Use Case Diagram Based Scenarios Design for a Biomedical Time-Series Analysis Web Platform

Alan Jovic[*], Davor Kukolja[*], Kresimir Jozic[**] and Mario Cifrek[*]
[*] University of Zagreb Faculty of Electrical Engineering and Computing, Zagreb, Croatia
[**] INA - industrija nafte, d.d., Zagreb, Croatia
Corresponding author: alan.jovic@fer.hr

Abstract - Biomedical time-series analysis deals with detection, classification and prediction of subjects' states and disorders. In this paper, we present requirements and scenarios of use for a novel web platform designed to analyze multivariate heterogeneous biomedical time-series. The scenarios of use are described with the corresponding UML Use Case Diagrams. We also discuss some architectural and technological issues, including parallelization, visualization and feature extraction from biomedical time-series. The goal of this paper is to present what we currently consider as the best approach for design of such a system, which may also be beneficial for similar biomedical software systems. The paper is focused on design and architectural considerations only, as implementation of the complex system has only just begun.

I. INTRODUCTION

Web and mobile applications development for biomedical services is continuously growing in the healthcare community [1,2]. While the developed software aims either at general population users [2] or at medical professionals for continuous monitoring of patients [3], there is very little interest involved in developing an integrative web platform that could benefit medical professionals, researchers and experienced general users in modeling subjects' states and disorders.

The goal of the currently running Croatian Science Foundation research project HRZZ-MULTISAB[1] is to develop a high quality system for an all-encompassing analysis of multivariate heterogeneous biomedical time-series. It is conceived that this system would be used by all interested users, including medical doctors, biomedical engineers, computer scientists, and others, depending on their goal. As the system will be developed as a web platform, the users will be able to access it from afar, which differentiates it from local or hospital-specific software solutions.

The problem that the project tackles at its core is the hard problem of efficient biomedical time-series features identification. There have been many efforts put forth to discover and describe both domain-specific and general time-series relevant features [4-6]. The platform that we are currently in the process of developing will offer significant advancements in this respect, as we will consider the use of time-series features there were proven to be effective in a wide range of sciences [6], as well as in specific biomedical signal domains [7].

Once the platform's implementation is completed, we plan to investigate and compare the best feature combinations advocated by domain experts with the best general time-series features and with no-features engineering approach using deep learning approaches [8]. The goal is to achieve as accurate as possible models of subject's states, including various medical disorders.

Presently, in this paper, the goal is to describe system requirements for such a web platform and to elucidate the way in which the platform will be built. We focus on the requirements, with a brief overview of the system's architecture and the technologies involved.

II. SYSTEM REQUIREMENTS

System requirements, which were agreed upon by the entire project working group were the following:

- An integrative software solution for the analysis of multivariate heterogeneous biomedical time-series.
- Software solution implemented in the form of a web platform, as thin client - fat server.
- Software logic layer on the server written in Java programming language.
- Interface towards the user implemented with a set of contemporary web development technologies (HTML5, CSS3, TypeScript...).
- Multiple input file formats: European data format (EDF) and EDF+, textual format for signals and annotations, images formats. Files that contain meta-data (anamnesis, disorder annotations, etc.)
- Visualization of signals in 2D (records inspection) and specific body disorders in 3D using graphical hardware.
- Biomedical time-series preprocessing, such as signal filtering, R peak detection in ECG series, data transformations in: time, frequency, and time-frequency domains.
- Biomedical time-series feature extraction - features would be chosen by: 1) a medical expert system implemented in the platform, which would be based on current medical knowledge from guidelines and

[1] This work has been fully supported by the Croatian Science Foundation under the project number UIP-2014-09-6889.

relevant scientific papers, 2) an expert user, manually; a large number of features need to be supported by the platform, both general and domain-specific biomedical time-series features.

- Feature selection methods, classification, regression, and prediction machine learning algorithms should be used to construct accurate subject state models.
- Results reporting in contemporary formats.

