Deliverable 2.2

ENSAFE system infrastructure architecture

Lead Partner:	Gaia
Authors:	Fredrik Knutson
Contributors:	UNIPR, SICS, MTD, GAIA
Date:	2016-09-08
Revision:	V1.0
Dissemination Level:	PUBLIC

Project Acronym:	ENSAFE
Project full title:	Elderly-oriented, Network-based Services Aimed at independent liFE
AAL project number:	AAL 2014-1-112
With support of:	

Content

1 Introduction

1.1 Document overview

This document describes the ENSAFE system infrastructure architecture

It describes:

- A general description of the system
- The physical architecture of the hardware on which runs the software
- The logical architecture of software, the layers and top-level components
- The justification of technical choices made

1.2 Abbreviations and Glossary

1.2.1 Abbreviations

Abbreviation	Word/phrase
AMQP	**A**dvanced **M**essage **Q**ueue **P**rotocol
API	**A**pplication **P**rogramming **I**nterface
BLOB	**B**inary **L**arge **Ob**ject
C2D	**C**loud **To D**evice
D2C	**D**evice **To C**loud
HTTP	**H**ypertext **T**ransfer **P**rotocol
IaaS	**I**nfrastructue **as a S**ervice
IoT	**I**nternet **of T**hings
MQTT	**M**essage **Q**ueue **T**elemetry **T**ransfer
PaaS	**P**latform **as a S**ervice
REST	**R**epresentational **S**tate **T**ransfer
SaaS	**S**oftware **as a S**ervice
SAS	**S**hared **A**ccess **S**ignatures

1.2.2 Glossary

Term	Definition
Azure	Microsoft **Azure** is a cloud computing platform created by Microsoft

1.3 References

Label	Reference
Azure IoT SDKs	https://github.com/Azure/azure-iot-sdks
Azure Event Hubs SDKs	https://github.com/Azure/azure-event-hubs
Azure Event Hub Documentation	https://azure.microsoft.com/en-us/documentation/services/event-hubs/
Azure IoT Hub Documentation	https://azure.microsoft.com/en-us/documentation/services/iot-hub/
Azure Storage Documentation	https://azure.microsoft.com/en-us/documentation/services/storage/
Azure Storage SDKs	https://github.com/Azure/azure-storage-android https://github.com/Azure/azure-storage-cpp https://github.com/Azure/azure-storage-ios https://github.com/Azure/azure-storage-java https://github.com/Azure/azure-storage-net https://github.com/Azure/azure-storage-node https://github.com/Azure/azure-storage-php https://github.com/Azure/azure-storage-python https://github.com/Azure/azure-storage-ruby
DropBox website	https://www.dropbox.com/about
Shared Access Signatures	https://azure.microsoft.com/sv-se/documentation/articles/storage-dotnet-shared-access-signature-part-1/
Swedish eHealth Agency website	https://www.ehalsomyndigheten.se/other-languages/english/

Püffell website	https://puffell.com/account/About

2 Architecture

2.1 *Principles*

A number of principles or design goals has been set for the architecture, namely:

- Scalability
  In principle the platform should be able to scale to support both the wide range of different flavours and actors within elderly healthcare in the EU but also to support a massive number of clients.

- Technical openness
  The architecture of the platform should be possible to integrate with different technical environments

- Commercial openness
  The architecture of the platform should support, and embrace third party suppliers and consumers

- Respect for the individual's wish and integrity

2.2 *Architecture overview*

The ENSAFE "system" consists of a number of different subsystems or components, integrated to support an information flow from sensors in the caretaker's vicinity via a data ingestion and distribution platform to backend services and then back to applications used by caretakers, informal caregivers or formal caregivers

