INRIA INNOVATION LAB

CERTIVIBE

V1.0

SOFTWARE REQUIREMENT SPECIFICATION

Document Approval			
	Name	Function	Date
Originated by			
Reviewed by			
Reviewed by			

TABLE OF CONTENTS

1. Purpose

The purpose of this document is to list CertiViBE software requirements.

2. Scope

Requirements will be divided into two families:

- Functional requirements describing features or services delivered by the software;
- Non-functional requirements describing operational constraints and behavior (e.g. security, performance, availability, maintainability, certifiability…).

3. References

DOCUMENT #	TITLE
EN ISO 13485	Quality systems – Medical devices – System requirements for regulatory purposes
EN ISO 14971	Medical devices – Application of risk management to medical devices
IEC 62304	Medical device software – Software lifecycle processes
MEDDEV 2.1/6 January 2012	Qualification and Classification of standalone software

4. Definitions

Specifications	A specification is defined as “a document that states requirements.” There are many different kinds of written specifications, e.g., system requirements specification, software requirements specification, software design specification, software test specification, software integration specification, etc.
Requirements	A requirement is a need, expectation, or obligation. It can be stated or implied by an organization, its customers, or other interested parties. A specified requirement is one that has been stated (in a document for example), whereas an implied requirement is a need, expectation, or obligation that is common practice or customary.. There can be many different kinds of requirements (e.g., design, functional, implementation, interface, performance, or physical requirements).

Verification	Verification is a process. It uses objective evidence to confirm that specified requirements have been met. Whenever specified requirements have been met, a verified status is achieved. Objective evidence can be collected by performing observations, measurements, tests, or using other suitable methods.
Validation	Validation is a process. It uses objective evidence to confirm that the requirements which define an intended use or application have been met. Whenever all requirements have been met, a validated status is established. Validation can be carried out under realistic use conditions or within a simulated use environment.
Intended purpose	'Intended purpose' means the use for which the device is intended according to the data supplied by the manufacturer on the labelling, in the instructions and/or in promotional materials.

5. Abbreviations

DHF	Design History File	Dossier containing all documents and items relating to software design and development process: this is the chronological history of the development.
DD	Design Dossier	Dossier containing the documents and items related to the software itself
DCR	Design Change Request	
IL	Innovation Lab	Inria innovation lab CertiViBE
CC	Critical Characteristic	
SD	Software Definition	
SRS	Software Requirements Specification	
RSK	Risk evaluation	
RQ	Regulatory Quality	

6. Technical glossary

BCI	Brain computer Interface
GUI	Graphical User Interface
API	Application Programming Interface
EEG	Electroencephalography

7. General Behavior

The CertiViBE software is based upon the OpenViBE software, and could be seen as the core feature OpenViBE.

The goal of CertiViBE software is to create or load an existing processing pipeline for BCI purpose, and especially for the Neurofeedback applications. The software can load, process and save EEG data.

A processing pipeline could be seen as a scenario with several plugin (boxes) connected, to process EEG data provided to the scenario. The scenario is the way to manage the EEG signal, and for example extract relevant signal properties you want to use or monitor.

The plugin also call boxes are a collection of features that can perform mathematical treatment, classification, signal processing, signal acquisition etc. One of the main idea is that final user could develop new plugin to add features instead of rewriting all CertiViBE.

The CertiViBE software won't provide a GUI, a tool to create scenario or an acquisition device, but it will provide tools, API and documentation for building EEG signal processing systems.

For the validation and verification purpose, CertiViBE will also have testing tool to launch existing scenario.

8. Functional requirement

This section describes functional requirements involved in CertiViBE software.

8.1 General Requirements

Tag # / CC	SRS-001
Requirement	The system must provide functionalities to manage a data processing pipelines. A data processing pipeline is a collection of interconnected processing algorithms/filters. The system must be able to: - Declare and create a pipelines - Add/Remove filters to/from a pipeline - Connect/Disconnect filters - Delete a pipeline
Acceptance Criteria	Create a processing pipeline, add 3 compatible filters, connect the 3 filters, remove the last one and delete the pipeline.
Rationale	Flexibility to be able to build a personalized processing pipeline from individual filters.
Reference (SD,SRS,RSK)	SD-04
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-002
Requirement	The system must provide functionalities to configure a processing pipeline. Each filter in the processing pipeline must provide a configurability state. If the filter is configurable, then it must be possible to update its configuration. Assumption / Prerequisites: Requirement SRS-001 is full filled.
Acceptance Criteria	Create a processing pipeline, add one configurable filters, and modify its settings.
Rationale	Flexibility through late-binding.
Reference **(SD,SRS,RSK)**	SRS-001, SD-04
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-003
Requirement	The system must provide functionalities to execute one or more processing pipelines in parallel. It includes: - Starting/Stopping the execution - Pausing the pipeline execution - Executing the pipeline as quickly as possible - Executing the pipeline step by step pausing after each step Execution pipeline can be controlled directly or on stimulation receiving. Assumption / Prerequisites : Requirement SRS-001 is fulfilled.
Acceptance Criteria	Create at least two processing pipelines, start a pipeline execution, pause the pipeline execution, execute the pipeline step-by-step, restart the execution as quickly as possible, stop the execution. Do the same for other pipeline(s). All processing pipelines should be able to be executed independently in parallel.
Rationale	Flexibility through different level of execution.
Reference **(SD,SRS,RSK)**	SRS-002, SD-04, SD-41
Priority	-
Criticality	-
Non-functional RQ	1. Configurability (For control on stimulation receiving): Input stimulation type to trigger control, type of control.