The platform will necessary support the analysis of the following biomedical time-series:

- ECG – electrocardiogram – cardiovascular activity
- HRV – heart rate variability – a series of RR-intervals whose variability is analyzed
- EEG – electroencephalogram – brain activity
- EMG – electromyogram – muscle activity

Additionally, optionally, the platform will support:

- electrooculogram (EOG), electroglottogram (EGG), cardiotocogram (CTG), electrocorticogram (ECoG), photoplethysmogram (PPG), pressure (arterial blood pressure - ABP, pulmonary artery pressure - PAP, central venous pressure - CVP), respiration (impedance), oxygen saturation (SpO2), CO2, gait rhythm, galvanic skin resistance, etc.

The platform will support multivariate heterogeneous analysis (e.g. ECG + HRV + SpO2).

In the implementation phase, the focus will be primarily to demonstrate the working capabilities of the platform on a smaller set of signals and medical disorders.

III. UML Use Case Diagrams for System Requirements Visualization

System requirements have been detailed in the form of UML Use Case Diagrams, v.1.4+ [9], drawn in Astah Community edition tool [10], which present how a user would interact with the system. These diagrams provide a behavioral description of a system's use, without insight into a detailed temporal sequence in which the interaction with the system is performed. Regardless of the absence of a temporal view on the interaction process, all the scenarios of use can be easily identified. This facilitates web platform development.

There were eight UML Use Case Diagrams that we identified, which describe the phases of the analysis:

1. Analysis type selection
2. Scenario selection
3. Input data selection
4. Records inspection
5. Records preprocessing
6. Feature extraction
7. Model construction
8. Reporting

Some of these phases may be skipped entirely, depending on the user.

Additionally, the 9. diagram, User account, depicts registration, login and profile management. Finally, the 10. diagram, Platform administration, shows the administrative access to the web platform.

In the subsequent subsections, we present the diagrams and their detailed explanation.

A. Analysis Type Selection

Analysis type selection, Fig. 1, is conducted in a way that a user chooses the goal for an analysis that he wants to perform: detection, classification, prediction, or inspection and visualization. He also chooses the data type on which the analysis will be based. The data type may include exclusively biomedical time-series or biomedical time-series with additional domain data (e.g. subject anamnesis, metadata which describes record type, etc.).

Biomedical time-series may be univariate by type, which means that there is only a single measured data array available in the record, e.g. the times of heart beats, or multivariate, which means that there are several measured data arrays present in the record. Additionally, multivariate data may he homogeneous, which means that they come from the same source measurement device, e.g. EEG data, or heterogeneous, which means that there are more than one measurement device involved at the same time, e.g. ECG + HRV + systolic ABP. An example of analysis type selection would be classification based on EEG data.

B. Scenario Selection

Scenario selection, Fig. 2, enables the user to select a predefined scenario, based on previously selected analysis type. It also allows the construction of a completely new scenario through full customization. A list of predefined analysis scenarios with all of the corresponding aspects (used components) will be provided to the user. The user will be able to confirm a scenario or change some of its aspects, such as preprocessing methods, feature extraction methods or model construction methods. When defining a

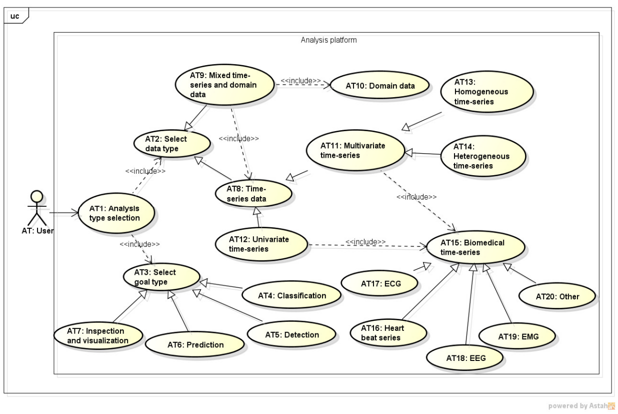

Figure 1. Analysis type selection.

new custom scenario, or changing an existing scenario, the user will be able to select any of the methods for accomplishing his goals. The predefined scenarios will contain features that are to be extracted from signals, which will be selected by the medical expert system specifically designed for this purpose. The medical expert system will be implemented based on current relevant medical guidelines, standards, and relevant medical and biomedical engineering literature. An example of a scenario would be to detect epilepsy in EEG, using wavelet transform modulus maxima and correlation dimension features, and a C4.5 decision tree.

