BONVOYAGE

From Bilbao to Oslo, intermodal mobility solutions, interfaces and applications for people and goods, supported by an innovative communication network

Research and Innovation Action GA 635867

Deliverable D6.3:

Modelling and performance analysis in realistic scenarios

Deliverable Type:	Report
Deliverable Number:	6.3
Contractual Date of Delivery to the EU:	31/12/2017
Actual Date of Delivery to the EU:	27/12/2017
Title of Deliverable:	Modelling and performance analysis in realistic scenarios
Work package contributing to the Deliverable:	WP6
Dissemination Level:	Public
Editor:	CEA
Author(s):	CEA (Andréa Vassilev, Christelle Godin, Oumayma Sakri, Etienne Labyt), FLU (Stephan

	Strodl), CNIT (Andrea Detti, Giulio Rossi, Riccardo Paolillo, Giuseppe Tropea, Mauro Losciale), TRIT (Giampiero De Angelis), CRAT (Silvia Canale, Francesco Delli Priscoli, Raffaele Gambuti, Federico Lisi, Alessio Martino, Antonio Pietrabissa)
Internal Reviewer(s):	Trenitalia (Giampiero De Angelis)
Abstract:	This deliverable is about results of the tests and analyses of the BONVOYAGE components we have developed, in realistic scenarios. Tailored realistic scenarios (number of users, representative test area, etc.) have been defined for each component under test. Possible bottlenecks have been identified and tested. Usability tests of the BONVOYAGE mobile App, as well as validation and performances tests of user sensing functionalities, are described. Performance of the OGB-based discovery service, of the "Green Score Policy" (considered as part of the Membership Management Module) and of the "Personalization" modules has been qualitatively and quantitatively tested, based on the realistic scenarios and modelled loads, as well as on information coming from a dedicated focus group.
Keyword List:	Android Application, Application Programming Interface, performance test, usability test, user sensing, load model, realistic scenario, membership management, Green Score Policy, multimodal trip.

TABLE OF CONTENTS

List of Figures

List of Tables

Abbreviations

ABBREVIATION	DEFINITION
ACC	Accelerometer
API	Application Programming Interface
APP	Android Application
BLE	Bluetooth Low Energy
CPU	Central Processing Unit
DB	Database
EC	European Commission
EU	European Union
Gb	Gigabytes
GPS	Global Position System
GSR	Galvanic Skin Response
GTFS	General Transit Feed Specification
HMI	Human Machine Interface
IBI	InterBeat Interval
MAG	Magnetometer
OS	Operating System
PPG	Photoplethysmogram
RAM	Random Access Memory
SLA	Service-level agreement
ST	Skin Temperature

VM	Virtual Machine

Table 1: Abbreviations

BONVOYAGE Glossary

1.1.1.1 BONVOYAGE GLOSSARY	
TERM	DEFINITION
Membership Management	This functional module is in charge of monitoring and recording the user's usage of the BONVOYAGE platform in order to collect and update user scores according to the current score assignment policy. It must also return the ranked list of the scores as well as the list of awards for a given user.
User	End user of the BONVOYAGE platform who looks for a single travel or for freight transportation service (for present moment or a future one) for him/herself or another user.
Training	Step when the model is trained to recognize transport mode or stress level, based on an annotated database.
Validation	Step when the model is run on new sub parts of the annotated database, which were not used during the training step. This validation step is used to assess model's performance.
Test	Step when the previously validated model is run on new data.
Multi-modal trip	Passenger [freight] involving multiple modes of transportation [rail, ship, plane, car, local public transport, bicycle, walk, [truck]], where each mode is offered by a different transport service provider through a specific vehicle [container].

Table 2: BONVOYAGE Dictionary

1 Introduction

1.1 Deliverable Rationale

Work-Package 6 (WP6) is the important technological WP that, in BONVOYAGE, is dedicated to the development of the Multimodal Integrated Interfaces and Apps. This WP will issue the final BONVOYAGE Android application, which is going to be a ready-to-download product by interested public users. WP6 was divided into three tasks. This deliverable is the document that reports work carried out by Task 6.3, which is dedicated to "Modelling and Performances analysis in realistic scenarios". The work carried out here is of outmost importance for assessing impact of the system at the users' level, and it is going to be used by the integration work-package (WP7) for further improvements of the developed solutions.

1.2 Quality review

The internal Reviewer responsible of this deliverable is Giampiero De Angelis from Trenitalia.

VERSION CONTROL TABLE			
VERSION N.	PURPOSE/CHANGES	AUTHOR	DATE
0.1	Front page and tentative ToC	E. Labyt - CEA	17/09/29
0.2	Final Table of Contents	E. Labyt - CEA	17/11/06
1.0	First partial draft	E. Labyt - CEA	17/10/26
2.0	Second partial draft	E. Labyt - CEA	17/11/08
3.0	Final complete draft and contributions	E. Labyt - CEA	17/11/15
3.1	Comments to all contributions	ALL authors	17/11/24
3.2	Assembly of reviewed contributions	E. Labyt - CEA	17/12/05
3.3	Quality Review of the document	TRIT	17/12/12
3.4	Final version	CNIT - CEA	17/12/19

1.3 Executive summary

Deliverable D6.3 describes the results of the activities of Work Package 6 (WP6) "Multimodal integrated Interfaces and Apps" and particularly of task 6.3 "Modelling and Performances analysis in realistic scenarios", that spans the period November 2016 – November 2017 (M19 – M31). The aim of deliverable D6.3 is to show results of the tests and analyses of the BONVOYAGE components in realistic scenarios. We take into consideration realistic scenarios (number of users, representative test area, etc.) as defined in Chapter 4. Possible bottlenecks (e.g. bandwidth, data plans, CPU load, data storage etc.) have been identified and tested. Usability tests of the BONVOYAGE App (still undergoing development for what concerns the key integration aspects), as well as validation and performances tests of user sensing functionalities are also described. Additionally, performance of the "Green Score Policy" (considered as part of the Membership Management Module) has been qualitatively tested based on insights gained from a dedicated a focus group.

1.4 Deliverable description

WP6 aims at designing and developing all the mechanisms that are necessary to seamlessly interact with the heterogeneous external actors of the BONVOYAGE platform and particularly the sensing and actuation functionalities of the platform. Both functionalities are tightly related to different components of the BONVOYAGE architecture that have been developed inside other work packages.

As far as the sensing functionalities are concerned, they do not only include vertical and operator-technology-dependent adapters to access transport operators' data sources and systems, but also mobile applications and standardized interfaces to gather end-user data and feedbacks (participatory sensing): such an amount of information is forwarded (through the Internames Communication System developed in WP3) to the Metadata Handling Tool developed in WP5. Data gathering and forwarding takes place, either periodically and/or following specific events and/or following specific Metadata Handling Tool on-demand requests.

Actuation functionalities must operate within an ecosystem of possibly heterogeneous journey planning platforms, taking into consideration the BONVOYAGE service goals (e.g. those of the Intelligent Transport Functionalities developed in WP4) and tailoring to the involved transport operators' peculiarities by Service Adaptation (developed in WP5).

Overall, the whole WP6 comprises the following objectives:

1. Design and development of the technology dependent interfaces towards the external actors (transport operators' and related systems / data sources, end user applications).

2. Define Design Elements, Interactions and Guidelines as part of a User Centric Design process.
3. Design and development of an application for the Mobile Participatory Sensing Services.
4. Design and development of an application for Mobile Information Services
5. Perform preliminary functional unit and component tests of the implemented interfaces and applications; fine tune of the developed software components.

Out of the above, this document focuses on illustrating tests and analyses of the developed components in their relevant scenarios. Hence it will focus on the porting of user sensing algorithms into the real-world Android BONVOYAGE App, on usability tests of the BONVOYAGE App, on the definition of a realistic load model and user behaviour derived from use cases, which we use to evaluate performance of single components and test their outputs.

2 From algorithm to Android App: User sensing

2.1 Transport mode recognition

2.1.1 Introduction

Transport mode recognition aims at automatically classifying the transportation modes from the user smartphone sensor data recorded during his/her journeys.

The data flow, as detailed in Chapter 3 of deliverable D4.2, is briefly summarized below:

- From the Smartphone sensors data, namely
  - the accelerometer (acc),
  - magnetometer (mag)
  - and GPS data,
  - sampled at different frequencies (from 1 Hz for GPS to 200 Hz for acc),
  - compute different features over fixed time windows (typically 5 sec.)
- Depending on the availability of GPS data, the appropriate classifier (the one with GPS, or the one without GPS) is used. These classifiers take as input the previous features and produce a predicted class (transportation mode)

Features computation has been developed in Matlab [1].

To implement the features into our Android application (App), we first translate the Matlab code into C code, using Matlab coder [2]. Then, we use the Android NDK [3] that allows to integrate directly some functionalities using native-code languages such as C.

Classifiers have been developed using WEKA software [4]. WEKA produces Java objects so there is no specific effort to implement the classifiers into our App.

In this section, we address 2 problems:

- Validation of features computation and classification results,
- Assessment of classification performance.

2.1.2 Validation of features computation and classification results

The goals are:

- Validate the feature computation: as there are 2 important steps (translation into C with Matlab Coder and integration into our App with Android NDK), it is important to check that the computation is correct;

- Validate the switch to the proper classifier, depending on the GPS data availability;
- Validate the call to classifier.

Android Application Version is 01.05c

We define a simple test trip that combines indoor and outdoor location in order to check the call to the proper model. The test trip is composed of the following 5 steps:

1. Starting indoor, Smartphone 'still' on a desk,
2. Walk outside, smartphone in the left front trouser pocket,
3. Smartphone in a 'still' position, outside,
4. Come back indoor walking, smartphone in the left front trouser pocket,
5. Indoor, Smartphone 'still' on a desk.

We made this test trip with 2 different phones, a NEXUS 6 and a NEXUS 5X.

The first test has been done with a Samsung NEXUS 6. In Figure 1 the user original annotation (bottom), along with some metadata (top), is displayed.

```
 *** h5 info ***
There are 22 attributes
file name           : D6B7253989E67B0E_00002_20170531143828
creator_version     : BETA_9
format_version      : 01.00
precise start(dday): 7.368466e+05
precise start (UTC): 2017/05/31 14:38:28:180
start string (UTC)  : 2017/05/31 14:38:28
PhoneID             : D6B7253989E67B0E (NEXUS 6)
user                : av-user-003   (Andrea)
acqui_number        : 2
nbFeatures          : 13
nbClasses           : 8
collect_version     : 01.05
model_version       : V0401
user_comment        : ***
Buffers             : acc, mag, gps,
```

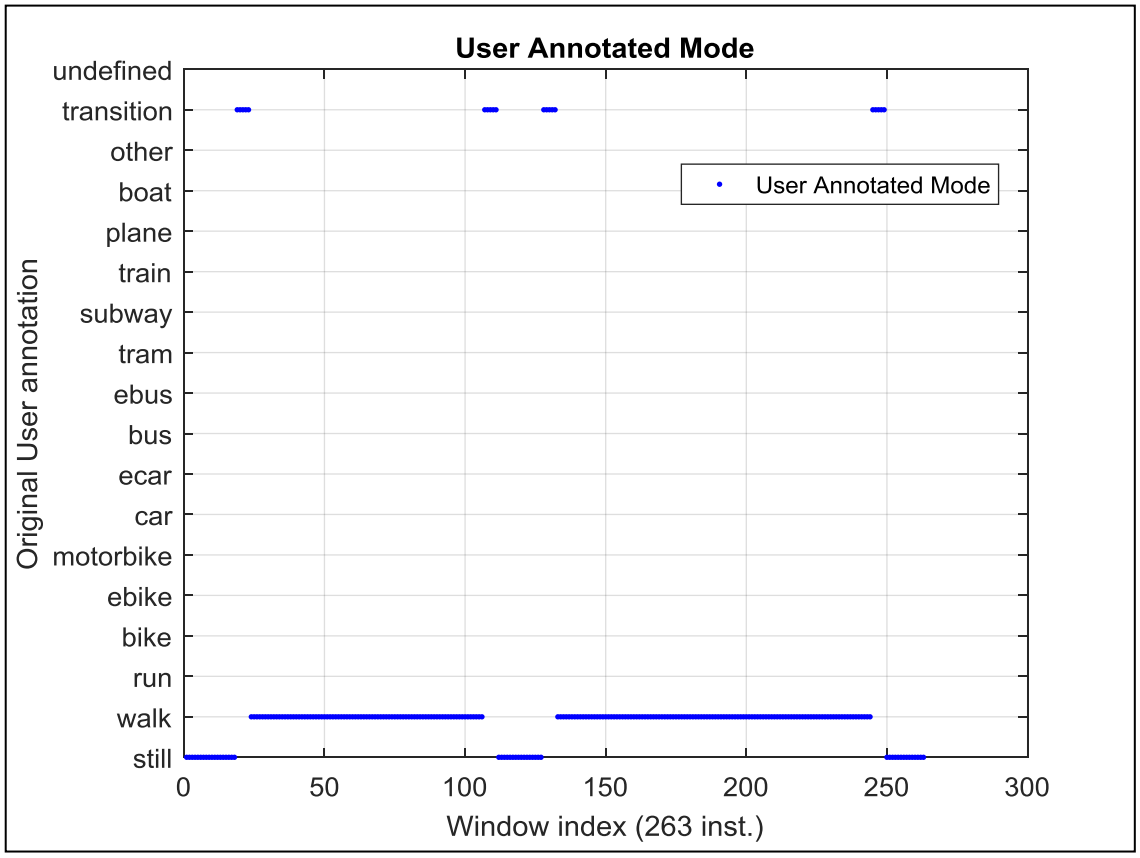


Figure 1. First Test with NEXUS 6. Reference user-annotated data.

In Figure 2 results of the test are displayed. We checked that GPS availability is correct (top) and compared it with the called model (bottom).