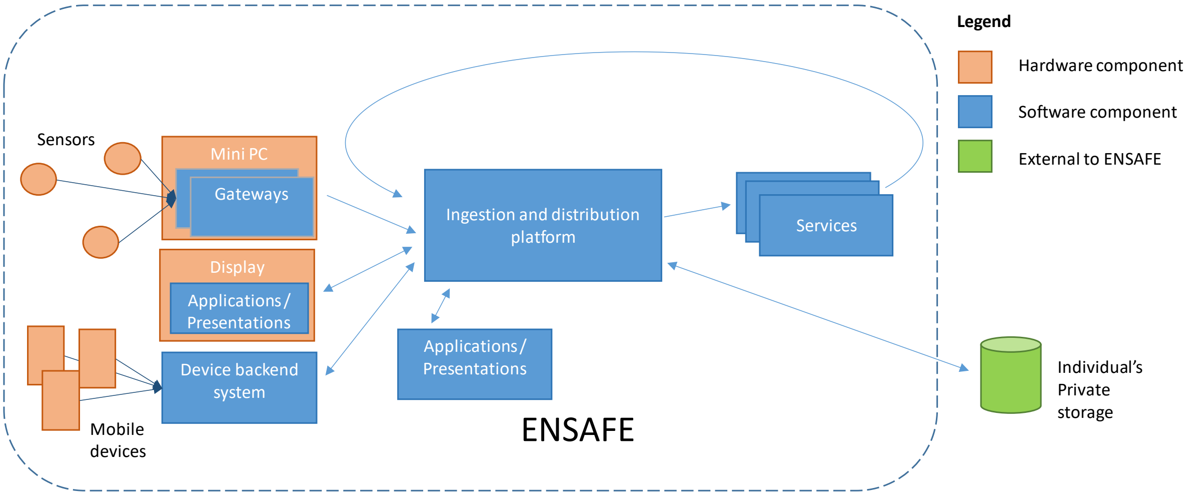

Figure 1 ENSAFE system overview

Figure 1 shows the information flow from the care recipient's vicinity to the left, to services and external storage to the right.

The main focus of this document is the architecture of the ENSAFE platform, named *Ingestion and distribution platform in* Figure 1.

2.3 Physical architecture overview

The physical architecture is composed of components residing either at the care recipient's home, the cloud (Microsoft Azure) or on premise with a partner or 3rd party. See Figure 2 which shows an alternative view of the system components in regards to where the components are physically deployed.

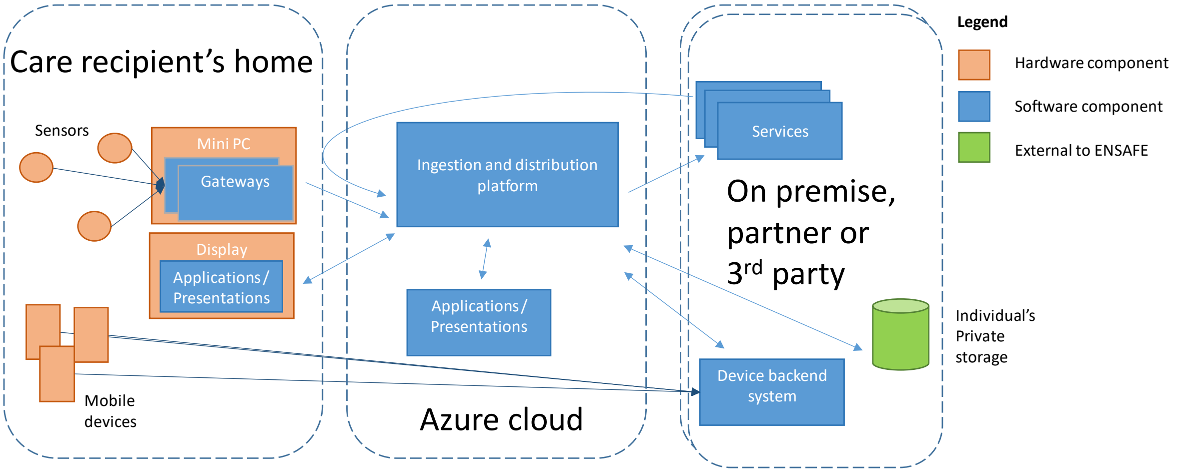

Figure 2 Physical deployment of ENSAFE components

2.3.1 Mini PC

The Mini PC in Figure 2 is a small computer where the Sensor gateway software executes.