C. *Input Data Selection*

Input data selection, Fig. 3, includes the choice of file extensions (or file formats) that will be uploaded for the analysis. These formats include EDF, EDF+ (with or without annotations), separate textual annotation files (e.g. heart beat annotations), textual raw signal files (without annotations) and image files (e.g. JPEG, TIFF). The user will have to select the appropriate file format in order for file upload to work without errors. Aside from selecting the format, the user also specifies the files that will be uploaded or selects the folder from which all the required files can be found. If there are some specific details related to records loading into platform (e.g. only partial file loading, limitations for some image formats, etc.), these can also be specified during this phase.

D. *Records Inspection*

After the records are uploaded into the platform, it will be possible to inspect and visualize (Fig. 4) each loaded record through the platform's graphical user interface. Here, the user will be able to: select specific record segment, select temporal and amplitude scaling of the signals, display some or all of the signal trails (arrays), inspect record headings and domain data (if such information exists). 3D visualization will be enabled for certain specific states for some biomedical signals (e.g. visualization of myocardial infarction location based on ECG). Additionally, the user will be able to annotate some of the available signals or change the existing annotation files and save these changes.

E. *Records Preprocessing*

Records preprocessing, Fig. 5, assumes various procedures for signal filtering and data transformations. Signal filtering includes noise filtering (e.g. notch filters for 50 / 60 Hz), baseline wandering corrections (e.g. for ECG because of breathing, body movements or electrodes impedance changes), records segmentation into windows of certain width. Window width may be chosen by the user or it may be implicit, determined by the very features that are to be extracted (e.g. a single RR interval for detailed ECG analysis). Other filtering methods include various linear and nonlinear filtering procedures with the goal to obtain higher quality signals.

Data transformations will be performed after the filtering procedures and will include various time (e.g. principal component analysis - PCA), frequency (e.g. fast Fourier transform, Hilbert Huang transform), time-frequency (e.g. wavelet transform) and other types of data

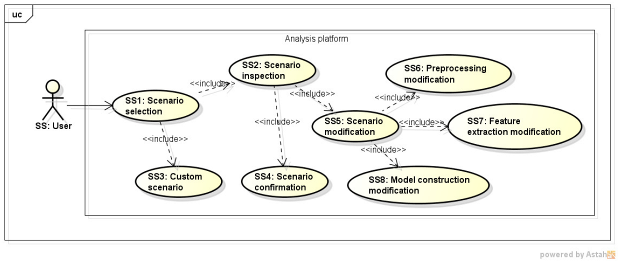

Figure 2. Scenario selection.

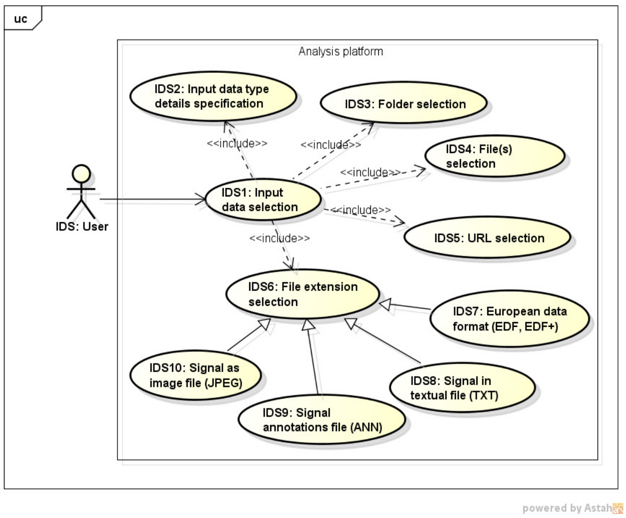

Figure 3. Input data selection.