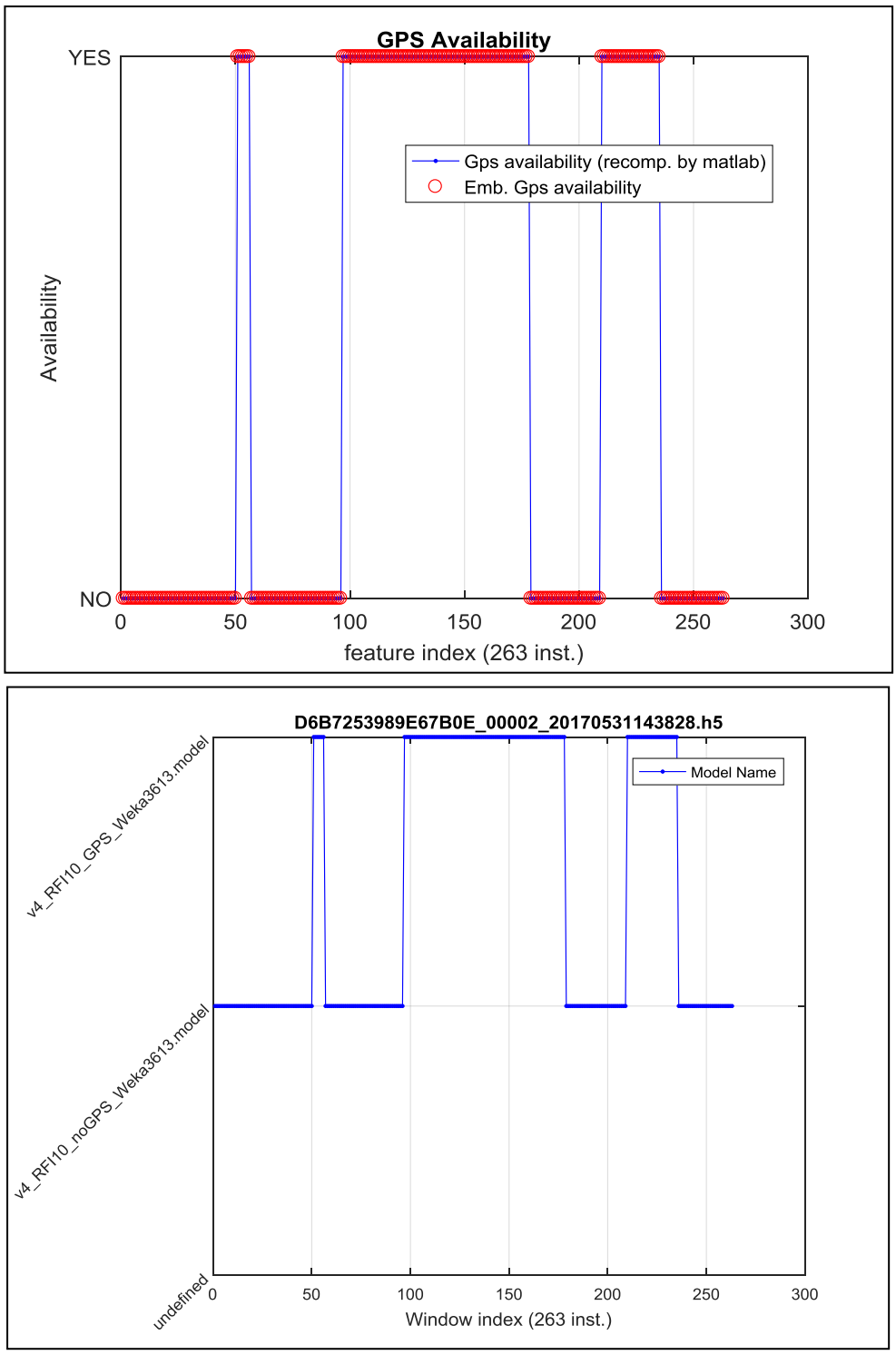


Figure 2. GPS availability is checked (top) and model is called (bottom)

This validates the switch to the proper classifier, depending on the GPS data availability.

The next step was to follow a step-by-step procedure to validate the feature computation:

- Get embedded segmentation, i.e. for each 5 sec long window, which samples for acc (resp. mag) have been used to compute the embedded features.
- Using the raw sensors data and the embedded segmentation, recompute in Matlab the features: this is our reference.
- For each feature X (computed by Android), there is a reference feature (Xref computed by Matlab), and we compute a relative difference: ef = norm(X - X_REF) / norm(X_REF)[1].
- Take the max over only the features available everywhere[2].

The results are very good with a maximum relative difference of 4.76e-07, as summarized in Figure 3.

```
      /// Compare Emb vs Matlab (thresh = 3.00e-03) ///
featuresBoolMatrix agrees
Rel. diff by feature (All are below 3.00e-03 ==> no further PLOT)
    MAG_NORM_STD        ACC_STD_V       ACC_STD_H1       ACC_STD_H2  ACC_V_BAND_EN_1  ACC_V_BAND_EN_2  ACC_V_BAND_EN_3  ACC_V_BAND_EN_4
        0.00e+00         8.72e-08         1.11e-07         1.19e-07         5.57e-08         5.96e-08         5.78e-08         3.56e-08
   Some features (because not always available) have not been taken into account into the global Emb vs Matlab error : GPS_SPD_MED
```

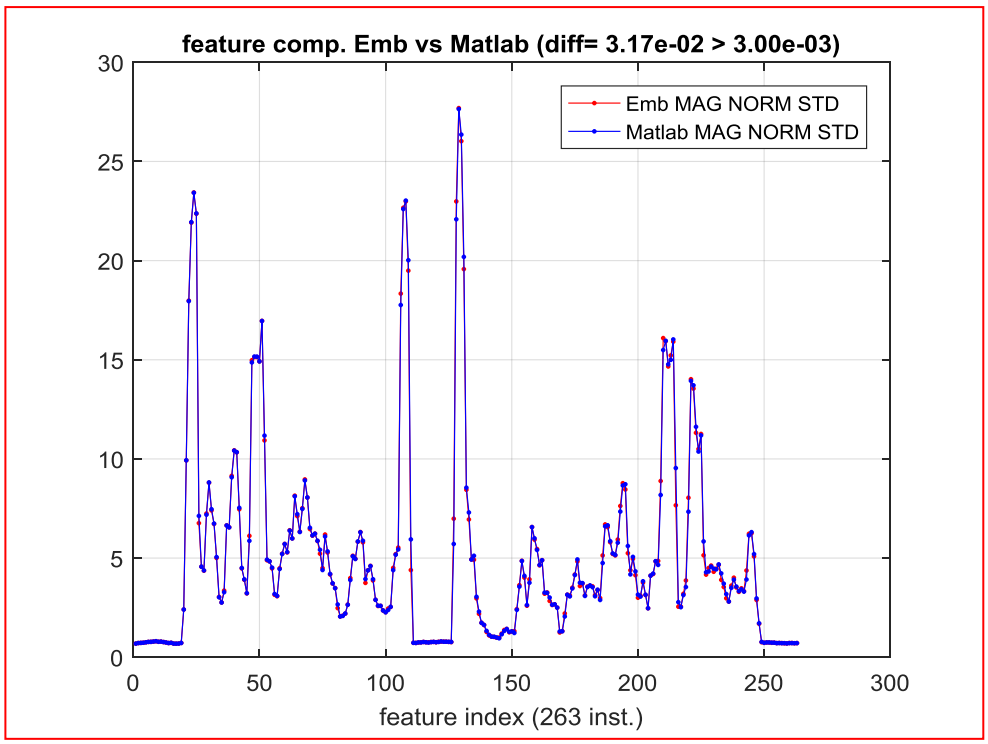
```
ACC_SPEC_CENTRO  MAG_SPEC_CENTRO  ACC_SPEC_SPREAD  MAG_SPEC_SPREAD
        3.03e-08         7.73e-08         3.79e-08         4.76e-07
```

Figure 3. Results of the feature computation validation

This validates the features computation.

It is relevant to highlight that using the embedded segmentation to recompute the features is important: if it is not available or if we recompute segmentation from the sensor timestamp, we obtain differences of 1-3%, as shown in the graph and summary results of Figure 4.

[1] non symmetric result
[2] In our case, the feature GPS_SPD_MED is not taken into account



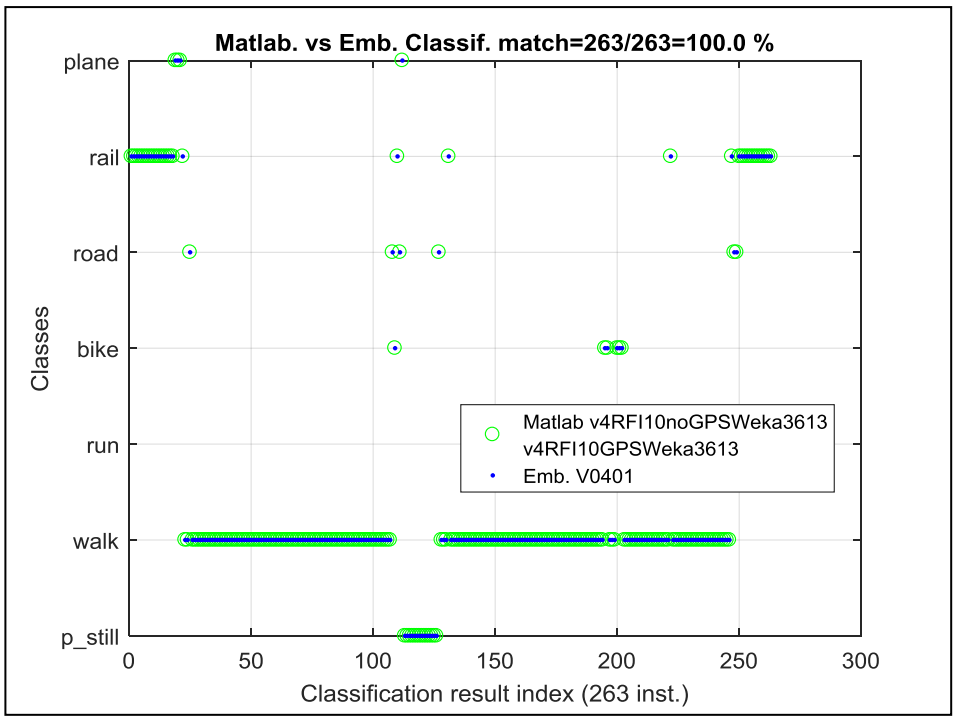

```
Rel. diff by feature (13 differ more than 3.00e-03)
    MAG_NORM_STD        ACC_STD_V       ACC_STD_H1       ACC_STD_H2  ACC_V_BAND_EN_1  ACC_V_BAND_EN_2  ACC_V_BAND_EN_3  ACC_V_BAND_EN_4
        3.17e-02         1.32e-02         1.95e-02         2.12e-02         1.61e-02         1.54e-02         2.19e-02         1.64e-02
  Some features (because not always available) have not been taken into account into the global Emb vs Matlab error : GPS_SPD_MED
```

Figure 4. Comparison between an embedded feature and a Matlab feature, if embedded segmentation is not available

As a next step, we compare the embedded classification with the result of the call to the classifier[3] with features computed (and validated) in Matlab. Figure 5 (top) shows that the match is perfect. This result validates the implementation of the classifier.

Finally, it is interesting to look at the performance of the classifier for this trip. First, the original user annotations (263 instances) are filtered in order to remove 'transition' and 'other' instances, leading to 241 instances (see details in Chapter 3, part 3.1.3.2 of deliverable D4.2). Then accuracy is computed, 84% (Figure 5, bottom). We can notice that 'still' in indoor is not well-classified and confused with 'rail'.

[3] in our case, the classifier is a WEKA object that is called from Matlab



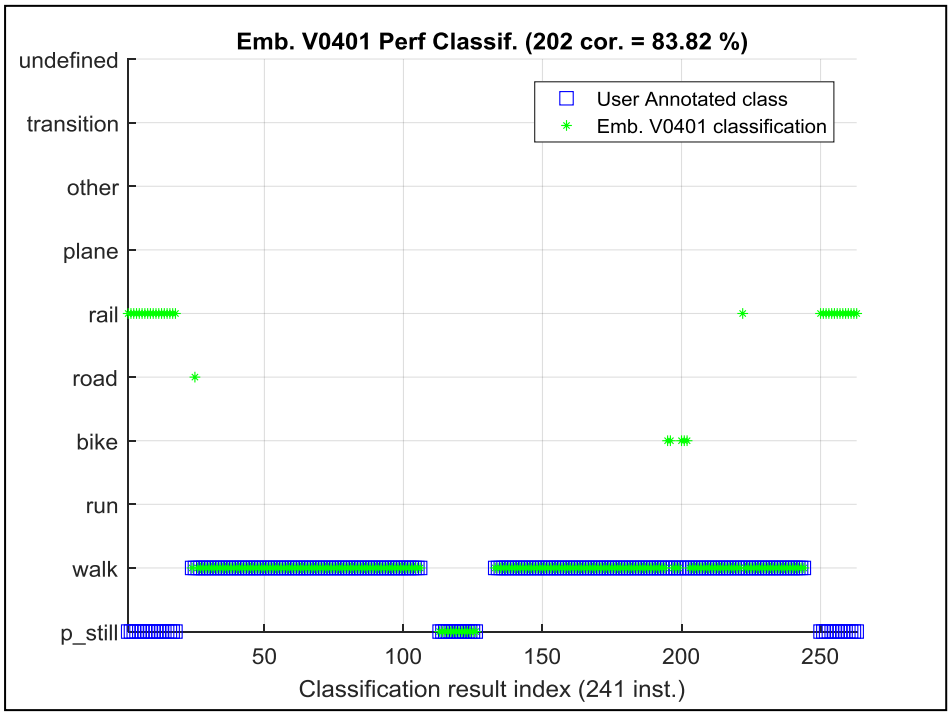


Figure 5. Embedded classification result vs Matlab classification result (top) and vs User Annotation (bottom)

A second test was conducted with a NEXUS 5X, in order to assess if device-dependent problems arise in the implementation. In Figure 6 the user-provided original annotation is shown, which we use as a reference.

```
 *** h5 info ***
There are 22 attributes
file name          : C21EFA0AA4DCD6A5_00002_20170601081110
creator_version    : BETA_9
format_version     : 01.00
precise start(dday): 7.368473e+05
precise start (UTC): 2017/06/01 08:11:10:697
start string (UTC) : 2017/06/01 08:11:10
PhoneID            : C21EFA0AA4DCD6A5 (NEX5.1 Andrea)
user               : av-user-003  (Andrea)
acqui_number       : 2
nbFeatures         : 13
nbClasses          : 8
collect_version    : 01.05
model_version      : V0401
user_comment       : ***
Buffers            : acc, mag, gps,
```