The mini PC has attached adaptors for communicating with the sensors via low power protocols like BTE and ZigBee.

The gateway software will communicate with the ENSAFE platform via the event messaging protocol AMQP.

During the lifecycle of the ENSAFE project this will probably be small PC running a Linux OS.

2.3.2 Display

The Display in Figure 2 is a small computer with a touch enabled display for executing application(s) with input and output capabilities in the care recipient's vicinity. Input can result in messages sent to the ENSAFE platform via the AMQP protocol and output will be downloaded via the HTTP protocol.

A goal for future development of the ENSAFE system is to merge the hardware of the Mini PC and the Display. The benefits of this would be:

- Less hardware installed at each care recipient's home
- Less hardware to monitor and remotely manage
- Open space for 3rd party sensor providers to integrate and execute their gateway software

2.3.3 Azure cloud – Microsoft Azure

Microsoft Azure which is Microsoft's cloud offering is, from physical perspective, a structure of massive datacentres distributed globally in a number of regions. Two datacentres reside within the borders of EU, namely the North Europe region (Ireland) and the West Europe region (Netherlands).

Benefits of cloud computing include scalability, elasticity and no upfront investment, you only pay for what you use (service offering).

The offerings of Azure can be divided into three main categories – IaaS (Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service) where PaaS is the most relevant from an ENSAFE standpoint.

2.4 Logical architecture overview

From a logical point of view the architecture of the platform can be divided into a number of functional blocks:

- **Ingestion** of data
  This includes ingestion of both event data from sensors and from mobile devices but also 2nd generation data in the form a results from analysis services that have analysed the event data.
- **Real time analysis** of data
  The platform is able to perform real time analysis of the ingested event stream. This could be used for enriching the data with for instance mean value calculations or alerts at the detection of a value exceeding a threshold.
- **Distribution of data** to connected systems
  Distribution of data to different external systems (external from a platform perspective) is one of the main tasks of the platform.
- Support for customized and **individualized storage** of the care recipient's data
  The individual care recipient (or someone commissioned) should be in control of whether and where his/her data should be stored. The platform will support a number of different data storage options and will be open for more.

2.4.1 Ingestion

Ingestion of data includes both ingestion of 1st generation event data from sensors and mobile devices but also ingestion of 2nd generation result data from analysis services that has been applied to the event data.

Event data can either be submitted from a gateway in the home of the care recipient directly to the platform in a near real time manner (as in the case of event data from sensors) or via a service that collects data from multiple users and the transmits it to platform in a batch like manner (as in the case of the GoCiety GoLiveEngine). In the first scenario each gateway will have a unique identity (per user) that is used when authenticating with the platform, in the latter scenario the intermediate service will have a key used for authenticating with platform but data will be transmitted for multiple users.