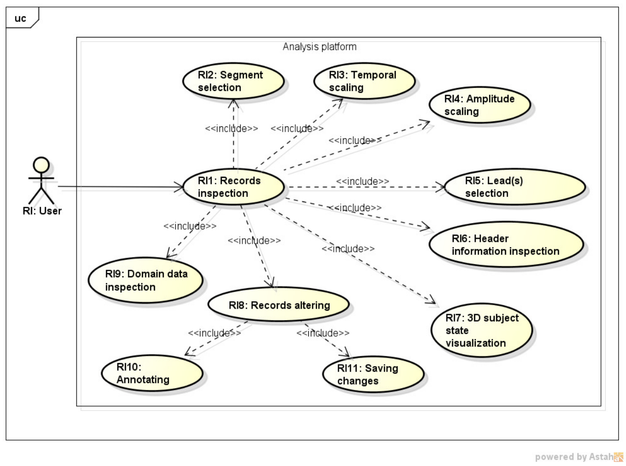

Figure 4. Records inspection.

transformations. The goal of data transformations is to obtain the data in a form from which it is easier to calculate precise domain or general features for describing specific subjects' states.

F. *Feature Extraction*

Feature extraction, Fig. 6, is the central step in the analysis of biomedical time-series. The user will already be provided with a list of features in the scenario selection phase. Regardless of the scenario, in the feature extraction phase, it will be possible so select additional features and

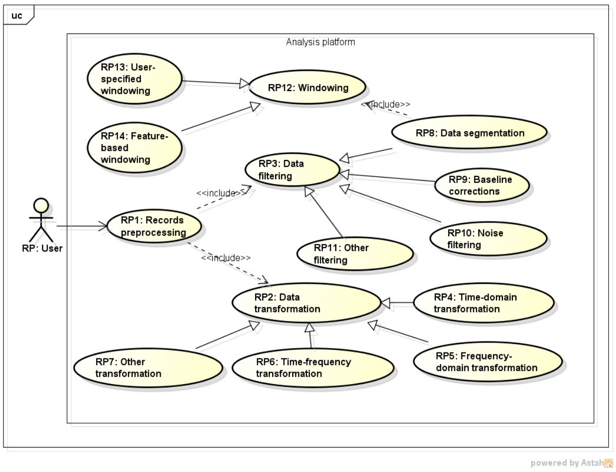

Figure 5. Records preprocessing.

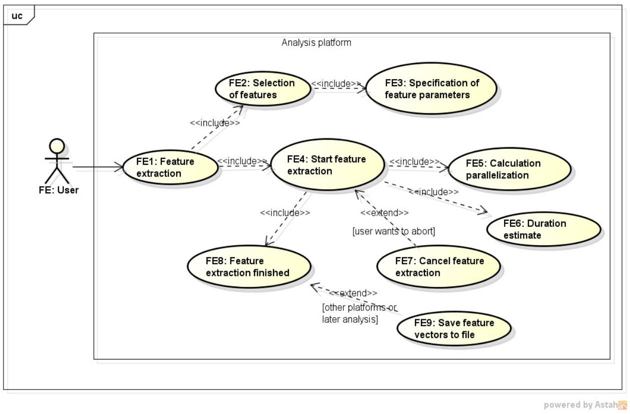

Figure 6. Feature extraction.

specify parameters for calculation of a particular feature, for those features that are parametric. When starting feature extraction, the user will be able to inspect the information about the estimation of the analysis duration and will be able to modify the type of calculation parallelization that will be performed. It will be possible to cancel feature extraction execution at any time. The user will be allowed to save the extracted feature vectors in a file for future analysis, if he wants to have the intermediate results recorded.

G. *Model Construction*

Model construction, Fig. 7, is a complex step in the analysis of biomedical time-series that takes place after feature extraction and includes various machine learning algorithms for the analysis of the extracted feature vectors. In this step, the user will be able to load the file with feature vectors that were calculated and saved earlier and resume the analysis or continue the analysis from the recently obtained and unrecorded extracted feature vectors. Model construction phase starts with the option to select feature vectors' preprocessing methods. Herein, one can use feature vectors' dimensionality reduction methods such as various transformations (e.g. PCA) or feature selection methods (filters, wrappers, embedded, and other methods [11]).

Dimensionality reduction is necessary in the case where a large number of features were extracted (either by the choice of the expert system or by the user's selection). The goal of dimensionality reduction is to keep only relevant and non-redundant features in order to improve speed and accuracy of model construction. Aside from feature space dimensionality reduction, it will be possible to use some methods to alter the feature vectors themselves (e.g. re-sampling, missing values replacement, etc.).

After the feature vectors preprocessing step, the user will be able to select a model construction method. During this step, first a method is selected and then, the method parameters are defined. Method parameters selection will be made possible if the modeling method is parametric (e.g. C4.5 decision tree, random forest).