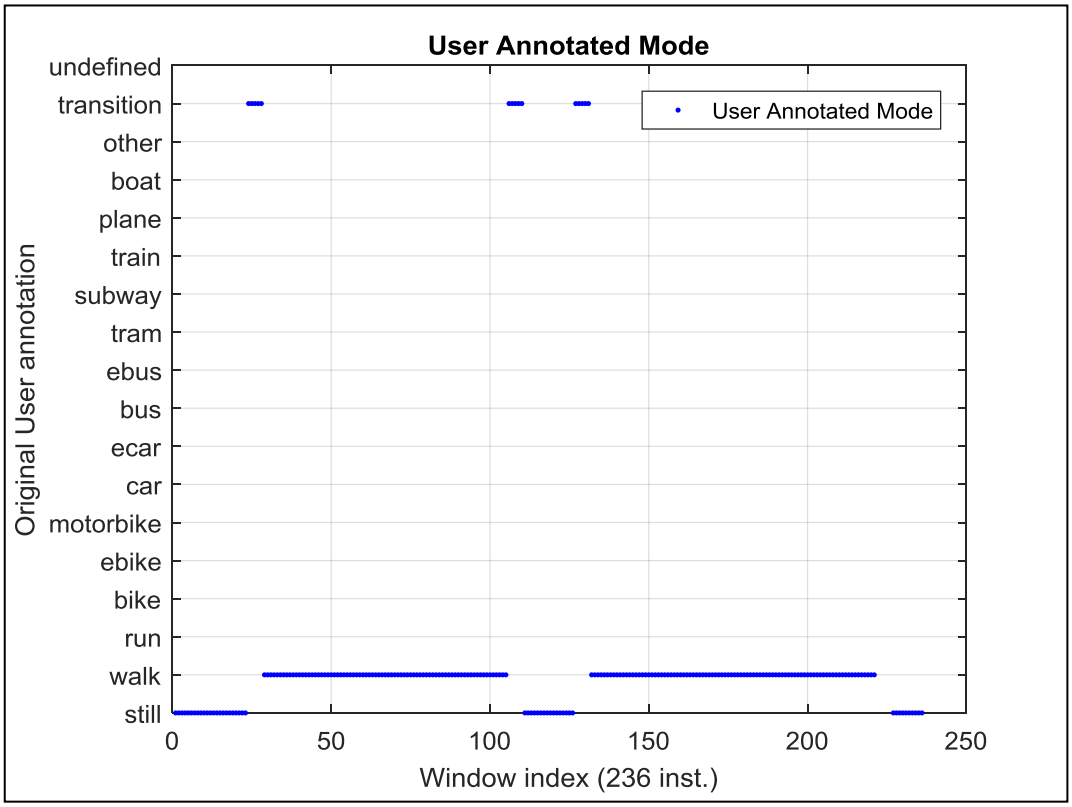


Figure 6. Second Test with NEXUX 5X. Reference user-annotated data

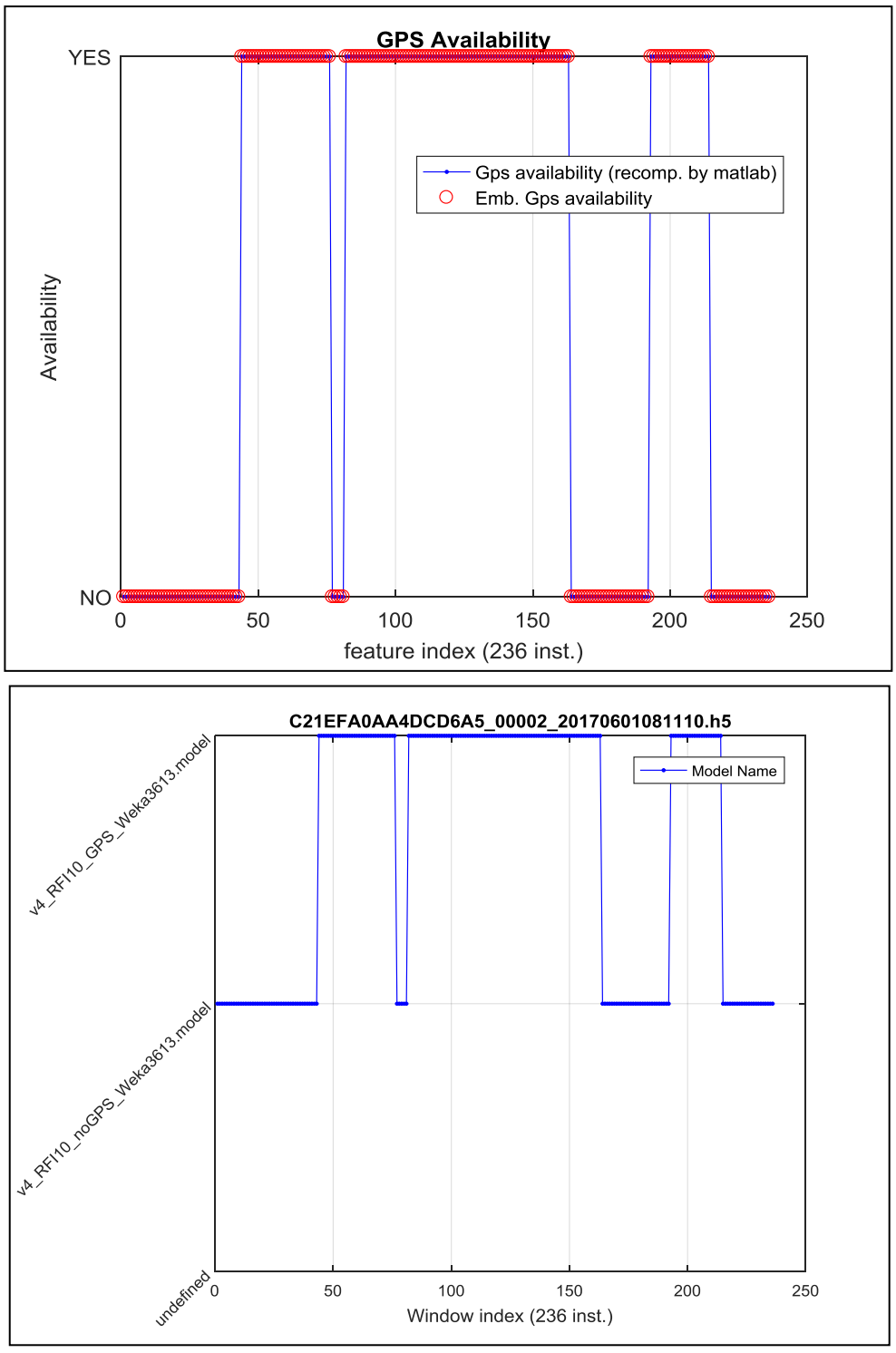


Figure 7. NEXUS 5X GPS availability (top) and model called (bottom)

In Figure 7 we can observe the results of the GPS availability test and the switch to the correct classifier.

Feature computation has been also validated with a maximum relative difference of 4.41e-07, as shown in Figure 8.

```
      /// Compare Emb vs Matlab (thresh = 3.00e-03) ///
featuresBoolMatrix agrees
Rel. diff by feature (All are below 3.00e-03 ==> no further PLOT)
    MAG_NORM_STD        ACC_STD_V       ACC_STD_H1       ACC_STD_H2  ACC_V_BAND_EN_1  ACC_V_BAND_EN_2  ACC_V_BAND_EN_3  ACC_V_BAND_EN_4
        0.00e+00         7.12e-08         1.21e-07         1.11e-07         5.87e-08         6.67e-08         6.15e-08         4.62e-08
   Some features (because not always available) have not been taken into account into the global Emb vs Matlab error : GPS_SPD_MED
===> MEX Features validated 4.30e-07 < 1.00e-06
===> ANDROID Features validated 4.41e-07 < 3.00e-03
```

Figure 8. NEXUS 5X feature computation summary results

Finally, we have validated the embedded classification with the result of the call to the classifier (Figure 9), and noticed that the performance of the classifier for this trip was very good (96%) because the 'still' indoor step is correctly detected in this case.

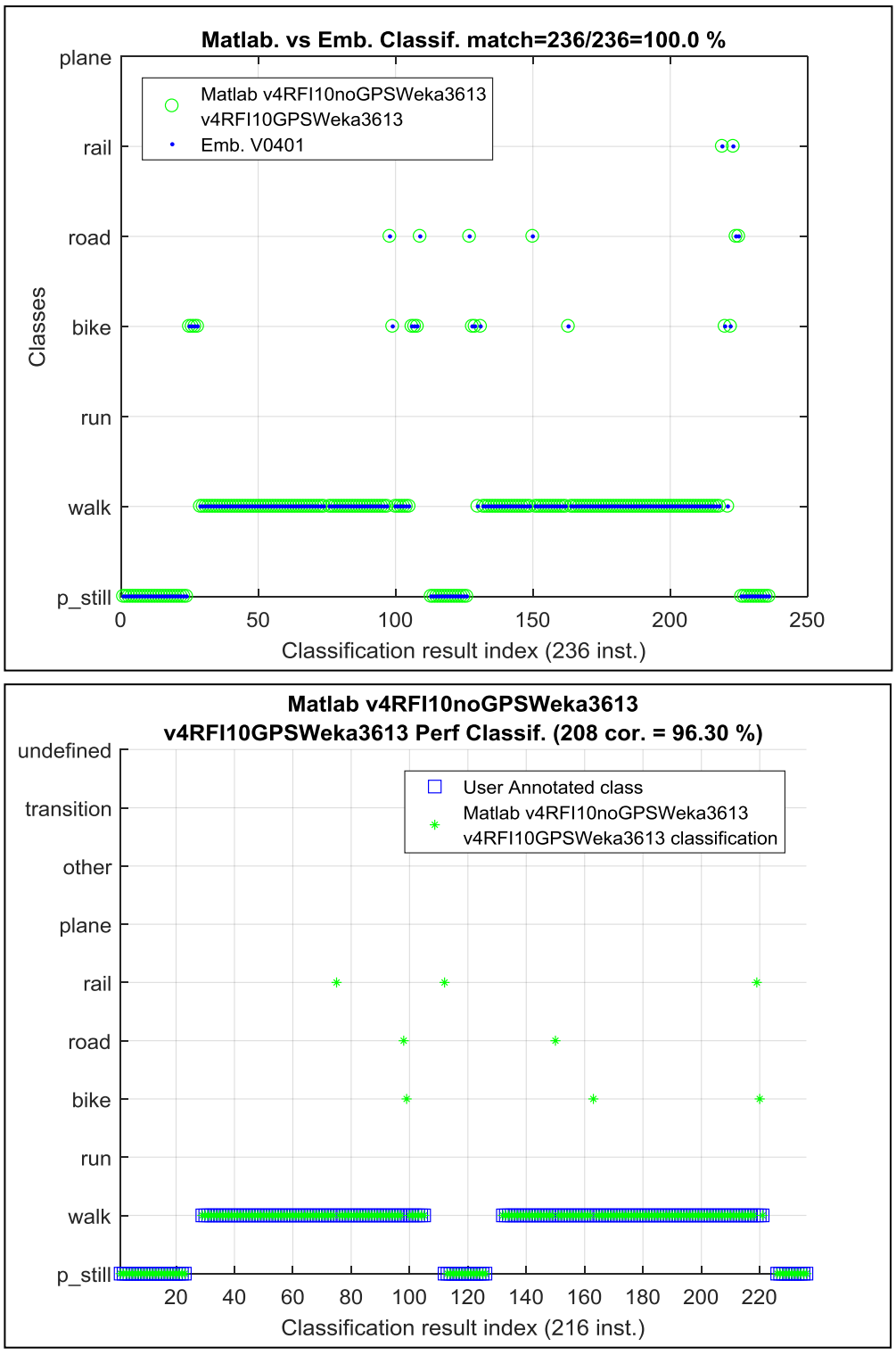


Figure 9. Embedded classification result vs Matlab classification result (top) and User Annotation (bottom)

2.1.3 Classification Performance assessment

In section 3.1 of deliverable D4.2, we have explained how our "Transport Mode Recognition" functionality has been developed using an annotated database, acquired with 22 different subjects. Classification performance has been assessed using a leave-one-subject-out cross validation procedure [5].

However, it is important to do also another test with a completely new subject, i.e. which did not participate to the database creation. The test consists in performing a trip, taking annotations of the real transportation mode and then in post process compare the mode prediction with the real evidence.

This subject will be named 'E'. 'E' did a trip from Grenoble to Brussels, and then from Brussels to Lille and Dunkerque. This trip is composed of 8 uni-modal trip legs (see Table 3). 'E' was holding two different smartphones (a Samsung S5 and a NEXUS 5). For each trip leg, 'E' tried to record the trip with both smartphones, but he did not always succeed (e.g. Trip leg #1 was only acquired on NEXUS 5). Therefore, we had 15 (and not 16) different tests, considering a test as a uni-modal trip leg for a given phone.

Trip Leg Numer	Test Number	Main Mode	Day	Device	Trip	Classification Performance
2	1	walk	20161121	Samsung S5	Walk CEA to Railway station	91.30%
3	2	bus	20161121		Train Grenoble Lyon	80.15%
4	3	train	20161121		TGV Lyon-Brussels	99.46%
5	4	walk	20161121		Walk in Brussels	90.90%
1	5	tram	20161121	NEXUS 5	Tramway	87.06%
2	6	walk	20161121		Walk CEA to Railway station	69.90%
3	7	bus	20161121		Train Grenoble Lyon	96.05%
4	8	train	20161121		TGV Lyon-Brussels	99.75%
5	9	walk	20161121		Walk in Brussels	70.97%
6	10	train	20161124	Samsung S5	TGV Brusses Lille	99.67%
7	11	train	20161124		Train Lille Dunkerque 1st part	76.15%
8	12	train	20161124		Train Lille Dunkerque 2nd part	87.71%
6	13	train	20161124	NEXUS 5	TGV Brusses Lille	99.04%
7	14	train	20161124		Train Lille Dunkerque 1st part	89.22%
8	15	train	20161124		Train Lille Dunkerque 2nd part	91.26%

Table 3: Classification performance for the 15 trips

On each test, we first performed the classification, second we have produced a mode prediction, comparing it with the truth coming from the annotation, and finally we computed a performance. The performance is simply the percentage of correct math (also known as accuracy).