Tag # / CC	SRS-004
Requirement	The system must provide functionalities for late-binding modifications. A user should be able to modify or create configuration tokens without recompiling the system. Late-binding modification must be provided through an editable resource file.
Acceptance Criteria	User can create a pipeline and write a configuration file for it which will be read when the pipeline is executed.
Rationale	Late-binding configurability brings flexibility by relieving the user from recompiling the system each time a configuration setting is modified.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-005
Requirement	The system must provide functionalities to log traces of APIs calls. Log management must provide at least 2 level of activations: - One for debugging activities - One for production activities Log must be provided on 2 different types of output: - Console output - File output Creating a new type of output must be possible and must be an add-only process that does not require to modify any existing features. The source of each log entry must be clearly identifiable and activation level should be modifiable at runtime.
Acceptance Criteria	Create a processing pipeline, execute the pipeline with different activation level and check some logs are written into the file accordingly. Check the design allows the addition of a new type of output without any modification of the current implementation.
Rationale	Traces are necessary for debugging and software usage analysis.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-
Non-functional RQ	-

8.2 Acquisition Requirements

Tag # / CC	SRS-006
Requirement	The system shall provide a way to connect to an acquisition device. This method should be extensible. Kernel must be able to synchronize its clock with the acquisition system.
Acceptance Criteria	The user will create a pipeline which acquires data and prints it. Kernel is instructed to connect to an acquisition device and the pipeline is executed. The data from the device must be produced as it arrived by the pipeline.
Rationale	Connection to an acquisition device is necessary to retrieve input data for further processing.
Reference (SD,SRS,RSK)	SD-52
Priority	-
Criticality	-
Non-functional RQ	1. Extensibility, Interoperability: The system should consider all incoming samples. If the system is connected to several devices, then their clocks must be synchronized.

Tag # / CC	SRS-007
Requirement	The acquisition system should be able to connect to the OpenViBE Acquisition Server. Assumption / Prerequisites: Requirement SRS-006 is fulfilled.
Acceptance Criteria	Test in SRS-006 shall be conducted using the OpenViBE Acquisition Server and a simulated device driver (Oscillator).
Rationale	The acquisition server is considered as another driver from the acquisition device perspective.
Reference **(SD,SRS,RSK)**	SD-09, SRS-006, SD-52
Priority	-
Criticality	-
Non-functional RQ	1. Robustness: The flow coming from the Acquisition Server must be considered "perfect" in timings.

8.3 File I/O Requirements

Tag # / CC	SRS-008
Requirement	The system must provide functionalities to store (and then to retrieve) data output by filters at any time in the processing pipeline. Data will be stored in a binary file and can contain data from multiple outputs at the same time.
Acceptance Criteria	A known signal is read by the “File Reader” component from a file. This file is then written by the “File Writer” component. The written signal must be equal to the original.
Rationale	Storage of data is important for archiving and offline analysis.
Reference (SD,SRS,RSK)	SRS-521, SRS-523, SD-16, SD-17
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-038
Requirement	The system must provide functionalities to store (and then to retrieve) matrices and stimulations at any time in the processing pipeline. Data will be stored in a csv format.
Acceptance Criteria	A known signal is read by the “CSV File Reader” component from a file. This file is then written by the “CSV File Writer” component. The written signal must be equal to the original.
Rationale	Storage of data outputs by filters in a textual format is important for debugging and testing purposes.
Reference (SD,SRS,RSK)	SRS-521, SRS-523, SD-53, SD-54
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-009
Requirement	The system must provide functionalities to store (and then to retrieve) the whole processing pipeline (structure and configuration). The pipeline is stored in xml format following a pre-defined structure (DTD or XML Schema).
Acceptance Criteria	Create a processing pipeline and save it in XML format. Then load it and save it again with another name. Check both saved pipelines are identical.
Rationale	Saving processing pipelines is needed for reusability of task-oriented pipelines and testability.
Reference (SD,SRS,RSK)	SD-04, SRS-520, SRS-522
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-010
Requirement	The system must provide functionalities to store (and then to retrieve) filter and any signal processing module settings. The settings are stored in xml format following a pre-defined structure (DTD or XML Schema).
Acceptance Criteria	A processing pipeline is created containing a filter with default settings. A file containing new settings for this filter is created alongside. Upon execution of the pipeline, the filter should use the settings from the file.
Rationale	Saving filter settings is needed for reusability of task-oriented pipelines and testability.
Reference (SD,SRS,RSK)	SD-04
Priority	-
Criticality	-
Non-functional RQ	-

8.4 Data Generation Requirements

Tag # / CC	SRS-011
Requirement	The system must provide functionalities to generate a linear signal (y (amplitude) = x (time)) on a single channel.
Acceptance Criteria	The source of signal is placed in a scenario with a spy connected to it. When the scenario is executed, the spy should see the incoming signal and validate that its level is correct.
Rationale	This type of data is used for testing purpose when no data is available.
Reference (SD,SRS,RSK)	SD-04, SD-19
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Sampling frequency.

Tag # / CC	SRS-012
Requirement	The system must provide functionalities to generate periodic stimulations.
Acceptance Criteria	The source of stimulations is placed in a scenario with a spy connected to it. When the scenario is executed, the spy should see the incoming stimulations.
Rationale	Some experiments/treatments require periodic impulses to drive stimulation, e.g.: auditory beeps.
Reference (SD,SRS,RSK)	SD-04, SD-18
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Output stimulation type and sampling frequency.