Before starting model construction, it is necessary to specify the learning method that will be used: holdout

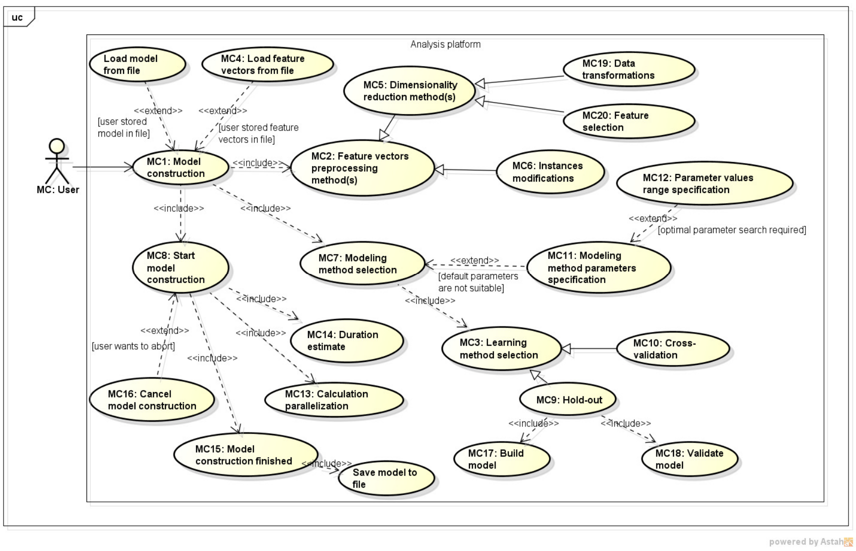

Figure 7. Model construction.

procedure, where a part of dataset is used training and the other part for testing, or cross-validation, where testing is performed on dataset segments so that all the dataset is eventually used for testing. It will also be possible to perform only model testing on a set of new samples, if there is a model present in the system that was trained earlier on the same feature set. When starting model construction, the user will be able to inspect the information regarding the estimation of the construction duration and will be able to modify the type of computing parallelization that will be performed. The construction procedure could be cancelled at any time. After model construction, it will be made possible to save the trained model into a file for later use.

H. Reporting

Reporting, depicted in Fig. 8, is the step in which a user chooses the way in which the results of the analysis are displayed. Thereafter, the report is displayed in the selected form (on a web page only, in PDF, MS Excel, Word or ODF form). It will be possible to save the copy of the report onto the client's computer, whereas its copy will be stored on the server. Additionally, the user will be able to grant access to the report to other registered users, at his convenience.

I. User Account

An account for a new user will be opened by registering the user onto the platform, which will include entering the data about the user (necessary data: user name, password and e-mail address; additional data: first and last name, title, affiliation, affiliation's address, telephone), Fig. 9. Logging into the system will be allowed after confirmation that the registration was accepted. Profile management will be enabled after a successful login. It will include editing personal information, accessing generated (finished) reports that are linked to the user or resuming the last analysis conducted by the user.

J. Platform Administration

Platform administrator will have to login into the platform in the same way as the user, by entering his user name and password, Fig. 10. The administrator will have at his disposal a number of options. First, he will be able to edit his personal profile. He will also be able to manage user accounts for all the platform's users. This will include opening a new user account (as if a user has registered), confirming the registration of a new user, and removing an existing user account. For each account, administrator will be able to inspect the user's personal information (except password), without the possibility of modification.

The administrator will be able to inspect and search all of the reports generated in the platform. Also, he will be allowed access to all of the other files stored in the system, including all the saved input files, all the feature vectors files, and all the constructed models file. He will be able to delete them at his convenience. Moreover, the administrator will have access to all platform's logs, which will be sorted by date. The logs will contain relevant information about the users' actions during the use of the platform. Additional information about the platform as well as its usage statistics will also be at his disposal.