Note that, at the time when the test was done (Nov 2016) even if the "Transport Mode Recognition" functionality was embedded into our App, it was not validated (see 2.1.3). Therefore, the whole classification process (features computation and call to classifier) was done in Matlab.

From the 15 results, we can compute basic statistics (see Table 4).

Mean	88.57%
Median	90.90%
Standard Deviation	9.80%
Min	69.90%
Max	99.75%

Table 4: Statistics for Classification Performance

As it can be seen, performances are quite good, with a median and a mean around 90%, and a minimum of 70%.

2.2 User's stress level monitoring

2.2.1 Introduction

User stress level monitoring makes the platform able to evaluate the real-time stress level of the traveller that is using BONVOYAGE during his/her trip. This tool is useful in order to adapt the trip by considering the traveller history of stress with respect to each transport mode or specific trip events (e.g. short or long commutation time). It can also be used in order to adapt the trip in response to an unexpected event. To use this tool, the traveller must wear sensors that are able to monitor his/her activity as well as his physiological parameters. The chosen wearable system is the wristband Empatica E4. It monitors PPG (photoplethysmogram), IBI (interbeat interval), GSR (galvanic skin response), skin temperature (ST) and ACC (accelerometer). An android software library, plus a companion application, is being developed in WP6 by integrating algorithms developed in WP4 (see Chapter 3 of D 4.2). The library offers a function to collect data such as measurements from E4, to calculate features from raw data and estimate a level of stress by combining those features.

In this deliverable we discuss the evaluation of the android application, which is built upon the software library, during experiments in real-life scenarios.

2.2.2 First Android code validation

In WP4, Matlab code has been developed in order to process offline raw data collected from E4 wirstbands and to compute features. This code has been translated into android by using Matlab realtime coder® tool. Even if this tool should be an automatic translation, several functions are not supported and need to be coded again in Java.

After implementing all these corrections, a comparison is done between the features computed by Matlab code and by Java code. The root mean square error between them is about 5.15e-14, which validates the translation. An example of comparison is given in Figure 10. Each one of the 29 features is correctly computed.

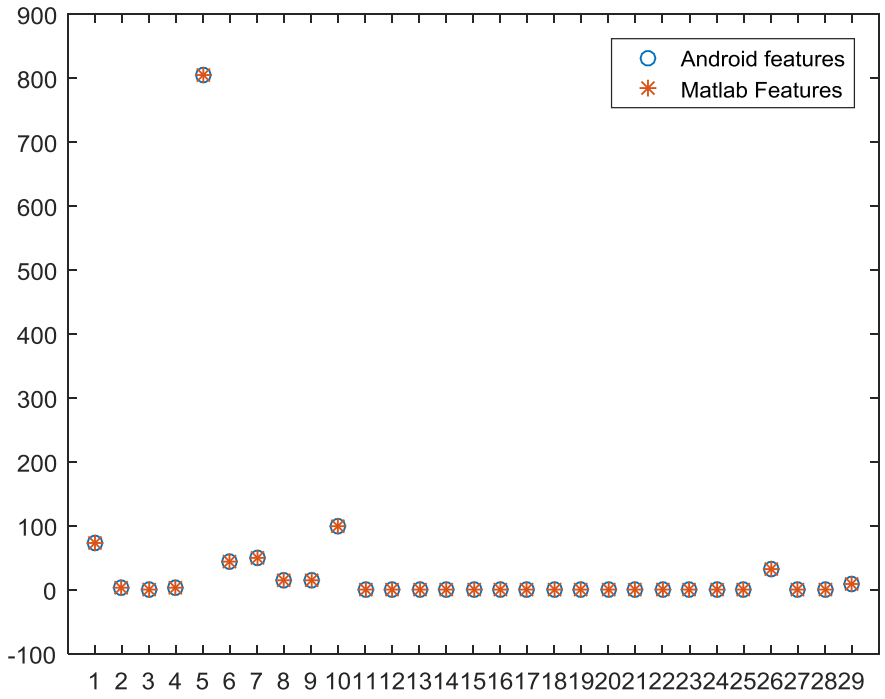


Figure 10. Perfect match of features computed by Matlab code vs. Java code

After features' evaluation, data matching has being done using a machine learning tool named Weka. The model learned by Weka is translated into Java code by the tool itself and integrated in the test application.

2.2.3 HMI to visualize results

An HMI has been developed in order to visualize the results of the tool.

This HMI has 2 modes:

- In real-time mode we can see that the results are described through a sliding curve. Each new stress evaluation creates a new point on the curve. Measurements, features and stress levels are also recorded in a file.
- In replay mode: we can choose a file and display the corresponding curves.

The displayed curves are IBI (interbeats interval) and EDA (Electrodermal activity) in one graph, and estimated stress level in another graph.

Using this HMI we can see that IBI is very often missing. It is very sensitive in motion artefacts. This result had also been appreciated in laboratory experiments by comparison between wearable and laboratory sensors. We can also follow in real-time the evolution of stress level estimation, which needs to be further enhanced.

2.2.4 Results using the application during testing phases

A preliminary data fusion algorithm has been deployed, which doesn't take into account the inter-individual variability. This algorithm is tuned for one person and tested on the same person during a week, our reference period of time considered for testing experiments. Within the reference week that person identifies 3 particular events, 2 of which have being noted as particularly stressful by the subject, and another one particularly peaceful. The corresponding HMI outputs for high stress level are represented in Figure 12: the graphs correspond to the two stressful events.

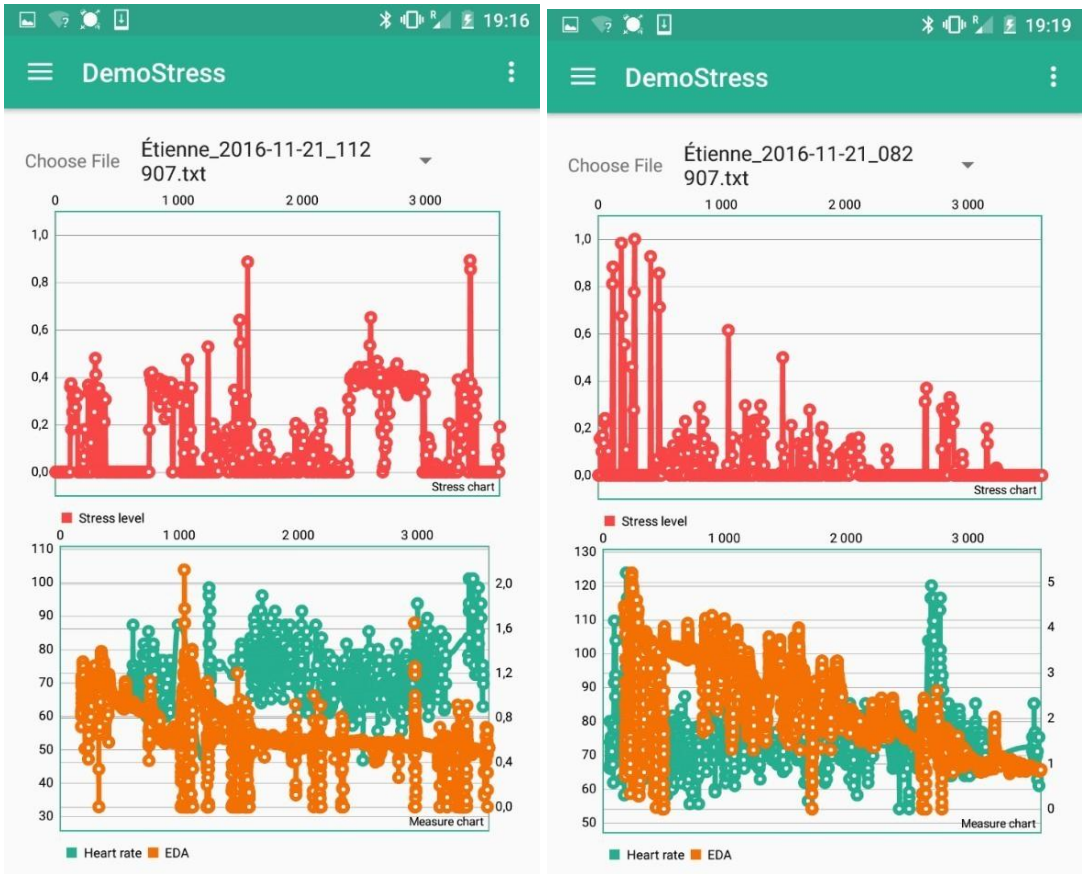


Figure 11. Examples of HMI outputs for high stress levels

In Figure 12 the graph that corresponds to the peaceful event is shown.

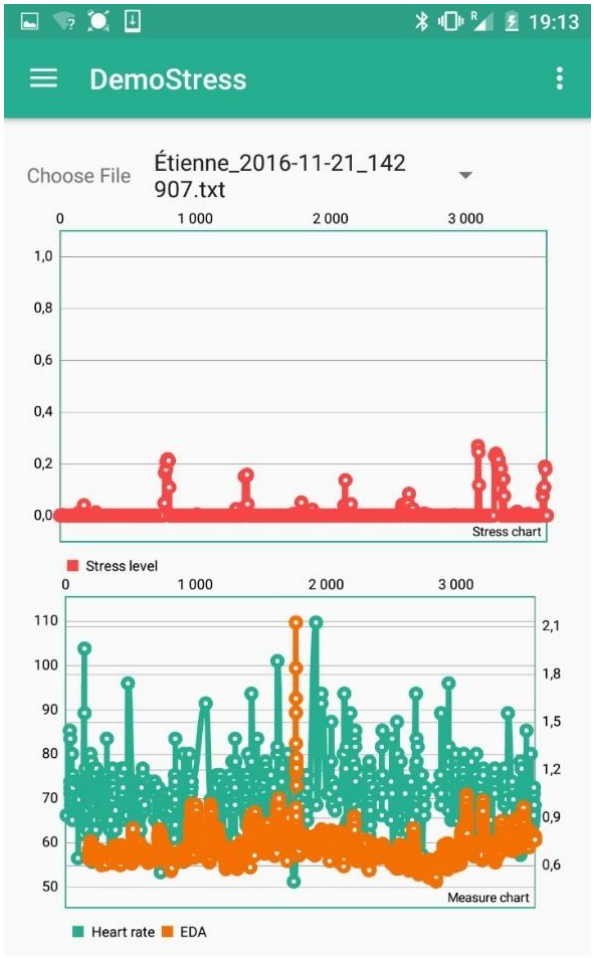


Figure 12. Example of HMI outputs for low stress levels

After that, the model has been tested on other persons. Results obtained on new subjects indicate that an individual model needs to be calibrated with respect to the specific subject.

2.2.5 Model with calibration

In order to take into account the specificities of each user, a model including a calibration step has been set up. To that end, a new model has been created, based on normalized features computed from a calibration step.

To determine the adequate manner to normalize the features, several tracks were explored such as the use of a single baseline period before all acquisitions or an average of some baseline periods acquired over initial acquisition phase.

After this analysis, and for practical reasons of use of the application in everyday life, a baseline period (rest period) was selected before the start of the acquisition. The features computed on the recorded signals over all acquisitions will be normalized against this rest period.

To illustrate improvements of model performance obtained with a calibrated model, performances of a model whose features have not been normalized (Figure 13, 1) and a model whose features have been normalized (by using an average of the features computed on signals acquired during the rest period) (Figure 13, 2) are plotted below for the training, validation and test phases of the model.

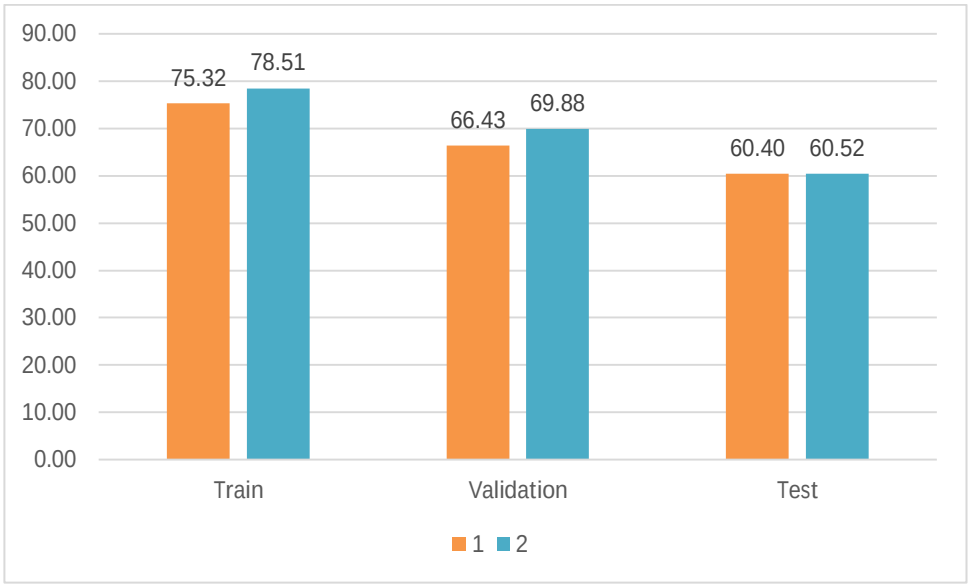


Figure 13. Comparison of classification performances between a model whose features are not normalized (1) and a model whose features have been normalized (2).

2.2.6 Code transposition and validation

Regarding the first implemented model (without calibration), the features' computation was transposed into Java with the Matlab coder tool and the classifier model was created with Weka tool and transposed to Java.

The current model, including the calibration step (calculation of normalized features) and classification, has been transposed straight from Matlab to Java for Android using Matlab coder.

Due to this process change, a new comparison between the features computed by Matlab and those computed by the Android application was done in order to validate the model implemented in Android. An example of comparison of the 29 computed features is illustrated below in Figure 14.