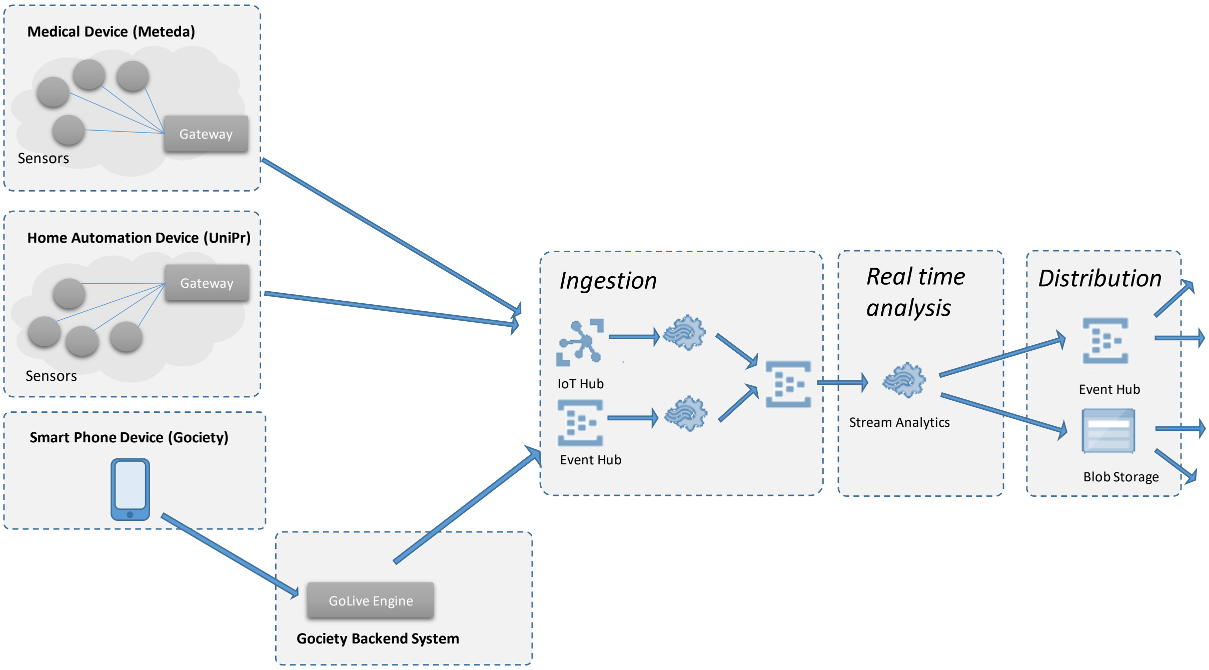

Figure 3 Platform is responsible for ingestion, real time analysis and distribution

2.4.1.1 Real time ingestion

The real time ingestion component of the system is constituted of an Azure PaaS-service called Azure IoT Hub. Azure IoT Hub supports different protocols for D2C (device-to-cloud) communication, such as AMQP, MQTT and HTTP. The *Azure IoT SDKs* (Software Development Kit) for developing clients to the IoT Hub can be found at GitHub and it supports a number of different development languages and environments.

Each device that connects to the IoT Hub has its own identity in a device repository of the IoT Hub and the communication can be bidirectional (D2C and C2D). Authentication is done by means of signature created with a symmetric key associated with the device.

See the *Azure IoT Hub Documentation* for more information about how to use the Real time ingestion API.

2.4.1.2 Batch ingestion

The batch ingestion component of the system is constituted of an Azure PaaS-service called Azure Event Hub. Azure Event Hub supports two different protocols for submitting events, namely AMQP and HTTP. The *Azure IoT SDKs* (Software Development Kit) for developing clients to the Event Hub can be found at GitHub and it supports a number of different development languages and environments.

Authentication is done by means of a SAS (Shared Access Signature) token created with a key.

The batch ingestion API can be used for transmitting data to the platform both from a collection service like the Gociety GoLive Engine but also from analysis service like the UniPr analysis service.

See the *Azure Event Hub Documentation* for more information about how to use the Batch ingestion API.

2.4.2 Real time analysis

Real-time analysis is enabled by an Azure PaaS service called Azure Stream Analytics. The real time analysis could be used for enriching the data with for instance mean value calculations or alerts at the detection of a value exceeding a threshold.

2.4.3 Distribution of data

Data is distributed and exposed to connected systems (both partner systems/services and 3rd party systems/services) in two ways.

- Azure Event Hub
- Azure BLOB Storage

2.4.3.1 Distribution via Azure Event Hub

Connected systems can retrieve data from the ENSAFE platform via an Azure Event Hub API. Consuming data from the Azure Event Hub can only be done via AMQP. The *Azure Event Hubs SDKs* found at GitHub contains source code for developing a data consuming clients to the Azure Event Hub from different programming languages and environments.

See the *Azure Event Hub Documentation* for more information about how to use the Event Hub based Distribution API for consuming data from the ENSAFE platform.

2.4.3.2 Distribution via Azure BLOB Storage

Connected systems can retrieve data from the ENSAFE platform via an Azure BLOB Storage. Consuming data from the Azure BLOB Storage can be done via a REST API or natively from a number of languages and environments by using one of the *Azure Storage SDKs* found at GitHub. contains source code for developing a data consuming clients to the Azure Event Hub from different programming languages and environments.