Tag # / CC	SRS-013
Requirement	The system must provide functionalities to generate a stimulation after a period of time without receiving any signal.
Acceptance Criteria	A file reader is placed in a scenario along with the algorithm detecting periods without signal. When the scenario is executed, a stimulation should be triggered after all of the signal has been read plus the “timeout” amount.
Rationale	Sometimes it is important to know that no more data is incoming. This can indicate, for example, an end of a file. Since the streams inputs are not necessarily continuous, it is important to be able to set the time after which the signal stream is considered finished.
Reference (SD,SRS,RSK)	SD-04, SD-44
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Timeout delay and output stimulation type

Tag # / CC	SRS-014
Requirement	The system must provide functionalities to generate a stimulation when the input signal sign changes.
Acceptance Criteria	Scenario is set to read artificial signal that crosses the 0 level at a known time. The system should trigger a stimulation at this moment.
Rationale	Many systems are based on detecting that the signal has reached a certain threshold. This function combined with linear algebra can detect any signal level.
Reference (SD,SRS,RSK)	SD-04, SD-45
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Output stimulation type on switch to positive value, output stimulation type on switch to negative value, index of the channel to operate on.

8.5 Data Aggregation Requirements

Tag # / CC	SRS-015
Requirement	The system must provide functionalities to merge several signals into one, provided they are sampled in the same manner.
Acceptance Criteria	Artificial signal is read by the scenario; the signal is separated into channels. A mathematical formula (the identity) is applied to each channel's signal. After that the signal is merged. Each channel of the resulting signal must be equivalent to the individual signal of each channels before the split.
Rationale	Signal can have several sources, sometimes this is due to the fact that part of it is processed in a different way from the rest – the incoming signal is split. We need to be able to merge the signal back to obtain a single flow.
Reference (SD,SRS,RSK)	SD-04, SD-46, SD-47, SRS-022
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: This algorithm is only required to merge signals with the same metrics.

Tag # / CC	SRS-016
Requirement	The system must provide functionalities to merge several stimulation streams into one stream. The resulting stream must contain all input stimulation ordered by start date.
Acceptance Criteria	Two known streams of stimulations are merged. The resulting stream of stimulations must contain the correct stimulations.
Rationale	Stimulations can come from several sources. However sometimes they have to be treated in a single stream.
Reference (SD,SRS,RSK)	SD-04, SD-42
Priority	-
Criticality	-
Non-functional RQ	1. Robustness: The resulting stream of stimulations must be ordered correctly and no stimulations can be lost.

8.6 Classification Requirements

Tag # / CC	SRS-017
Requirement	The system must provide functionalities to classify data given features. The system shall be extensible and provide a way to choose the classification algorithm at runtime. Both phases of classification must be available: 1. Training phase to build training classes 2. Testing/Prediction phase used to classify signal features The following multiclass strategies must be available: 1. One-vs.-all 2. One-vs.-one 3. Native multiclass Adding a new classifier must be possible and must be an add-only process that does not require to modify any existing features.
Acceptance Criteria	A known signal presenting two distinguishable features during known periods of time is generated and used in a classifier training pipeline. This pipeline must generate a configuration for a classifier implementation. The same signal is then used in a second pipeline and its features are extracted to feed the classifier configured in the previous step. The classifier must label the signal with correct classes. Check the design allows the addition of a new classifiers without any modification of the current implementation.
Rationale	Successful classification is one of the primary uses of the CertiViBE toolkit. A classification problem has frequently more than two classes. It is thus important to have a solution for multi-class classifiers.
Reference (SD,SRS,RSK)	SD-04, SD-13, SD-14, SD-15, SD-51
Priority	-
Criticality	-
Non-functional RQ	1. Performance: The overhead of the classifier structure should be negligible in comparison with the classifier algorithm processing.

Tag # / CC	SRS-018
Requirement	The system must provide functionalities to classify data using the Regularized LDA (Linear Discriminant Analysis) algorithm.
Acceptance Criteria	The tests of SRS-17 will be performed using an LDA classifier implementation.
Rationale	LDA classifier is a very good starting point for any classification problem.
Reference (SD,SRS,RSK)	SD-04
Priority	-
Criticality	-
Non-functional RQ	1. Performance: The implemented LDA classifier must have state of the art performance. Ref: http://statweb.stanford.edu/~tibs/ElemStatLearn/

8.7 Signal Processing Requirements

Tag # / CC	SRS-019
Requirement	The system must provide functionalities to average data represented as time series (signal, spectra, and feature vectors for classification). The averaging shall be possible on channel wise and sample wise.
Acceptance Criteria	A known signal of various types is produced (signal, spectra) and averaged. The resulting data must be correct.
Rationale	There are many applications for averaging. One of them is to increase robustness of a system by using multiple data points over time.
Reference (SD,SRS,RSK)	SD-04, SD-22, SD-31, SD-39
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-020
Requirement	The system must provide functionalities to reference the signal according to several methods: • Common average reference • Single channel reference
Acceptance Criteria	References shall be tested by applying them to known signal and comparing the results to pre-calculated values.
Rationale	Non-referenced signal contains an important amount of noise. Referencing the signal to a source helps removing it.
Reference (SD,SRS,RSK)	SD-04, SD-34, SD-36
Priority	-
Criticality	-
Non-functional RQ	1. Performance: Performance should be linearly dependent on the number of channels and sampling rate.

Tag # / CC	SRS-021
Requirement	The system must provide functionalities to rename channels of a multichannel time series.
Acceptance Criteria	A stream of channels with known names is read by the renaming algorithm and names are renamed to a different known values. The resulting signal must be identical with exception of the names.
Rationale	Avoiding channel name collision with channels coming from different signals.
Reference (SD,SRS,RSK)	SD-04, SD-33
Priority	-
Criticality	-
Non-functional RQ	1. Performance: Renaming of channels should not have any impact on performance of the system. 2. Configurability: the list containing the new names of the channels

Tag # / CC	SRS-022
Requirement	The system must provide functionalities to select a subset of channels from a multichannel time series. Removal can be performed on a name or index basis.
Acceptance Criteria	A signal with known channels is used as an input to a pipeline. Remove the first channel by index and the last one by name. The pipelines output is signal that only contains all channels but the first and last one.
Rationale	The incoming signal may contain undesirable channels, e.g.: non-connected channels, noisy channels, non-EEG channels etc.
Reference (SD,SRS,RSK)	SD-04, SD-32
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: The channels to be retained or removed from the stream

Tag # / CC	SRS-023
Requirement	The system must provide functionalities to threshold the signal.
Acceptance Criteria	A signal with known amplitude is thresholded with a maximum/minimum threshold. The amplitude of the resulting signal must not be higher/lower than the maximum/minimum demanded value.
Rationale	Signal can present artifacts that can be removed using cropping.
Reference (SD,SRS,RSK)	SD-04, SD-30, SD-024
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Thresholding method (minimum thresholding, maximum thresholding, both thresholding), thresholding values.