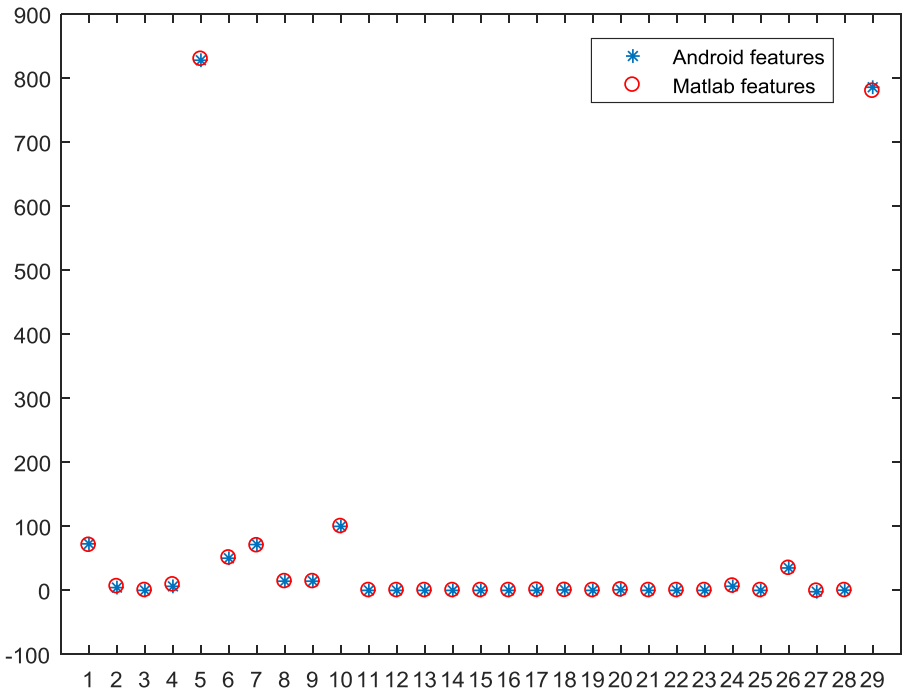


Figure 14. Perfect match of features computed by Matlab code vs. Java code (model including calibration step)

Again, the root mean square error between features computed in Matlab and Java code is very low (1.7 e^{-3}), which validates the translation.

2.2.7 Classification performance

In deliverable D4.2 we explained that we have developed the current model by using a database collected in our laboratory (DB2). The database was acquired from 20 volunteers, who were subject to four different types of stress during four different tasks, in addition to the related control task that serves to differentiate the variances that depend on the task from those due to the stress itself.

We performed the validation and the test of the classifiers by a leave-one-subject-out and a leave-one-task-out process.

To evaluate our model in ambulatory conditions, the classifier was tested on a subject of a third database (DB3) acquired in real life conditions. Figure 14 below displays the predicted stress level by our model, based on physiological data recorded in real life condition (one trip from Grenoble to Brussels) and the stress level assessed by the subject (used as ground truth). As observed on this figure, our model predictions were in accordance with subject's stress levels.

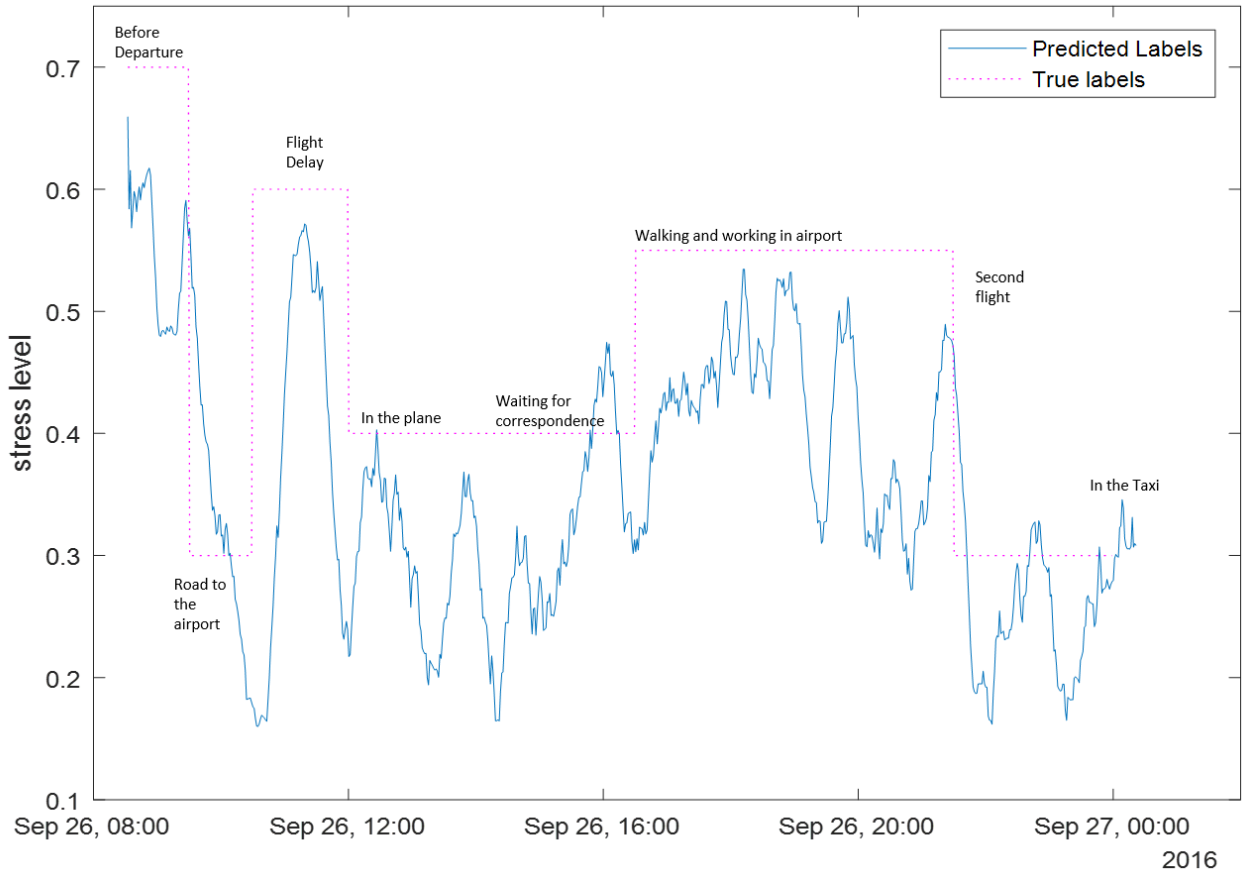


Figure 15. Comparison of the predicted stress level by our model, and stress level as assessed by the subject

3 Usability tests of BONVOYAGE App

3.1 Definition Usability

Within the ISO 9241, which is a multi-part standard from the International Organization for Standardization (ISO) covering ergonomics of human-computer interaction, the "Ergonomics of human-system interaction" are defined[4]. Particularly the ISO 9241-11 describes Usability as following: "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use."

The critical measures of usability are effectiveness, efficiency and satisfaction. Whitney Quesenbery[5] defines the following five dimensions:

- Effective (Accuracy of achieving goals)
- Efficient (Speed of completing the goals)
- Engaging (Pleasantness of use)
- Error tolerant (Preventing of errors)
- Easy to learn (Supporting of initial orientation and continued learning) (Barnum 2011, p. 11-12)

3.2 Usability Testing

Testing activities can be subdivided into two subgroups: formative testing and summative testing. Usually, while formative testing is performed in the meantime the target App is being developed, only and after finishing the product summative testing follows. The goal of formative testing is to monitor the user activity of "learning" the App and support the user in the various tasks. Differently, a summative assessment evaluates the user interaction with the App against a benchmark without interaction to the user.

"Elaborate usability tests are a waste of resources. The best results come from testing no more than 5 users and running as many small tests as you can afford."[6]

Out of these findings, just 3-5 subjects are needed to get useful results. The following elements need to be integrated: user profile, task-based scenarios, think-aloud process, after making changes test again.[7]

[4] https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-1:v1:en
[5] http://www.wqusability.com
[6] https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users

3.3 Elements of Usability Testing

Usability Tests typically consist of the following elements:

1. Define the user profile: One subgroup of the user population must be defined. Occasionally, depending on the budget, more than one subgroup can be defined.
2. Create task-based scenarios: Descriptions framed around the user's goals are defined. These tasks are the base for observing the methods for achieving a goal.
3. Use a think-aloud process: Thinking aloud offers additional benefits concerning the whole user experience (thoughts, pleasure, reactions…). You see what a user does and hear why he or she decided that way.
4. Make changes and test again: A follow-up study is applied to test the redesigned solutions.

In line with this approach, a Usability Test was conducted and is documented in the next chapter 3.4. Results are discussed in chapter 3.5.

3.4 Documentation of the Usability Tests

Below the documentation of the conducted Usability Tests is provided.

3.4.1 Define the user profile

The ecologically aware person, age 25 to 40, who travels a couple times per week for business and has a high level of mobile device experience.

3.4.2 Create task-based scenarios

The user wants to travel to a target destination. The user either wants to take the current position as start-point or wants to plan for the future and therefore defines a custom start-point. After defining the route the user wants to get a complete view on stopovers, interchange facilities, duration and eco points. From time to time the user wants to view his/her total amount of eco points.

5. Define your route from Bilbao to Oslo via Rome and save it.
6. Describe your route with stopovers, interchange facility, duration and eco points.
7. Check your total amount of eco points
8. View your saved routes
9. Give feedback

[7] Barnum, C. M. (2011). Usability Testing Essentials. Burlington: Elsevier Inc.

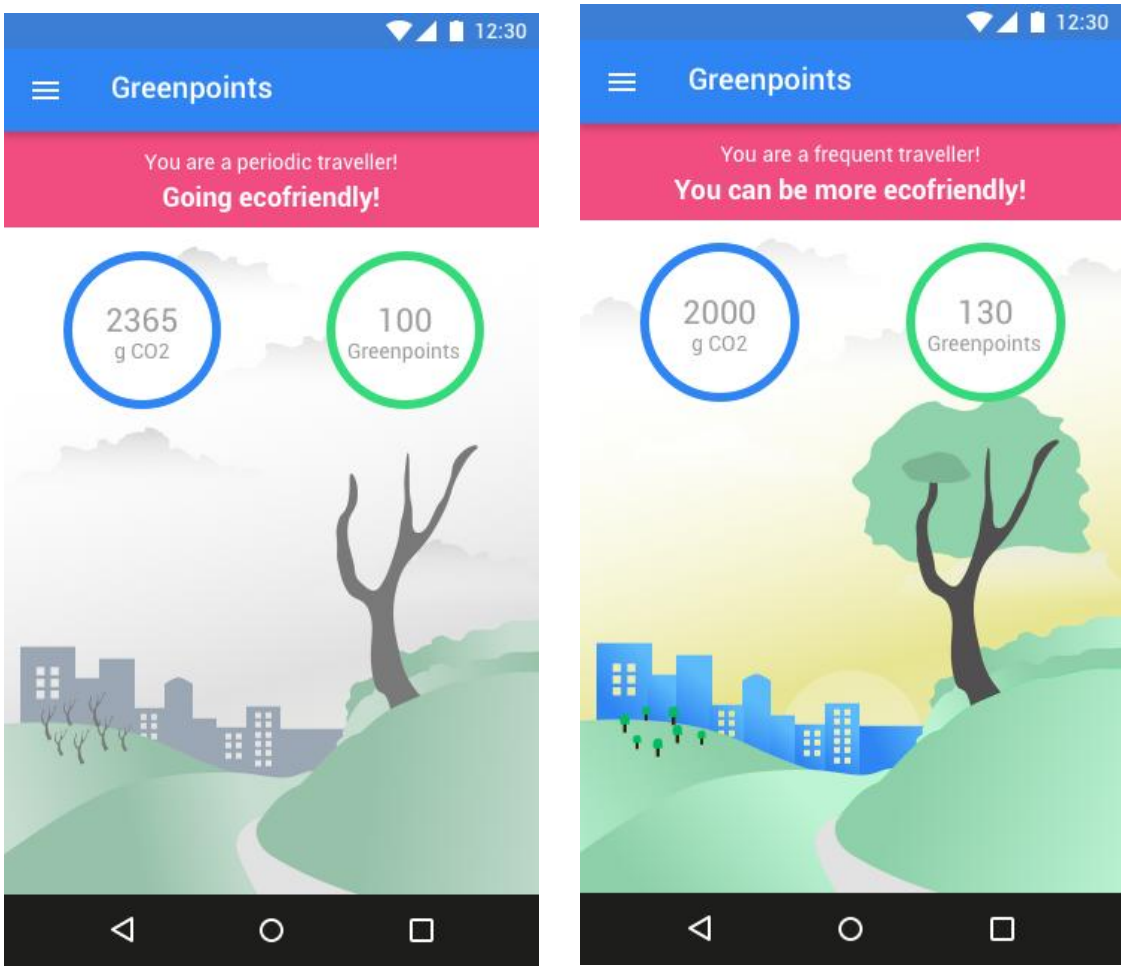
3.4.3 Scenarios given to the user

The following test is designed to test the system, not the subjects.
Describing thinking aloud, explain the setup:

BONVOYAGE is designed for frequent travellers who want to drive eco-friendly. It is a platform optimizing transnational multimodal door-to-door transport of passengers and goods, which integrates travel information, planning and saving trips through multiple countries.

1) You plan a trip from Bilbao to Oslo via Rome in two weeks. On your smartphone the BONVOYAGE App is installed. Describe your steps from saving to viewing your route.

2) Look at the printed screens (see Figure 16). What can you see? How do you interpret these things?



