias, S. A. Hoseinitabatabaei, M. Nati, A. Gluhak and K. Moessner, "Accurate detection of real-world social interactions with smartphones," 2015 IEEE International Conference on Communications (ICC), London, 2015, pp. 579-585.
3. Niklas Palaghias, Seyed Amir Hoseinitabatabaei, Michele Nati, Alexander Gluhak, and Klaus Moessner. 2016. A Survey on Mobile Social Signal Processing. *ACM Comput. Surv.* 48, 4, Article 57 (March 2016), 52 pages.
4. T. Choudhury and A. Pentland. Sensing and modeling human net- works using the sociometer. Seventh IEEE International Symposium on Wearable Computers, 2003. Proceedings., pages 216–222, 2003.
5. G. Groh, A. Lehmann, J. Reimers, M. R. Friess, and L. Schwarz. De- tecting Social Situations from Interaction Geometry. 2010 IEEE Sec- ond International Conference on Social Computing, pages 1–8, Aug. 2010.
6. J. Stehle´, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M.Quaggiotto,W.VandenBroeck,C.Re´gis,B.Lina,andP.Van- hems. High-resolution measurements of face-to-face contact patterns in a primary school. PloS one, 6(8):e23176, Jan. 2011.
7. A. Antoniou, E. Theodoridis, I. Chatzigiannakis, and G. Mylonas. Monitoring physical space using mobile phones for inferring social and contextual interactions. 2011 IEEE SENSORS Proceedings, pages 1616–1619, Oct. 2011.
8. E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: The design, implementation and evaluation of the cenceme application. In Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems, SenSys '08, pages 337–350, New York, NY, USA, 2008. ACM.
9. K. A. Li, T. Y. Sohn, S. Huang, and W. G. Griswold. Peopletones: A system for the detection and notification of buddy proximity on mobile phones. In Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services, MobiSys '08, pages 160–173, New York, NY, USA, 2008. ACM.
10. N. Eagle and A. Pentland. Social serendipity: mobilizing social soft- ware. Pervasive Computing, IEEE, 4(2):28–34, 2005.
11. H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell. Sound- sense: Scalable sound sensing for people-centric applications on mobile phones. In Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services, MobiSys '09, pages 165– 178, New York, NY, USA, 2009. ACM.
12. I. Carreras, A. Matic, P. Saar, and V. Osmani. Comm2sense: De- tecting proximity through smartphones. In Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE In- ternational Conference on, pages 253–258, 2012.
13. S. Liu, Y. Jiang, and A. Striegel. Face-to-Face Proximity Estima- tion Using Bluetooth

On Smartphones. IEEE Transactions on Mobile Computing, pages 1–1, 2013.

14. W. Hu, G. Cao, S. V. Krishanamurthy, and P. Mohapatra. Mobility- assisted energy-aware user contact detection in mobile social net- works. 2013 IEEE 33rd International Conference on Distributed Computing Systems, 0:155–164, 2013.

15. Y. Wang and L. Cuthbert. Bluetooth positioning using RSSI and tri- angulation methods. 2013 IEEE 10th Consumer Communications and Networking Conference (CCNC), pages 837–842, Jan. 2013.

16. A. Ghose, C. Bhaumik, and T. Chakravarty. Blueeye: A system for proximity detection using bluetooth on mobile phones. In Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication, UbiComp '13 Adjunct, pages 1135–1142, New York, NY, USA, 2013. ACM.

17. N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner. Virtual compass: Relative positioning to sense mobile social interactions. In Proceedings of the 8th International Conference on Pervasive Computing, Pervasive'10, pages 1–21, Berlin, Heidelberg, 2010. Springer-Verlag.

18. A. Matic, V. Osmani, A. Maxhuni, and O. Mayora. Multi-Modal Mo- bile Sensing of Social Interactions. Proceedings of the 6th International Conference on Pervasive Computing Technologies for Health- care, pages 105–114, 2012.

19. E. Hall. The hidden dimension. Anchor Books, New York, NY, 1969.

20. A. Kendon. Conducting interaction: Patterns of behavior in focused encounters. CUP Archive, vol 7 edition, 1990.

21. BehavAuth D1.1 Initial design based on the use case and requirement analysis

22. H. Liu and R. Setiono. A probabilistic approach to feature selection - a filter solution. In 13th International Conference on Machine Learning, pages 319–327, 1996.

23. G. I. Webb. Multiboosting: A technique for combining boosting and wagging. Machine Learning, Vol.40(No.2), 2000.

24. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

25. R.Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997. Special issue on relevance.

26. Wang, Rui, Fanglin Chen, Zhenyu Chen, Tianxing Li, Gabriella Harari, Stefanie Tignor, Xia Zhou, Dror Ben-Zeev, and Andrew T. Campbell. "StudentLife: Assessing Mental Health, Academic Performance and Behavioral Trends of College Students using Smartphones." In Proceedings of the ACM Conference on Ubiquitous Computing. 2014.
27. Inman, James *(1835) [1821].* Navigation and Nautical Astronomy: For the Use of British Seamen *(3 ed.). London, UK: W. Woodward, C. & J. Rivington*. Retrieved 2015-11-09.
28. Haversine formula error - https://geopy.readthedocs.io/en/1.10.0/

29. Vincenty T. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. Survey review. 1975 Apr 1;23(176):88-93.
30. Seber, G.A. and Lee, A.J., 2012. Linear regression analysis (Vol. 329). John Wiley & Sons.
31. Mika, S., Ratsch, G., Weston, J., Scholkopf, B. and Mullers, K.R., 1999, August. Fisher discriminant analysis with kernels. In Neural networks for signal processing IX, 1999. Proceedings of the 1999 IEEE signal processing society workshop. (pp. 41-48). Ieee.
32. Peterson, L.E., 2009. K-nearest neighbor. Scholarpedia, 4(2), p.1883.
33. Safavian, S.R. and Landgrebe, D., 1991. A survey of decision tree classifier methodology. IEEE transactions on systems, man, and cybernetics, 21(3), pp.660-674.
34. John, G.H. and Langley, P., 1995, August. Estimating continuous distributions in Bayesian classifiers. In Proceedings of the Eleventh conference on Uncertainty in artificial intelligence (pp. 338-345). Morgan Kaufmann Publishers Inc.
35. *Fawcett, Tom (2006).* "An Introduction to ROC Analysis" *(PDF). Pattern Recognition Letters*. 27 (8): 861–874. doi:10.1016/j.patrec.2005.10.010.
36. Powers, David M W (2011). "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation" (PDF). Journal of Machine Learning Technologies. 2 (1): 37–63.
37. Hall, E.T., 1966. The hidden dimension.
38. TagItSmart D1.2 Initial TagItSmart! Reference Architecture and API description - http://www.tagitsmart.eu/D1.2.pdf
39. TagItSmart D4.2 Service composition platform description - http://www.tagitsmart.eu/D4.2.pdf
40. BehavAuth D2.2 Final trials and validation report.

41. Best practices for an authentication system by IEEE Center for Secure Design - https://cybersecurity.ieee.org/blog/2016/06/02/design-best-practices-for-an-authentication-system/
42. BehavAuth D2.4 Ethical review