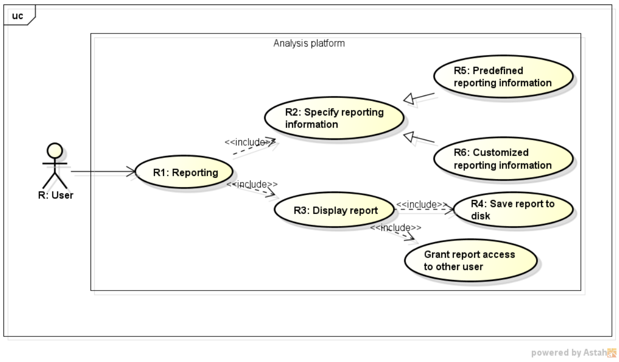

Figure 8. Reporting.

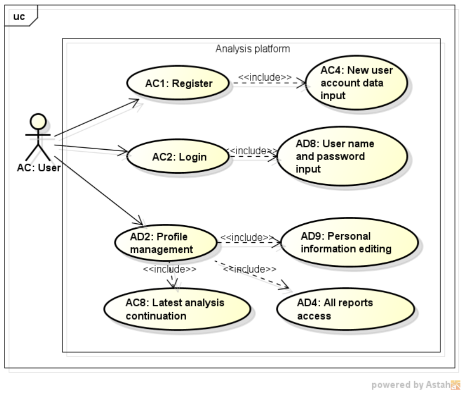

Figure 9. User account.

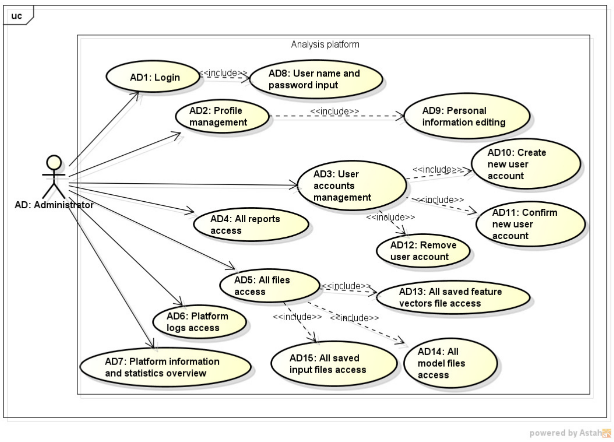

Figure 10. Platform administration.

IV. PLATFORM ARCHITECTURE

A. Web Technologies

System architecture was envisioned as a web portal, which will be accessed by the users through their browsers. This type of architecture was selected, as it is necessary to enable access to the platform from any remote location. With this, the range of potential users is widened, which is important for platform recognition, acceptance, and breadth of applications.

On the client hand side, during platform development, HTML5, CSS3, and Typescript languages will be used for designing web pages. As aids in developing web pages, Angular 2 framework will be used for programming and Bootstrap framework will be used for better visual experience and platform responsiveness. Also, WebGL technology will be used for 3D visualization of human body sections that are of interest to the user.

On the server hand side, Java programming language will be used. This language was chosen because of a large number of existing signal processing libraries, data parsing, implemented machine learning algorithms, ease of web development, and efficient parallelization support. Server side will communicate with the client side through the RESTful protocol. In order to lower the requirements on the server's resources, we will test the use of 1) a standard Java application server with a minimum set of options, and 2) a stand-alone framework that supports the required functionality (e.g. Spring Boot). A minimum set of options will include a user authentication library and an object relational mapping library (e.g. Hibernate).

For reporting, JasperReports® Library, an open-source Java library that supports all the required formats (including HTML) will be used.

B. Architectural Details

The developed architecture should be both modular and scalable. Its modularity should enable the development of various add-ons, while its scalability should support the concurrent work of many users. The platform should also support two important aspects of contemporary software: 1) parallelization and 2) dynamic module loading.

Parallelization will be used to accelerate calculations, which is especially efficient for the algorithms that can be divided into independent execution sections, without the need for synchronization. Parallelization will be enabled in two phases of the analysis: for feature extraction and for model construction, because these phase are the most computationally demanding. We may also consider employing it for records preprocessing, if deemed necessary. In any case, the goal will be to maximize the resource capacity on the server with respect to the number of available CPU cores, as well as employing general purpose GPU parallelization. GPU parallelization will be achieved using JCuda or jocl frameworks, depending on the available hardware. The platform will evaluate the available hardware and the number of running processes in order to enable maximum support for a user's demands.