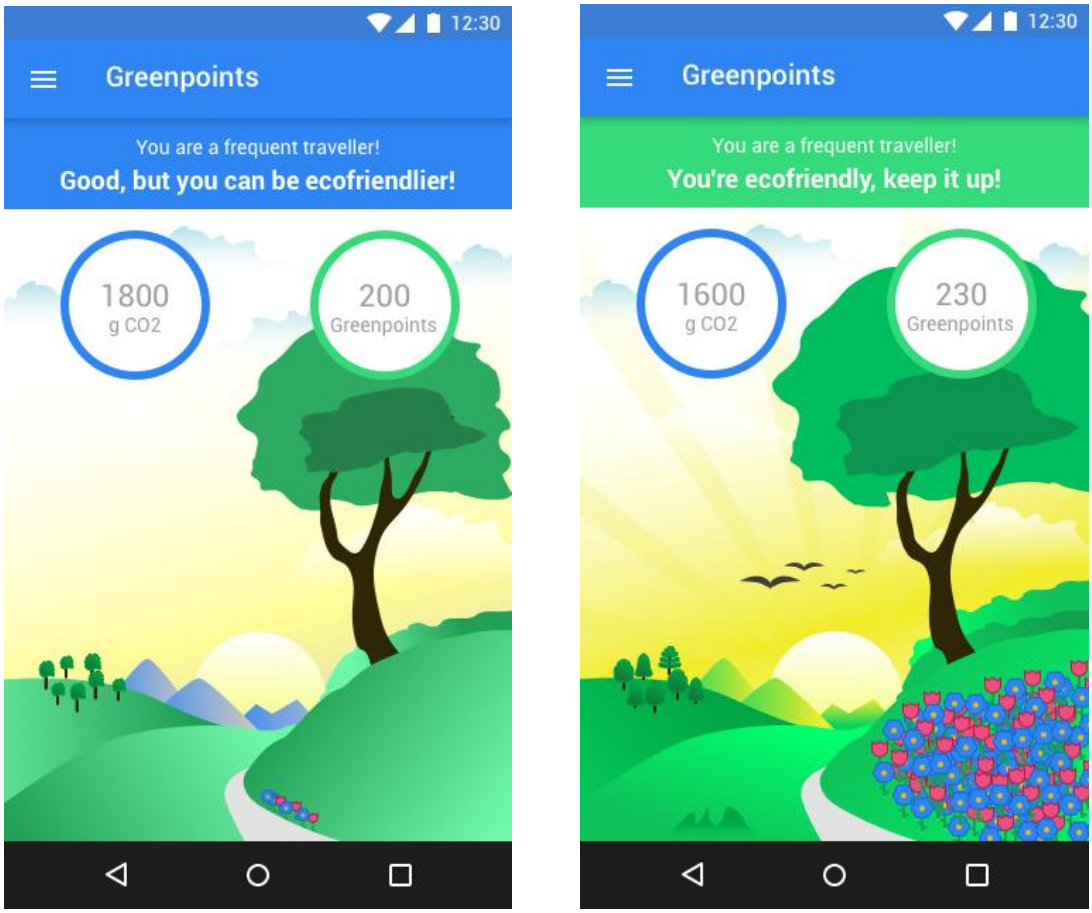


Figure 16. Screenshots of the App for Usability Testing

3) How do you like the route? Give feedback. Describe your steps.

4) How do you like the app? Give feedback. Describe your steps.

3.4.4 Test-Setting

- Android 6.0 Nexus 5
- Printed screens of the greenpoints
- Notepad and pencil for the tester
- Table sheet for documenting the start time and results

3.4.5 Sociodemographics and Background

Participants:

- Lukas Brand, Age 31, media experience with iOS, no knowledge about mobility
- Marvin Schwoiger, Age 30, average experience with Android, knowledge about mobility planning apps for travelling and developing
- Michael Pliskal, Age 26, average experience with Android and iOS, no knowledge about mobility planning apps

Statistics:

- 2 of 3 participants had experience with iOS
- 2 of 3 participants had experience with Android
- 1 participant had experience with both Android and iOS
- 1 of 3 of the participants had knowledge about mobility planning apps
- 2 participants had no knowledge about mobility planning apps

3.5 Results

POSITIVE ASPECTS	IMPROVABLE ASPECTS
Functionality: core functionality is clear after rather short time of exploring the app	Wording & Pictograms: in some places not clear or understandable
Design: nice look and feel, the association with the images at the Ecopoints screen, the star icon for save a trip, helps the user to find the saved trip in menu	Navigation Concept (some aspects): In some places improvable e.g. interaction from list to map
Navigation Concept (some aspects): easy to understand	

Table 5: Positive and improvable aspects

This section describes the found issues. Each issue has a description, prioritization and a recommendation on how to solve this issue. The issues are prioritized as follows:

- Prio 1: Threats - immediate action required!
- Prio 2: Weaknesses - take action!
- Prio 3: Opportunities - further improve!

3.5.1 Issue: Optimize route detail

ISSUE	PRIORITY	RECOMMENDATION
Two participants did not understand what "POINTS" in the route detail screen means. Two participants had the problem that the modality title was cut off / abbreviated (CITYB... instead of CITYBIKE) and they did not understand which service provider / modality was meant here.	Prio 1	New layout of the leg element on the routes detail screen. In the new layout, the complete modality name is presented in the first row and co2 and Greenpoints in the second row. In addition, the text ”POINTS” can be changed to “Greenpoints“, so it is more understandable for the user.

Table 6: Optimize route detail

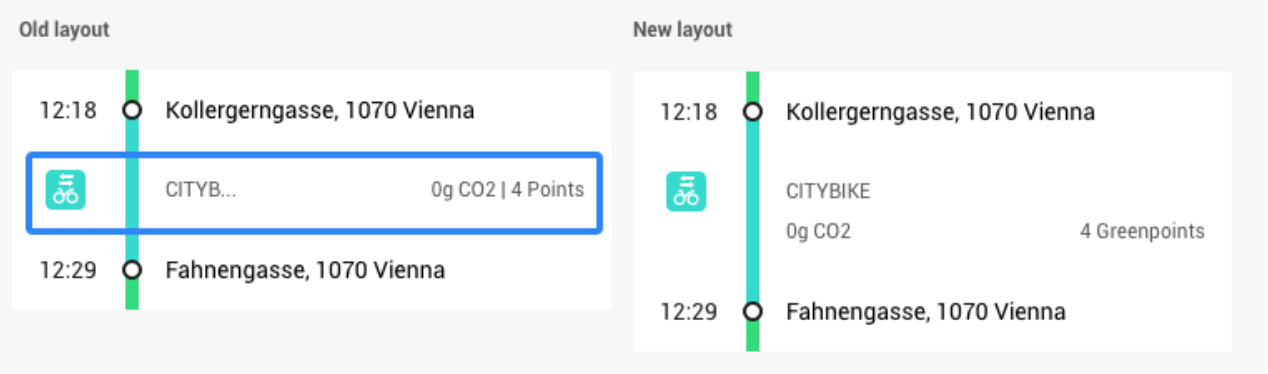


Figure 17. ***Recommendation “Optimize route detail”***

3.5.2 Issue: Explanation of Greenpoints

ISSUE	PRIORITY	RECOMMENDATION
All participants did not understand Greenpoints and did not know how they get Greenpoints. They did not understand what frequent and periodic travelers means or where this information comes from.	Prio 1	FAQs for Greenpoints Screen, here the user can get information about Greenpoints and travellers status.

Table 7: Explanation of Greenpoints

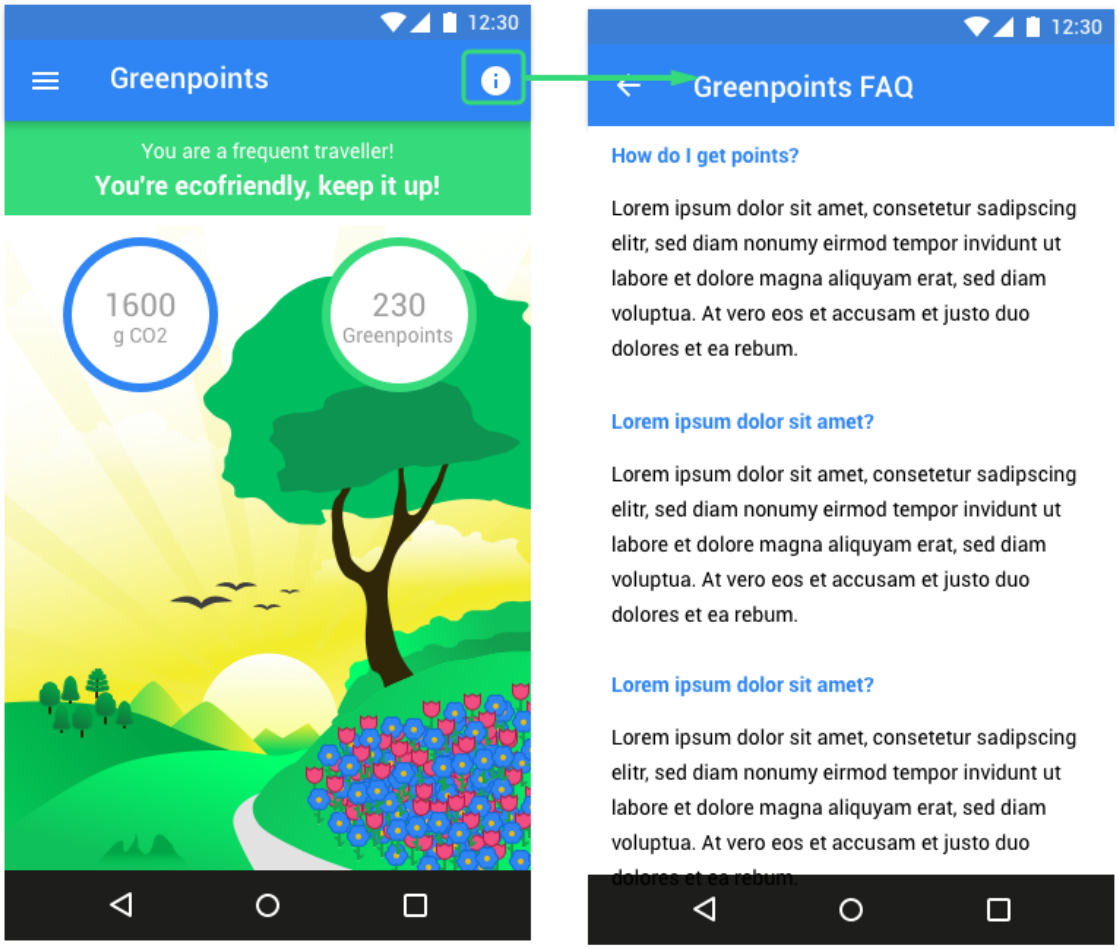


Figure 18. ***Recommendation "Explanation of Greenpoints"***

3.5.3 Issue: Interaction from list to map

ISSUE	PRIORITY	RECOMMENDATION
One participant tried to interact with a trip segment and expected to reach the map.	Prio 2	When the user clicks on the address in the list, he jumps to the map and zooms to the address on map.

Table 8: Interaction from list and map

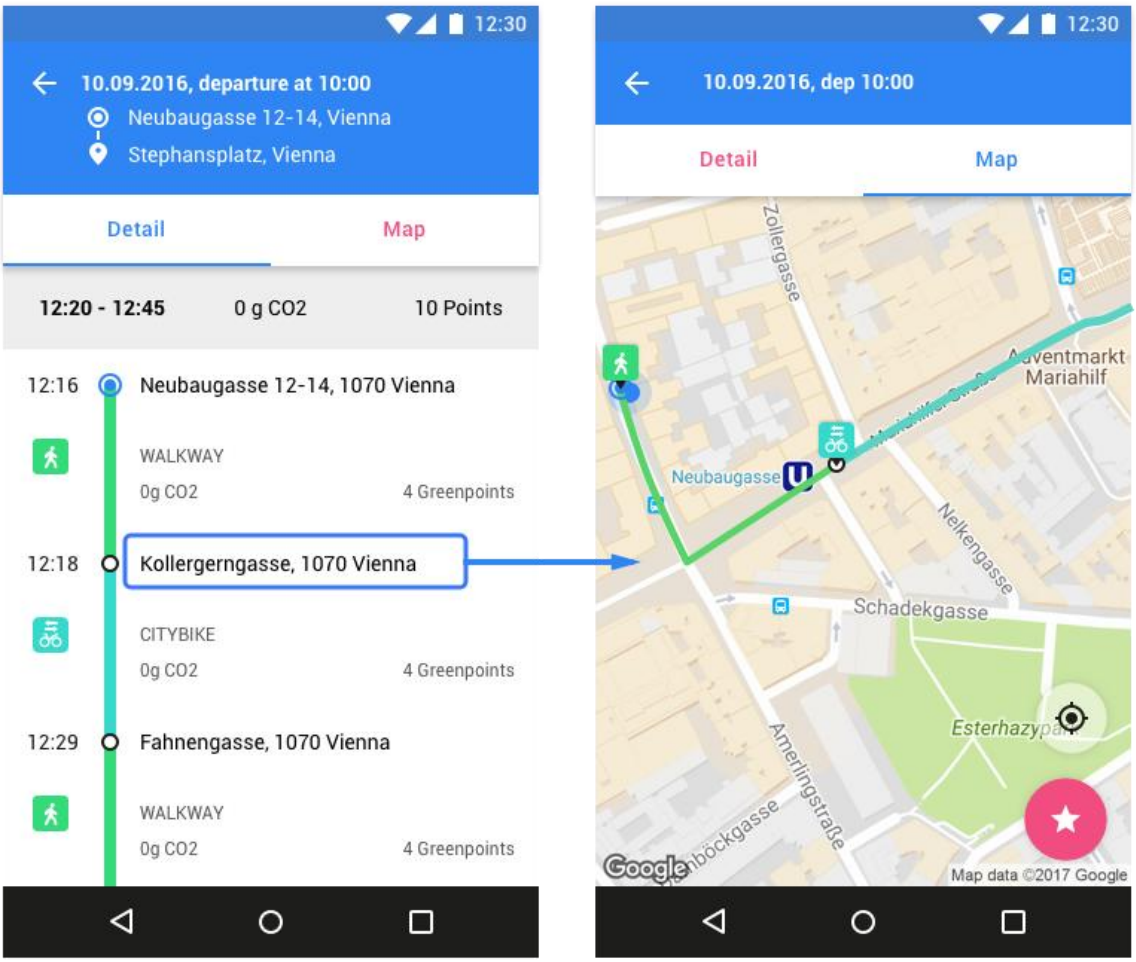


Figure 19. ***Recommendation "Interaction between list and map"***

3.5.4 *Issue: Naming menu item "Selected Routes"*

ISSUE	PRIORITY	RECOMMENDATION
The menu item "Selected Routes" was not clear because for saving a route, the user clicked on a star icon. They associated the star with favourites, so it would be better to name the menu item "Favourite Routes"	Prio 2	Rename the menu item to "Favourite Routes".