Tag # / CC	SRS-024
Requirement	The system must provide functionalities to apply mathematical formulae to each sample of a signal or combination of signals. The system can either perform a mathematical operation on two signals, or a conditional selection between two signals. The required functionalities are: - Mathematical: +, -, *, /, abs, acos, asin, atan, ceil, cos, exp, floor, log, log10, sin, sqrt, tan, pow - Comparison: >, >=, <, =, != - Boolean: and, or, not, xor
Acceptance Criteria	All of the available functions must produce correct values. A known signal will be sent to the mathematical processor and the results will be recorded for each of the available operators. The results will be compared to a previously calculated and validated values.
Rationale	
Reference (SD,SRS,RSK)	SD-04, SD-23
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Formula to apply. 2. Performance: The computing time should scale linearly.

Tag # / CC	SRS-025
Requirement	The system must provide functionalities to re sample (down sample or up sample) the signal.
Acceptance Criteria	Known source signal is re sampled by several different factors and compared to the same signal re sampled by a different, known and tested method.
Rationale	Some tasks are very time consuming if they are done on a signal with high or low sampling rate. For this purpose it is necessary to be able to re sample the signal.
Reference (SD,SRS,RSK)	SD-04, SD-40, SD-21
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: New sampling rate. 2. Robustness: Re sampling should not introduce artifacts to the signal, beyond what is accepted in the state of the art (in terms of delay and phase shifting).

Tag # / CC	SRS-026
Requirement	The system must provide functionalities to temporally filter the signal. The system should provide low-pass, high-pass, band-pass and band-stop filters. The method of filtering should be configurable.
Acceptance Criteria	A known signal will be temporally filtered using the four filter methods and the results compared to the same signal filtered with a known and tested method. The results must be identical with leeway for rounding errors.
Rationale	Temporal filtering is the basis for a great number of signal processing processes, feature extraction and de-noising to name a few.
Reference (SD,SRS,RSK)	SD-04
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Boundary frequencies, filtering type. 2. Robustness: Temporal filtering should not introduce artifacts to the signal, beyond what is accepted in the state of the art (in terms of delay and phase shifting).

Tag # / CC	SRS-027
Requirement	The system must provide the Butterworth method for temporal filtering.
Acceptance Criteria	SRS-026 shall be tested with a Butterworth filtering method.
Rationale	Butterworth algorithm is known and tested and accepted as state of the art.
Reference (SD,SRS,RSK)	SD-04, SRS-026, SD-27
Priority	-
Criticality	-
Non-functional RQ	See SRS-026

Tag # / CC	SRS-028
Requirement	The system must provide functionalities to spatially filter the signal. The filtered signal is composed of channels being a linear combination of input channels.
Acceptance Criteria	Known signal with constant values is sent to the spatial filter with several different settings. The output values are thus known and must be equal to the expected results.
Rationale	Several signal processing methods rely on spatial filtering. Referencing, for example is a particular case of spatial filtering.
Reference (SD,SRS,RSK)	SD-04, SD-37
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Spatial filter coefficients. 2. Performance: The complexity of the algorithm must be linear in regards to the number of channels and sampling rate.

Tag # / CC	SRS-029
Requirement	The system shall be able to generate xDAWN spatial filter coefficients from an appropriately marked signal. The algorithm is described in the scientific paper: http://www.ncbi.nlm.nih.gov/pubmed/19174332.
Acceptance Criteria	A signal with labeled P300 brain responses is sent to the filter trainer. An LDA classifier is trained on spatially filtered signal and on non-spatially filtered signal. Afterwards, a classification challenge with both of the classifiers must yield better results with the spatially filtered signal.
Rationale	xDAWN spatial filter is state of the art in pattern-recognition based BCIs such as P300.
Reference (SD,SRS,RSK)	SD-04, SD-29
Priority	-
Criticality	-
Non-functional RQ	See SRS-028

Tag # / CC	SRS-030
Requirement	The system shall be able to generate CSP (Common Spatial Pattern) spatial filter coefficients from an appropriately marked signal. The algorithm is described in the scientific paper: http://www.ncbi.nlm.nih.gov/pubmed/10400191 http://www.ncbi.nlm.nih.gov/pubmed/20889426 .
Acceptance Criteria	A signal with labeled Motor Imagery brain responses is sent to the filter trainer. An LDA classifier is trained on spatially filtered signal and on non-spatially filtered signal. Afterwards, a classification challenge with both of the classifiers must yield better results with the spatially filtered signal.
Rationale	CSP is an established technique for EEG feature extraction used as parts of nonlinear classification pipelines. It may help in situations where the class membership of the signal is expected to be reflected in the signals per-class covariance structure.
Reference (SD,SRS,RSK)	SD-04, SD-25
Priority	-
Criticality	-
Non-functional RQ	See SRS-028

Tag # / CC	SRS-031
Requirement	The system must be able to divide temporal series into blocks and then apply the various signal processing algorithms on the blocks. The blocks can be continuous, discontinuous or overlapping.
Acceptance Criteria	SRS-032, SRS-033
Rationale	Signal slicing provides many critical functionalities, e.g.: it can be used to select periods of time during which the subject was in a particular state.
Reference (SD,SRS,RSK)	SD-04
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-032
Requirement	The system must provide functionalities to select a block of signal based on a trigger.
Acceptance Criteria	A known artificial signal with known triggers (of different names) is sent to the algorithm. Selection is performed on one of the trigger names. The resulting chunks will correspond to the correct periods of the signal.
Rationale	Selecting the relevant signal block corresponding to a given stimulation.
Reference (SD,SRS,RSK)	SD-04, SD-35
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Block duration, block start time offset from the trigger occurrence time and trigger type. 2. Performance: This procedure must have no impact on the signal samples and no significant processing impact.