Dynamic module loading will be a useful feature of the platform, which is significant from the perspective of server resources' optimization. If, for example, a user wants to classify ECGs using C4.5 decision trees, without any additional algorithms, then there will be no need to load all the other platform modules into memory. The modules will be implemented so that they will have maximum cohesion of services within the module, while retaining minimum coupling between other similar modules [12]. This will be enabled through an early definition of analysis scenario (see section III. *B*). The exact sequence of module execution to achieve the required goal could be determined. Auxiliary packages would contain only signatures (name, parameters) of the available components for a specific analysis phase.

The platform will contain a large number of biomedical time-series features, both general and domain-specific, i.e. related to a specific type of time-series (e.g. EEG). In the platform, we plan to use the features from HRVFrame [5], EEGFrame [13] and Comp-Engine [6] frameworks, with appropriate modifications, language translations and testing. Additionally, other domain-specific frameworks, such as the one for ECG analysis will have to be implemented mostly from scratch.

V. CONCLUSION

We have shown the requirements and architecture for an integrative biomedical time-series analysis web platform. Future work will involve implementing the requirements and reporting on the achieved modeling results of the subjects' states.

REFERENCES

[1] J. Oster, J. Behar, R. Colloca, Q. Li, Q. Li, and G. D. Clifford, “Open source Java-based ECG analysis software and Android app for atrial fibrillation screening,” in: Computing in Cardiology Conference (CinC) 2013, IEEE, pp. 731–734, 2013.

[2] A. Szczepanski and K. Saeed, “A Mobile Device System for Early Warning of ECG Anomalies,” Sensors, vol. 14, pp. 11031–11044, 2014.

[3] G. Paliwal, A. W. Kiwelekar, “A Product Line Architecture for Mobile Patient Monitoring System,” in: Mobile Health, Vol. 5 of Springer Series in Bio-/Neuroinformatics, pp. 489–511, 2015, doi: 10.1007/978-3-319-12817-7_22.

[4] A. Bravi, A. Longtin, and A. J. E. Seely, “Review and classification of variability analysis techniques with clinical applications,” BioMed. Eng. OnLine, vol. 10, p. 90, Oct. 2011.

[5] A. Jovic, N. Bogunovic, and M. Cupic, “Extension and Detailed Overview of the HRVFrame Framework for Heart Rate Variability Analysis,” in: Proceedings of the Eurocon 2013 Conference, I. Kuzle, T. Capuder, and H. Pandzic, Eds. Zagreb: IEEE Press, pp. 1757–1763, 2013.

[6] B. D. Fulcher, M. A. Little, and N. S. Jones, “Highly comparative time-series analysis: the empirical structure of time series and their methods,” J. Roy. Soc. Interface, vol. 10, p. 20130048, April 2013.

[7] A. Jovic and N. Bogunovic, “Evaluating and Comparing Performance of Feature Combinations of Heart Rate Variability Measures for Cardiac Rhythm Classification,” Biomed. Signal Process. Control, vol. 7 no. 3, pp. 245–255, May 2012.

[8] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, Book in preparation for MIT Press, 2016, url: http://goodfeli.github.io/dlbook/.

[9] Object Management Group, “Unified Modeling Language® (UML®) Resource Page”, http://www.uml.org/, last accessed on: 2016-02-21.

[10] Change Vision Inc., “Astah, Community Edition,” http://astah.net/editions/community, last accessed on: 2016-02-21.

[11] A. Jovic, K. Brkic, N. Bogunovic, “A review of feature selection methods with applications,” in: Proceedings of the MIPRO 2015 Conference, P. Biljanovic, Ed., Rijeka: MIPRO Croatian Society, 2015, Rijeka: MIPRO Croatian Society, pp. 1447–1452, 2015.

[12] K. Praditwong, M. Harman, X. Yao, “Software Module Clustering as a Multi-Objective Search Problem,” IEEE Trans. Software Eng., vol. 37, no. 2, pp. 264–282, April 2011.

[13] A. Jovic, L. Suc, and N. Bogunovic, “Feature extraction from electroencephalographic records using EEGFrame framework,” in: Proceedings of the MIPRO 2013 Conference, P. Biljanović, Ed., Rijeka: MIPRO Croatian Society, pp. 1237–1242, 2013.