Table 9: Naming menu item "Selected Routes"

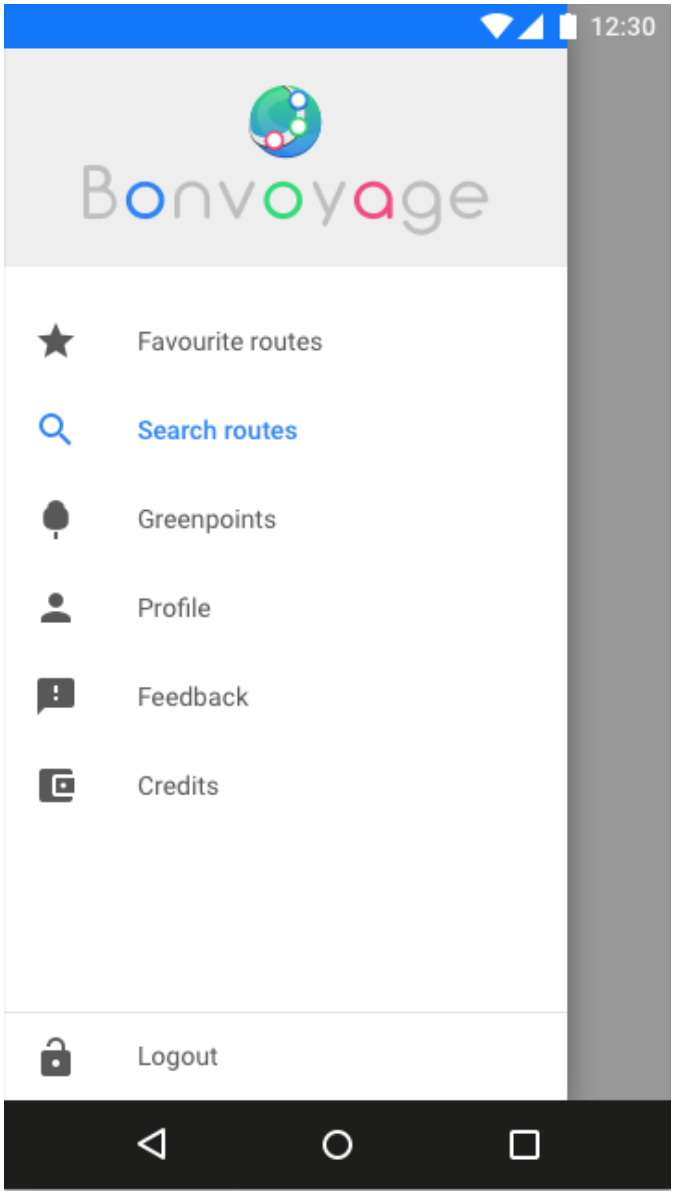


Figure 20. Recommendation *"Re-**Naming menu item Selected Routes**"*

3.5.5 Issue: Current location changing automatically

ISSUE	PRIORITY	RECOMMENDATION
The start field was automatically filled with the address of the current location. Two participants had the problem that the address was updated and changed every second.	Prio 3	Don't update the current location every time automatically, but only when user interacts with the app e.g. first start, after searching an address.

Table 10: Current location changing automatically

4 From BONVOYAGE scenarios to realistic load models

This section is dedicated to explain how realistic load models (user behaviour) have been extracted from BONVOYAGE scenarios in order to assess performances of the components of BONVOYAGE architecture.

Different realistic load models can be defined, depending on each component's specificity.

4.1 OpenGeoBase

In this section we will explain how load models have been extracted and used in order to assess performances of the discovery service.

Through the BONVOYAGE discovery service, which is based on the OGB component plus a standardized GeoJSON for ITS format (see deliverables D5.1 and D5.2), authorized users are allowed to publish geo-referenced information related to their data or planning services, in order to make them discoverable, and they can retrieve all information available for a particular area. This is necessary in order to plan an optimal multimodal trip.

In this Deliverable, as a test case, we consider geospatial data to be a collection of GTFS feeds stored in a dedicated instance of the OpenGeoBase spatial database. OpenGeoBase allows transit agencies to publish their data and solvers to fetch transit information.

In order to simulate a traffic load related to GTFS feeds being periodically updateed, which is as close as possible to the reality, we study updates history from https://transit.land.

Transit.land has steadily been growing as a reference spot for worldwide GTFS data, namely it is the spot for transit agencies to connect with developers. It is also built to expose schedule data from a single consistent database, which makes it even easier for developers to access schedule information.

Transit.land offers per-feed update history through feed_versions API available at:

https://transit.land/api/v1/feed_versions?feed_onestop_id=feed-ID

By using this API we can easily examine the workload due to feeds update.

Based on this information we can calculate that the day with the greatest number of updates during our tests was 2017-03-17 with 44 updates from 1:00:16UTC to 13:19:36UTC.

Before starting the simulation we inserted all feeds retrieved from Transit.land in the OpenGeoBase database (637 feeds) in order to have realistic data load in the database.

Performances and results are shown in section 5.1.

Regarding data fetching load scenario, our vision is the following: we suppose one soloist per country which fetches transit data about its entire own country once a day (from 00:00 to 01:00 am), in order to have updated transit data every day.

Considering the version history of transit feeds, we observe that using smaller refresh period is not convenient due to the slowness of GTFS updates rate.

Data fetching load scenario is simulated in OpenGeoBase by performing 40 polygon queries, one per European country. The polygon shapes visible in Figure 21, which are the ones we have used for our queries, are available at:

https://github.com/johan/world.geo.json/blob/master/countries.geo.json

Figure 21. Country polygon shapes we used for queries

Query performance results are shown in next section 5.1.

4.2 Membership management functional module

The features embedded in the Membership Management include the green policies score and loyalty program, which are used by BONVOYAGE every time a user connects to the platform and searches for travel solutions.

Considering in particular the Green Score Policy, every route that is proposed to the user is matched with a score that depends on how much eco-friendly is the travel solution. More details on the Membership Management module definition and implementation are detailed in next section 5.2.

5 Performance analysis

5.1 OpenGeoBase

The following Figure 22 shows processing time for each update: as described in the previous section the simulation scenario reproduces a real trace obtained from Transit.land feeds' version history.

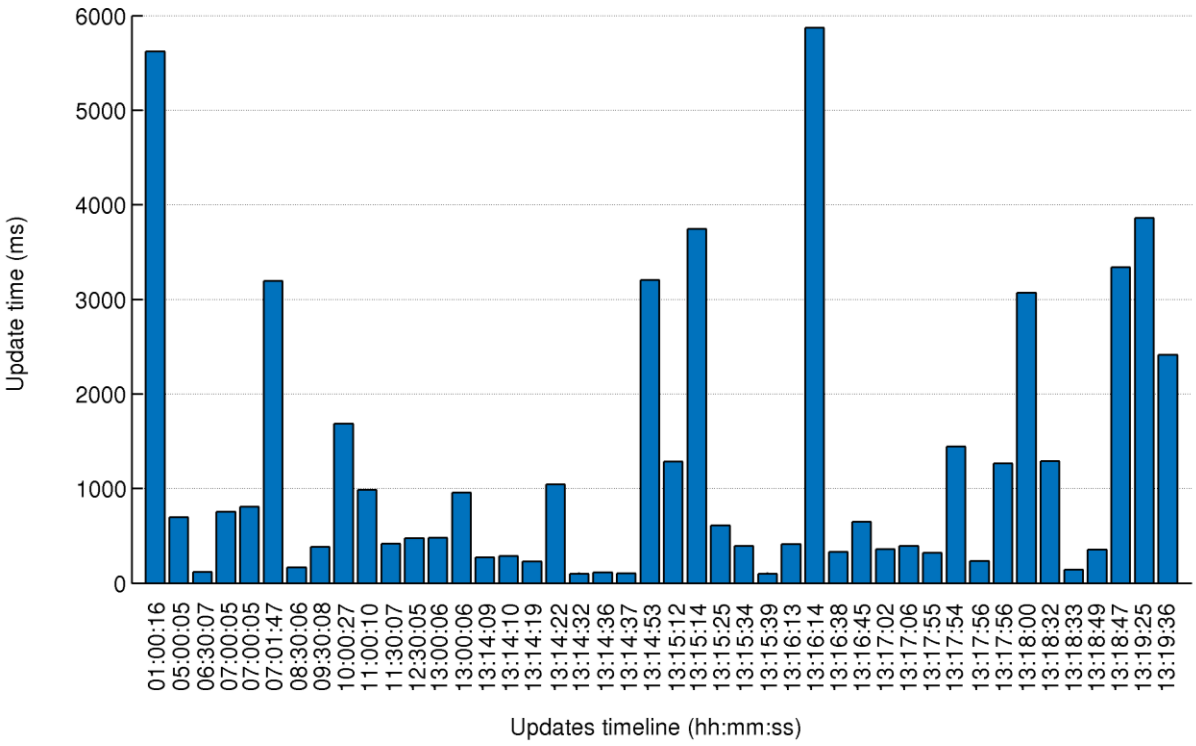


Figure 22. Results about the performance of Updates

The scenario was simulated performing 44 updates from time 01:00:16 to time 13:19:36, with an average inter-time of 17 minutes.

We can observe that the updates' processing time is not influenced by the inter-time between two updates, but only by feed dimension. Bigger update times are related to larger GTFS feeds, which may contain more than 10.000 stops to be processed.

Regarding the other load scenario, we simulated a soloist that wants to refresh transit feeds: in order to do it, the soloist sends to OpenGeoBase a polygon query, as described previously. We

can assume that the query is performed randomly from 00:00 to 01:00 (when the load is lower). We simulated 40 soloists, one per country.

The response time is shown in Figure 23.

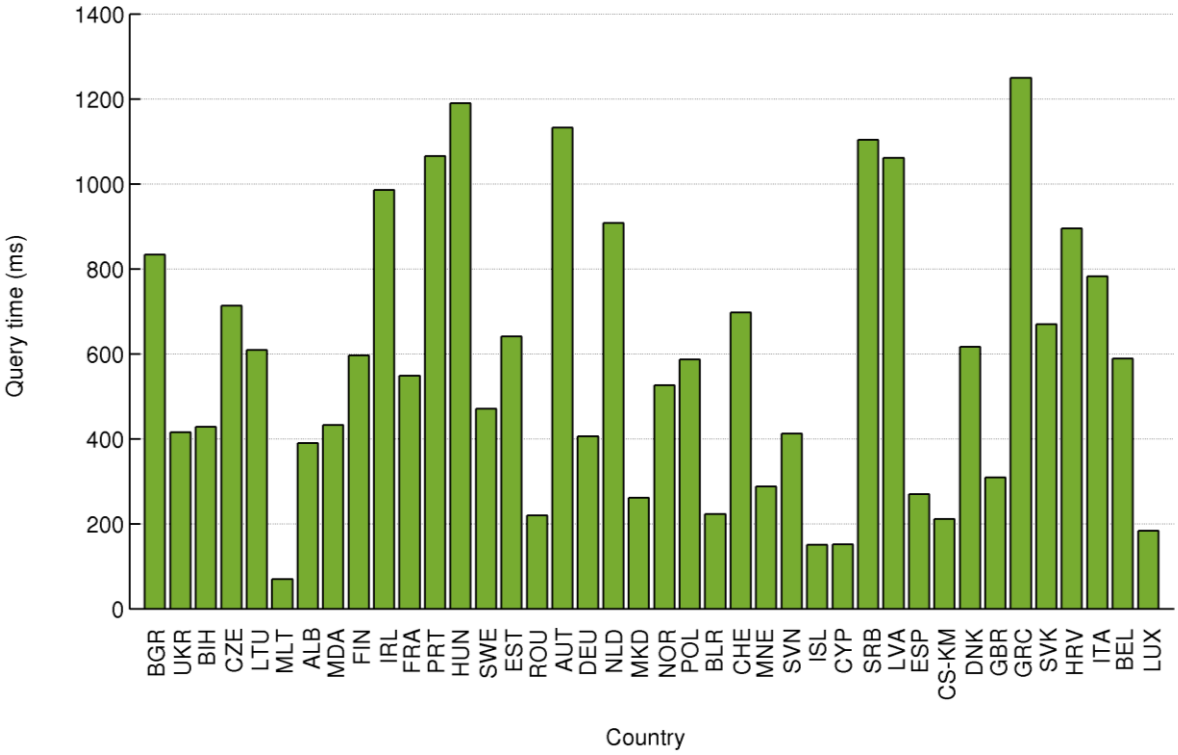


Figure 23. Results of performance of Queries

The above graph shows a non-uniform performance of the query time due to multiple factors:

- the size of requested area: larger queries need more processing time.
- the response size: the query response contains all GTFS feeds data within the country. Increasing the data response size, it will increase the data transfer time from OpenGeoBase database to the soloist.
- the shape of the polygon: complex shapes need more time to be processed.

Furthermore the graph shows that the simulated load that we have applied is not influencing the test, since load caused by the test average query time of 582ms and average query inter-time of 90sec does not, obviously, degrade performance.

5.2 Membership Management functional module

The BONVOYAGE Membership Management module, which manages the interaction with the end-users and tries to positively steer their behaviour, has been conceived combining a Green Score Policy and a Loyalty Program. Indeed, they both aim to reward users that use the BONVOYAGE platform to plan and purchase travel solutions. The difference between these two systems lies on their specific aims, namely:

- The Green Score Policy has the objective to "push" passengers towards eco-friendly mobility choices through the application of a set of scoring rules as well as a system of personalized incentives and penalties.
- The Loyalty program is intended to reward users that show a high level of “fidelity” to BONVOYAGE.

This Membership Management system has reached the stage of a prototype program that is stable in terms of its contents.

The performance of the Membership Management functional module in a real environment will depend on the number of users of the BONVOYAGE platform we consider. Indeed, this relevant data is highly variable depending on the target market and size (in a market or close-to market phase), or the size of the sample used to test its performance.