Tag # / CC	SRS-033
Requirement	The system must provide functionalities to select a block of signal on a regular time basis.
Acceptance Criteria	A known, artificially constructed signal is sent to the algorithm and sliced in different manners. The output slices should correspond to the correct spots in the original signal.
Rationale	Selecting the relevant signal block corresponding to a time point.
Reference (SD,SRS,RSK)	SD-04, SD-24
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Block duration, block start time offset from the trigger occurrence time and trigger type. 2. Performance: This procedure must have no impact on the signal samples and no significant processing impact.

Tag # / CC	SRS-034
Requirement	The system must provide functionalities to apply windowing methods on a block of signal.
Acceptance Criteria	Known artificial signal is first temporally epoched and then windowed. The resulting signal must be equal to signal which was processed in the same manner using different, tested and accepted tools.
Rationale	Preprocessing before spectrum analysis in order to reduce spectral leakage.
Reference (SD,SRS,RSK)	SD-04, SD-28
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Windowing method.

Tag # / CC	SRS-035
Requirement	The system must provide functionalities to compute spectral analysis on the signal using Fast Fourier Transform.
Acceptance Criteria	Artificial signal with known frequency components is created. This signal is sent to the FFT algorithm. The resulting spectrum must present the original components.
Rationale	Spectral analysis is key to many different signal processing methods. It is also a very useful monitoring tool.
Reference (SD,SRS,RSK)	SD-04, SD-26
Priority	-
Criticality	-
Non-functional RQ	1. Performance: The algorithm must have performance on par with the state of the art.

Tag # / CC	SRS-036
Requirement	The system must provide functionalities to compute the average of all the frequency band on a spectrum. The average is calculated with the following rule: chunks over time. Assumption / Prerequisites : SRS-035 is fulfilled.
Acceptance Criteria	A known artificial spectrum is averaged. The resulting value should be equal to previously calculated and validated average.
Rationale	Averaging increases robustness when working with signal containing noise.
Reference (SD,SRS,RSK)	SD-04, SRS-035, SD-35
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Frequency band.

Tag # / CC	SRS-037
Requirement	The system must provide functionalities to vote the most frequent stimulus in a given time window and send the selected stimulus as output.
Acceptance Criteria	Simulated stimulations are sent to the voter. After the configured number of stimulations, the voter should emit the correct stimulation.
Rationale	Classification according to the frequency of an event.
Reference (SD,SRS,RSK)	SD-04, SD-43
Priority	-
Criticality	-
Non-functional RQ	1. Configurability: Minimum number of stimuli to perform a vote, width of the time window, possibility to discard a stimuli after its participation to a vote, possibility to tag the output stimulus with time of vote, last occurrence of voted stimulus or time of the last participant to the vote, possibility to define a class of rejected stimulation types that can or cannot win the vote.

Tag # / CC	SRS-039
Requirement	The system should provide a way to select a subset of a spectrum matrix, turning all unselected frequency band to 0.
Acceptance Criteria	A full band spectrum is sent to the frequency band selector. After the configured frequency selected, all other bands should be set to 0.
Rationale	Preserves some spectrum coefficients and puts the others to zero depending on a list of frequencies / frequency bands to select
Reference (SD,SRS,RSK)	SD-38
Priority	-
Criticality	-
Non-functional RQ	-

9. Non-Functional Requirements

9.1 Usability/Maintainability

Tag # / CC	SRS-500
Requirement	The system must provide documentation for QA team members. Documentation must comply with class B software as defined in IEC 63604. This documentation must include: • Software Development Plan • Software Architecture • Software Detailed Specifications • User Guide • Test Plan • Tests Plan Reports • Requirements • Software Definition • Risk Management • Release Note • Boxes Documentation • Doxygen Documentation • Automated Unitary Tests • Quick User Guide • Test Program • Test Program User Guide
Acceptance Criteria	Check that all required documents as defined in the requirement are available in the delivery package.
Rationale	The software must be delivered with enough documentation to be used in a certified medical device without additional justification.
Reference (SD,SRS,RSK)	SD-01, SD-02
Priority	-
Criticality	-

Tag # / CC	SRS-501
Requirement	The system must provide documentation for scientists. This documentation must include: • Install procedure • Quick start guide with basic commands to build and run a pipeline • Public API reference guide • Detailed documentation of all scientific features • Release Note • Boxes Documentation • Doxygen Documentation • Automated Unitary Tests • Quick User Guide • Test Program • Test Program User Guide This package of documentation should be stored and available in a dedicated space define by the project team during the delivery step.
Acceptance Criteria	Check that all required documents as defined in the requirement are available in the delivery package. The delivery package will be available in a storage space define by the project (different between SRS-501 and 502).
Rationale	Usability from a scientist perspective. He should have enough documentation to get started with the system.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-