See the *Azure Storage Documentation* for more information about how to use the BLOB Storage based Distribution API for consuming data from the ENSAFE platform.

2.4.4 Individualized storage

In order to respect the individual’s wish and integrity the ENSAFE platform will support an individualized storage. This means that care recipient (or a trustee) can select where and how to store the data associated with the care recipient. The ENSAFE platform will have a small number of default storage options that can be selected from. Examples of such storage targets could be

- DropBox
  The individual’s private storage account with DropBox. See the *DropBox website* for more information about DropBox.

- Püffell
  The individual’s account with Püffell could be used as a target for the data associated with the individual. See the *Püffell website* for more information about Püffell.

- Health for Me (Hälsa för mig)
  Health for Me is a free personal health account available to everyone in Sweden aged 18 or over, sponsored by the Swedish eHealth Agency. See the *Swedish eHealth Agency website* for more information about Health for Me.

Future development and addition of more storage options is supported by the platform.

2.4.4.1 Storing of data in individualized storage

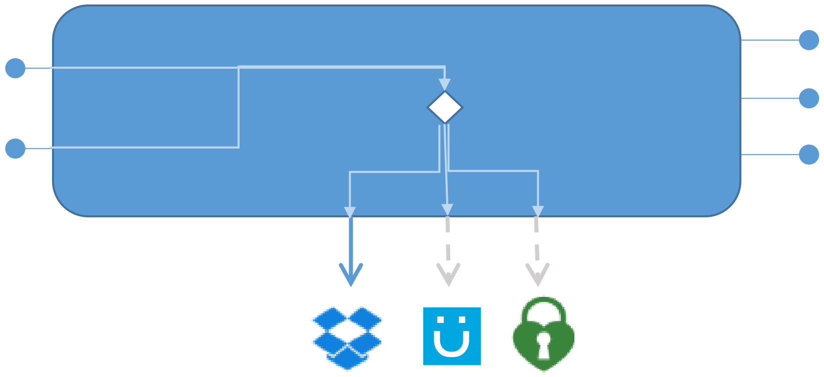

Figure 4 The individual's selected storage provider is picked when storing data

When the data associated with an individual care recipient has been identified, the choice of storage provider is looked up in a subscription database and the data is stored with that provider. Subscription and credential details for the relevant storage provider is held in a private encrypted database.

2.4.4.2 Retrieving data from the individualized storage

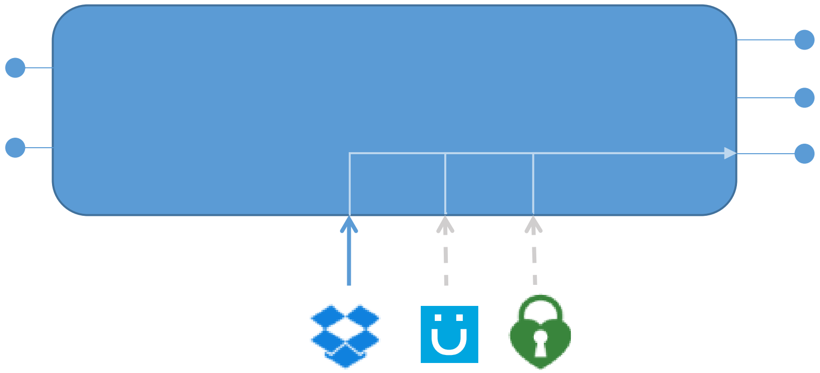

Figure 5 The individual's selected storage provider is picked when retrieving data

The client retrieving data for an individual user must authenticate with a token (SAS token), provided by the care recipient or created from the care recipient's secret key. When the individual in this way is identified, the choice of storage provider is looked up in the subscription database. The platform will retrieve the data from underlying storage provider and return it to the client via a common API. For more information about SAS tokens, see *Shared Accessing Signatures*.