In the stable prototype version the Membership Management model is implemented as a module of the BONVOYAGE application server, hence performance is strictly related to those of the application server itself. It provides two external APIs for getting user information and it allows other internal services to calculate the green points for travel solutions. The APIs are described in detail in D6.1. The limitation factors are the deployment infrastructure, either the DB-access times or the number of connections that the standard web service can handle. Both can be easily scaled for larger number of users with horizontal scaling. When we consider the workflow for route information requested by users, the routing orchestration service with the route calculation are the limit factors of the performance.

As the Membership Management functionality is embedded within the Application Layer of BONVOYAGE, only qualitative performance analysis on the “Green Score Policy” of the Membership Management system was conducted, by isolating the specific system (in particular the “Green” Score policy) from the rest of the App Layer components.

In detail, for the two Membership Management systems the following results have been collected.

Green Score Policy

Green Score Policy has been tested in the real transport service scenario of the urban context of Rome. The final scope was to verify the effectiveness and impact of the mechanism on a heterogeneous sample of users. After a first testing round on 10 users[8], Trenitalia has faced up a second testing activity.

In particular, the second assessment was implemented considering a first group of users that visualized the score points associated to each travel option, whereas a second group had no evidences of the Score Policy.

In detail the test was realized on a sample of 12 users. Several user categories were considered: Students (age from 20 to 25), Employees (age from 30 to 40) and retired people (age from 50 to 65).

Of these testers:

- about 60% of them were exposed to the Score-policy
- whereas 40% of testers were not exposed to the Score-policy

with the aim to compare the behaviour of users exposed to the score policy incentive mechanisms vs. behaviour of those that were not.

Final results confirm that the Green Score Policy works appropriately in terms of definition of the points to be associated to each travel choice and of performances over a period of use; moreover, relevant trends were appreciated in travel Eco Friendly preferences of the "Testers" that were aware about the Green Score Policy which can legitimately be associated to the impact of the policy incentive. Therefore, the score policy rewards the virtuous and sustainable behaviours of citizens in their transport solutions' choice.

Testing instances and results details are reported in Chapter 6 of D 4.2. In the following Figure 24 and Figure 25, an extract of the most relevant testing results has been reported (for full details please refer to D4.2).

[8] Andrea Iecher , "Score Policy in Intelligent Trasportation System for promoting ecofriendly behaviours", Thesis discussed at the Faculty of Information Engineering, Informaticsm and Statistics, La Sapienza Università di Roma, Academic year 2016/2017, pag 56-62.

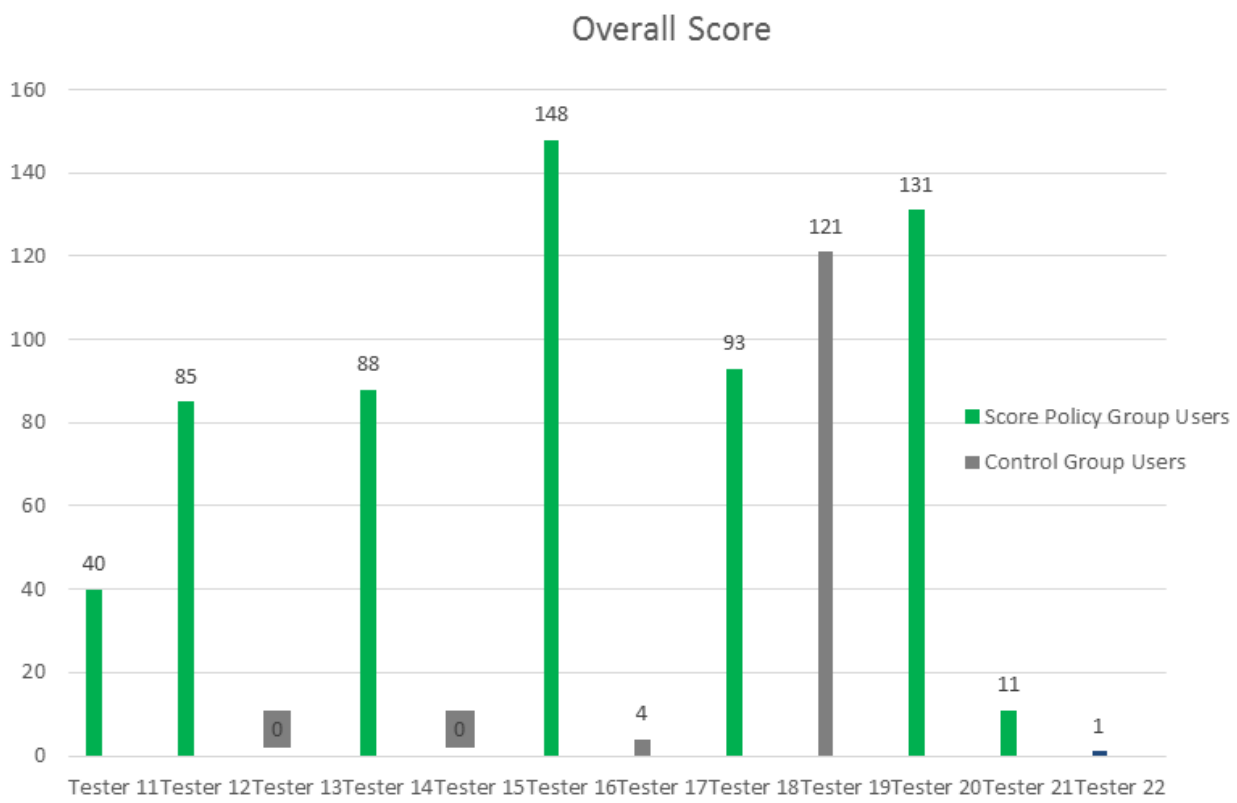


Figure 24. Overview of score accumulated by TEST2 users[9]

[9] Figure 58 of BONVOYAGE D4.2 "Development and validation of the Intelligent Transport Functionality".

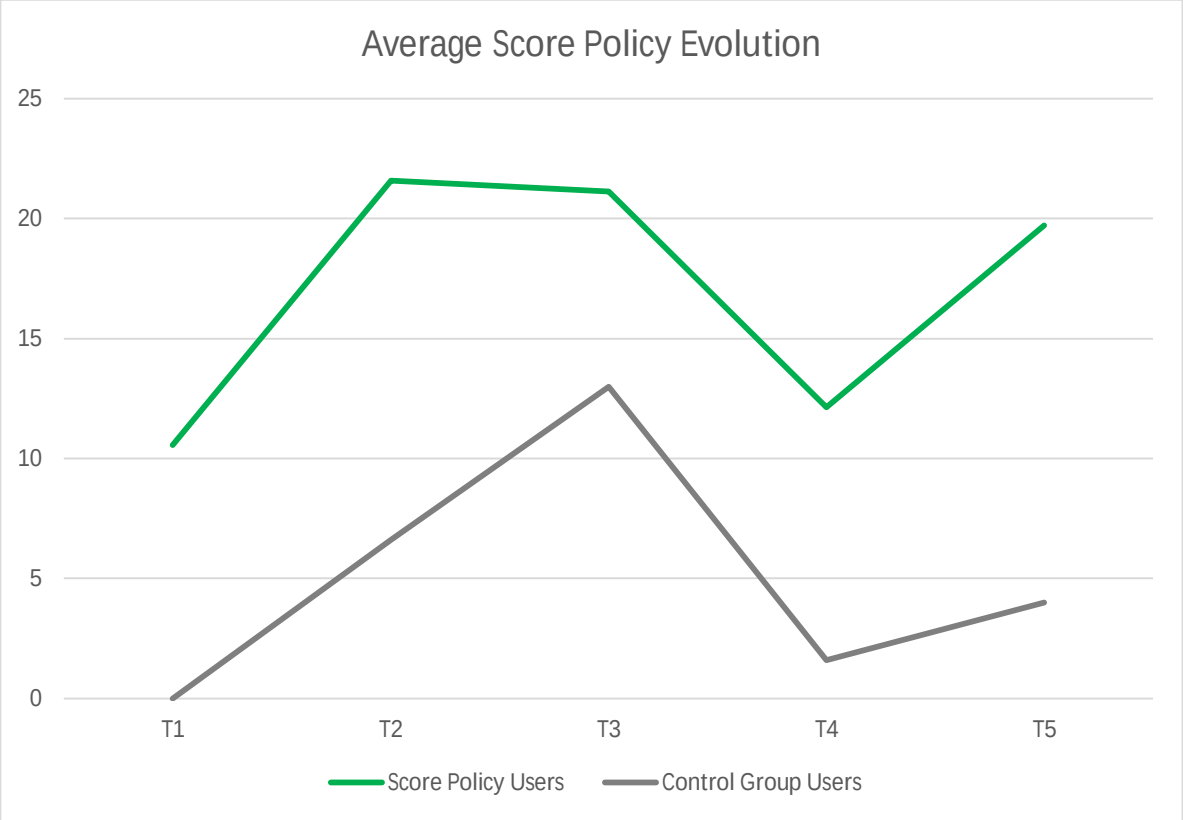


Figure 25. Score policy evolution in the TEST2[10]

Loyalty Program

Loyalty program has been developed in a proof-of-concept version, which is detailed in Chapter 6 of D4.1. Starting from the state-of-the-art analysis and the benchmarking evaluation of the most relevant transport loyalty programs, the BONVOYAGE Loyalty Program has been developed considering the specific characteristic of BONVOYAGE to act as door-to-door journey planner for the "End User".

Although performance of a loyalty program in proof-of-concept state, lacking real-life data about people's usage of the deployed system, is difficult to assess, we speculate that the final product's performance is be able to overcome the state-of-the-art and can be considered innovative because of the provision of profiled and customized awards, prizes and promotions. These features that are exactly relevant to the overall BONVOYAGE goals to provide a journey planner that always takes into account user's personal profile.

[10] Figure 59 of BONVOYAGE D 4.2 "Development and validation of the Intelligent Transport Functionality".

5.3 Personalization modules

Personalization of trip planning and monitoring services has been widely investigated and developed in WP4. Five software modules have been implemented and tested in scenarios when the volume of user requests in need to be personalized gets (very) large. Specifically, methodologies and validation results have been reported and detailed in two deliverables: D4.1 and D4.2. Results included test performances in realistic scenarios for off line procedures, we we think are the most challenging ones from the efficiency point of view.

In this contribution, test performances in realistic scenarios are reported in line with other BONVOYAGE Deliverables (i.e., D4.2 Section 4, D4.2 Section 5.2).

The personalization modules have been implemented as PHP web services exposing RESTful APIs services that allow them to be triggered. The personalization modules involve two different on-line services: Online User Profiling and Rank Tool. The two are the most active modules in realistic scenarios, since they are actively recalled with highest frequency in scenarios of practical interest.

The Online User Profiling is in charge of identifying in real time a set of personalization parameters to tailor the routes for each user, in particular it receives the user query and returns the set of personalized parameters.

The Rank Tool is in charge of ranking the computed travel solutions, proving a personalized ordered list of travel solutions, in particular it receives the set of un-ordered travel solutions and returns the set of travel solutions proving the suitable order.

In order to estimate the response time of each module (i.e., computational time used by modules for proving outputs) we have deployed a dedicated service in a private cloud virtual machine (provided by the Cloud Service Provider 1&1) with Apache server running Linux Ubuntu 16 with 16 virtual CPU, 8GB RAM and 120GB of SSD.

We have performed several different tests for evaluating key performances. Table 11 shows the average time consumed by the Online User Profiling averaged over 1, 10, 100, and 1000 test instances when receiving one request at the same time. The mentioned average time is the time needed by the service for reading the input, and returning the corresponding output.

For this service the average time includes the creation of suitable JSON file in SPROUTE format too.

NUMBER OF ITERATIONs	Avg. ONLINE USER PROFILING
1	309 ms
10	307 ms
100	353 ms
1000	336 ms

Table 11: Test Performances for Online User Profiling

Table 12 shows the average time consumed by the Rank Tool averaged over 1, 10, 100, and 1000 test instances when receiving one request at the same time with a variable number of travel solutions at each iteration. Please consider each iteration to have a random number n of travel solutions where n is between 2 and 7. Column "Avg. Number of Travel solutions" represents the average number of travel solutions randomly generated for the family of test instances and used to evaluate the corresponding order. The mentioned average time is the time needed by the service for reading input data, evaluating the weights, ordering the set of unordered travel solutions and returning the output. For this service the average time includes the creation of suitable JSON file in SPROUTE format too.

NUMBER OF ITERATIONs	Avg. RANK TOOL	Avg. Number of Travel Solutions
1	391 ms	3
10	318 ms	2,3
100	600 ms	5,3
1000	549 ms	4,8

Table 12: Test Performances for Rank Tool

Rome and Bilbao Urban Soloists

The test performances about Rome and Bilbao urban soloists have been presented in BONVOYAGE deliverable D4.2 Section 5.2.

6 Conclusion and future work

In this deliverable, steps of the implementation of user's sensing functionalities (transport mode recognition and stress level monitoring) in the Android environment are described and validation and performances tests of these algorithms, when embedded in an Android smartphone, are reported. These algorithms have been integrated in the BONVOYAGE App.

Further, usability tests of the BONVOYAGE App have been performed on three subjects and their results are described for testing and validating the final BONVOYAGE App.

Moreover, we have identified load models and users' behavioural patterns derived from BONVOYAGE realistic scenarios and reported performance tests for OpenGeoBase and the membership management modules, as singled-out parts of the whole BONVOYAGE platform.

Finally, results about performance of the personalization module are also reported in this deliverable.

All of the above performance analyses of the BONVOYAGE App and of the components that directly impact it are the prerequisite step for the validation, evaluation and usability tests of the integrated BONVOYAGE platform, which are planned within (and part of) the integration work-package, that is WP7.

7 References

[1] "MATLAB - Le langage du calcul technique." [Online]. Available: https://fr.mathworks.com/products/Matlab.html. [Accessed: 30-Mar-2017].

[2] "MATLAB Coder - MATLAB." [Online]. Available: https://fr.mathworks.com/products/Matlab-coder.html. [Accessed: 30-Mar-2017].

[3] "Android NDK | Android Developers." [Online]. Available: https://developer.android.com/ndk/index.html. [Accessed: 30-Mar-2017].

[4] "Weka (machine learning)," *Wikipedia*. 22-Mar-2017.

[5] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, vol. 103. New York, NY: Springer New York, 2013.