Tag # / CC	SRS-502
Requirement	The system must provide documentation for developers. This documentation must include: • Install procedure • Architecture and design documentation • Full API reference guide • Release Note • Boxes Documentation • Doxygen Documentation • Automated Unitary Tests • Quick User Guide • Test Program • Test Program User Guide This package of documentation should be stored and available in a dedicated space define by the project team during the delivery step.
Acceptance Criteria	Check that all required documents as defined in the requirement are available in the delivery package. The delivery package will be available in a storage space define by the project (different between SRS-501 and 502).
Rationale	Understandability. It should be easy for future developers to understand how the system works.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-

Tag # / CC	SRS-503
Requirement	The system must provide all services and functionalities through a set of C++ APIs.
Acceptance Criteria	Check that APIs are developed in C++ and distributed as a set of APIs.
Rationale	C++ is the base high-performance language used to develop medical device.
Reference (SD,SRS,RSK)	SD-03
Priority	-
Criticality	-

Tag # / CC	SRS-527
Requirement	The system should provide a way to help on debugging and check the validity of a streamed or derived matrix.
Acceptance Criteria	Checks if a matrix contains "not a number" or "infinity" elements
Rationale	The developer and tester need tools to validate the conformance of stream matrix.
Reference (SD,SRS,RSK)	SD-49
Priority	-
Criticality	-
Non-functional RQ	-

Tag # / CC	SRS-528
Requirement	The system should provide a way to help on debugging and log the stimulation received. The log message will be logged according to a log level set.
Acceptance Criteria	Use a clock stimulator to a stimulation listener. Now chose an appropriate log level for the stimulation listener and start a processing.
Rationale	The developer and tester need tools to log the stimulation codes received in readable way.
Reference (SD,SRS,RSK)	SD-50
Priority	-
Criticality	-
Non-functional RQ	1. Performance: Reduce log verbosity so the application could be real timed.

Tag # / CC	SRS-529
Requirement	The system should provide a way to help on debugging, by recording the structure of an EBML stream.
Acceptance Criteria	Use a reference EBML stream with a known and validated structure and compare with the output generated.
Rationale	The developer and tester need tools to have the EBML stream tree in readable way.
Reference (SD,SRS,RSK)	SD-48
Priority	-
Criticality	-
Non-functional RQ	1. Performance: Reduce log verbosity so the application could be real timed.

9.2 Changeability

Tag # / CC	SRS-504
Requirement	The software must be distributed as open-source.
Acceptance Criteria	Check that the source code is available in the delivery package.
Rationale	Some benefits of open-source in this case: auditability. Customizability is not in the list because any change in the source code will make associated documentation obsolete (e.g. tests reports, risk analysis).
Reference (SD,SRS,RSK)	SD-10
Priority	-
Criticality	-

Tag # / CC	SRS-505
Requirement	The software must be easy to extend. Addition of new processing modules should be possible without recompilation of the whole source code.
Acceptance Criteria	Check it is possible to add a new filter without recompiling the source code.
Rationale	Extensibility. New components can be easily added later.
Reference (SD,SRS,RSK)	SD-06
Priority	-
Criticality	-

9.3 Testability

Tag # / CC	SRS-506
Requirement	The system must provide functionalities to run unit tests automatically.
Acceptance Criteria	Check unit tests are scriptable i.e. it can be run from command line.
Rationale	Automatic testing.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-

Tag # / CC	SRS-507
Requirement	Building and running unit test should be fast.
Acceptance Criteria	Building and running all unit tests at code level should be less than 5mn. Integration tests that involve multiple components are not taken into account.
Rationale	Continuous Integration. Fast building unit testing allows quick feedback on commit stage. Traditionally, unit tests are run on each commit while longer integration tests are run on another time scale (e.g. nightly).
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-

Tag # / CC	SRS-508
Requirement	The system must provide functionalities to test processing pipelines automatically.
Acceptance Criteria	Check integration and regression tests are scriptable i.e. it can be run from command line.
Rationale	Automatic integration/regression testing.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-

Tag # / CC	SRS-509
Requirement	Running tests must not require extra-tools.
Acceptance Criteria	Install software system and test project on a clean platform. Check launching tests requires nothing else than the software system and corresponding dependencies.
Rationale	Portability of tests. There should be no need for platform-specific tool (other than tools shipped with the system) to run the tests.
Reference (SD,SRS,RSK)	SD-07
Priority	-
Criticality	-

9.4 Reliability

Tag # / CC	SRS-510
Requirement	The system must not find itself in erroneous state that is not detectable. If a component of the system detects an error, the cause of the error must be communicated to the caller component. Any error should be log for inspection. Error management must provide a comprehensive way to identify the error type and the criticality of the error. The way of handling error must be configurable according to the criticality and can be changed dynamically.
Acceptance Criteria	A scenario corrupted in different manners will be loaded, this error must be detected. Acceptance of error checks of all signal processing plugins.
Rationale	Error management is required to comply with medical device standards. Moreover, it is a useful tool for debugging (maintainability), testing (testability). User of the software system can rely on the system error management to build a user-friendly comprehensive error system for the end-user (usability).
Reference (SD,SRS,RSK)	SD-01, SD-02
Priority	-
Criticality	-

Tag # / CC	SRS-511
Requirement	If a running signal processing pipeline detects an error, it must stop and communicate the reason for doing so. It must not continue processing in erroneous state.
Acceptance Criteria	Open and run a scenario on corrupted data. Check running is stopped because an error occurred during processing.
Rationale	Keeping on processing in an erroneous state is not acceptable from a medical device standards perspective.
Reference (SD,SRS,RSK)	SD-01 SD-02
Priority	-
Criticality	-

Tag # / CC	SRS-512
Requirement	If a running signal processing pipeline fails to process data in real-time, it must report the problem to the user. The maximum latency of the pipeline is configurable.
Acceptance Criteria	Run a heavy-duty processing pipeline and check the system reports an error to comply with real-time processing expectations.
Rationale	The system user must be aware the processing pipeline load is too heavy. The system user will be responsible to take the right decision according to its use case.
Reference (SD,SRS,RSK)	SD-01, SD-02
Priority	-
Criticality	-

Tag # / CC	SRS-513
Requirement	The software system must stay available as long as needed by the user. This requirement must be true only if: • Enough platform resources (e.g. OS memory) are available when processing pipelines are all up and running (Minimum platform resources requirements MPRR), • MPRR do not decrease during time, • The processing executed is mathematically relevant. The minimum time of availability is: 12 Hours.
Acceptance Criteria	On a clean platform, run a set processing pipelines in parallel and run a greedy background process that let just enough resources for MPRR to be met. Check keeps on processing data for the minimum availability time.
Rationale	Longevity. It should not gradually use up a limited resource. Example longevity defects include memory leaks or filling the disk with log files.
Reference (SD,SRS,RSK)	SD-01, SD-02
Priority	-
Criticality	-

Tag # / CC	SRS-514
Requirement	If the processing load rises, the software must not lose or corrupt data chunk in the limits of physical hardware.
Acceptance Criteria	Run a heavy-duty processing pipeline that does not meet real-time requirement (see SRS-512) and check no data is lost or corrupted.
Rationale	Consistency under load.
Reference (SD,SRS,RSK)	SD-01, SD-02
Priority	-
Criticality	-

Tag # / CC	SRS-515
Requirement	Install and deployment errors must be reported to the user so that he is able to handle the problem.
Acceptance Criteria	Set an environment that makes the install procedure fails and check error is reported with detailed explanations.
Rationale	Install can fail but the user must be aware of it as he should not use the system afterwards.
Reference (SD,SRS,RSK)	SD-01, SD-02
Priority	-
Criticality	-

9.5 Performance

Tag # / CC	SRS-517
Requirement	The processing pipeline must provide low latency out of processing steps (only data streaming is taken into account). This low latency must not decrease with time. For a typical neurofeedback pipeline, end-to-end per data chunk processing must be kept constant from start to finish with no increase of memory resources except for events such as triggering of classifier training.
Acceptance Criteria	Run minimal processing pipeline during the minimum limit of time. Check benchmark results are compatible with requirements.
Rationale	Low latency and low jitter out of processing steps.
Reference (SD,SRS,RSK)	SD-04
Priority	-
Criticality	-

Tag # / CC	SRS-519
Requirement	Cold system start time must be kept under 10000ms. Starting procedure includes: • Loading the kernel • Initializing the kernel Target Environment: Surface pro 3 with intel i5 processor (no VM)
Acceptance Criteria	Execute a process that loads and initialize the kernel. Check benchmark results are compatible with requirements.
Rationale	Low latency at system start-up.
Reference (SD,SRS,RSK)	SD-08
Priority	-
Criticality	-

9.6 Interoperability

Tag # / CC	SRS-520
Requirement	Processing pipelines are compatible with OpenViBE software (version 1.1.0)
Acceptance Criteria	Open an OpenViBE scenario with the system. Open an OpenViBE scenario with the specific Identity box inside. Open a CertiViBE scenario with OpenViBE must work as long as it does not use features only available in CertiViBE.
Rationale	See SDD-11 Rationale.
Reference (SD,SRS,RSK)	SD-11, SRS-009, SD-20
Priority	-
Criticality	-

Tag # / CC	SRS-521
Requirement	Signal data recorded with the system must be compatible with OpenViBE software (version 1.1.0)
Acceptance Criteria	Open signal data recorded with OpenViBE with the system. Open signal data recorded with CertiViBE with OpenViBE.
Rationale	See SDD-12 Rationale.
Reference (SD,SRS,RSK)	SD-12, SRS-008
Priority	-
Criticality	-

Tag # / CC	SRS-522
Requirement	Processing pipelines are compatible with NeuroRT software (version 2.4.0)
Acceptance Criteria	Open a NeuroRT scenario with the system. Open a NeuroRT scenario with the specific Identity box inside. Open a CertiViBE scenario with NeuroRT must work as long as it does not use features only available in CertiViBE.
Rationale	See SDD-08 Rationale.
Reference (SD,SRS,RSK)	SD-08, SRS-009, SD-20
Priority	-
Criticality	-

Tag # / CC	SRS-523
Requirement	Signal data recorded with CertiVibe must be compatible with NeuroRT software (version 2.4.0)
Acceptance Criteria	Open signal data recorded with NeuroRT with the system. Open signal data recorded with CertiViBE with NeuroRT.
Rationale	See SDD-08 Rationale.
Reference (SD,SRS,RSK)	SD-08, SRS-008
Priority	-
Criticality	-

Tag # / CC	SRS-524
Requirement	Processing pipeline generated by another application can be imported in the system. The software system must only provide an extension mechanism that allows the user to define and use new software components developed for a specific processing pipeline format.
Acceptance Criteria	Add to the system component an importer for a new file format. Check it is possible to open a scenario that follows this format.
Rationale	Interoperability.
Reference (SD,SRS,RSK)	SD-06
Priority	-
Criticality	-

Tag # / CC	SRS-525
Requirement	The system must be able to work on systems that are not in Latin based languages, such as Japanese, Chinese and Korean. This includes loading files from locations that contain non-ASCII characters, loading files whose names contain non-ASCII characters and so on.
Acceptance Criteria	Open scenario files whose path, name and content contain non-ASCII characters.
Rationale	Internationalization. The software should not restrict client applications to specific languages.
Reference (SD,SRS,RSK)	SD-03, SD-04
Priority	-
Criticality	-

Tag # / CC	SRS-530
Requirement	Plugin’s identifier is unique and there is no conflict with an existing identifier in NeuroRT software. Define a CertiViBE range of identifier without OpenViBE identifier.
Acceptance Criteria	Extract all plugin identifier form OpenViBE and CertiViBE, and check all are unique.
Rationale	Importing a plugin from OpenViBE would be seamless as possible.
Reference (SD,SRS,RSK)	SD-09
Priority	-
Criticality	-

Tag # / CC	SRS-531
Requirement	Plugin’s identifier is unique and there is no conflict with an existing identifier in NeuroRT software. Define a CertiViBE range of identifier without NeuroRT identifier.
Acceptance Criteria	Extract all plugin identifier form NeuroRT and CertiViBE, and check all are unique.
Rationale	Importing a plugin from NeuroRT would be seamless as possible.
Reference (SD,SRS,RSK)	SD-09
Priority	-
Criticality	-

Tag # / CC	SRS-532
Requirement	The integration of a plugin (or box) from OpenViBE should be done without (or at least as possible) changing the source code of the plugin.
Acceptance Criteria	Successful build of CertiViBE software with an OpenViBE plugin out of the project perimeter.
Rationale	Importing a plugin from OpenViBE would be seamless as possible. A software could be built with CertiViBE and OpenVibe's plugin.
Reference (SD,SRS,RSK)	SD-09
Priority	-
Criticality	-

Tag # / CC	SRS-533
Requirement	The integration of a plugin (or box) from NeuroRT should be done without (or at least as possible) changing the source code of the plugin.
Acceptance Criteria	Successful build of CertiViBE software with a NeuroRT plugin out of the project perimeter.
Rationale	Importing a plugin from NeuroRT would be seamless as possible. A software could be built with CertiViBE and NeuroRT's plugin.
Reference (SD,SRS,RSK)	SD-09
Priority	-
Criticality	-

9.7 Portability

Tag # / CC	SRS-526
Requirement	The system is available under Windows and Linux platforms. Supported OS versions must be: • Windows 7, Windows 8, Windows 10 • Linux Ubuntu 14.04, Linux Ubuntu 16.04
Acceptance Criteria	Install the software and run integration tests on all supported platform.
Rationale	The software system should not dictate the platform the client application should be developed on. Windows portability is required because it covers the largest set of potential users while Linux is a popular platform for scientific development. Mac OS is not supported but portability should be easy as it is a Unix system.
Reference (SD,SRS,RSK)	SD-05
Priority	-
Criticality	-

10. Requirement Traceability

10.1 Traceability between Software Definition and Software Requirement Specification

SD	SRS
SD-01	SRS-500 SRS-510 SRS-511 SRS-512 SRS-513 SRS-514 SRS-515 SRS-516
SD-02	SRS-500 SRS-510 SRS-511 SRS-512 SRS-513 SRS-514 SRS-515 SRS-516
SD-03	SRS-525
SD-04	SRS-001 SRS-002 SRS-003 SRS-009 SRS-010 SRS-011 SRS-012 SRS-013 SRS-014 SRS-015 SRS-016 SRS-017 SRS-018 SRS-019 SRS-020 SRS-021 SRS-022 SRS-023 SRS-024 SRS-025 SRS-026 SRS-027 SRS-028 SRS-029 SRS-030 SRS-031 SRS-032 SRS-033 SRS-034 SRS-035 SRS-036 SRS-037 SRS-517 SRS-518 SRS-525
SD-05	SRS-526
SD-06	SRS-524
SD-07	SRS-004 SRS-005 SRS-501 SRS-502 SRS-503 SRS-504 SRS-506 SRS-507 SRS-508 SRS-509
SD-08	SRS-519 SRS-522
SD-09	SRS-007 SRS-523 SRS-530 SRS-531 SRS-532 SRS-533
SD-10	SRS-505
SD-11	SRS-520
SD-12	SRS-521
SD-13	SRS-017
SD-14	SRS-017
SD-15	SRS-017
SD-16	SRS-008
SD-17	SRS-008
SD-18	SRS-012
SD-19	SRS-011
SD-20	SRS-520 SRS-522
SD-21	SRS-025
SD-22	SRS-019
SD-23	SRS-024
SD-24	SRS-033
SD-25	SRS-030
SD-26	SRS-035
SD-27	SRS-027
SD-28	SRS-034
SD-29	SRS-029
SD-30	SRS-023
SD-31	SRS-019

SD	SRS
SD-32	SRS-022
SD-33	SRS-021
SD-34	SRS-020
SD-35	SRS-032 SRS-036
SD-36	SRS-020
SD-37	SRS-028
SD-38	SRS-039
SD-39	SRS-019
SD-40	SRS-025
SD-41	SRS-003
SD-42	SRS-016
SD-43	SRS-037
SD-44	SRS-013
SD-45	SRS-014
SD-46	SRS-015
SD-47	SRS-015
SD-48	SRS-529
SD-49	SRS-527
SD-50	SRS-528
SD-51	SRS-017
SD-52	SRS-006 SRS-007
SD-53	SRS-038
SD-54	SRS-038

10.2 Traceability between Software Requirement Specification and Software Definition

SRS	SD
SRS-001	SD-04
SRS-002	SD-04
SRS-003	SD-04 SD-41
SRS-004	SD-07
SRS-005	SD-07
SRS-006	SD-52
SRS-007	SD-09 SD-52
SRS-008	SD-16 SD-17
SRS-009	SD-04
SRS-010	SD-04
SRS-011	SD-04 SD-19
SRS-012	SD-04 SD-18
SRS-013	SD-04 SD-44
SRS-014	SD-04 SD-45
SRS-015	SD-04 SD-46 SD-47
SRS-016	SD-04 SD-42
SRS-017	SD-04 SD-13 SD-14 SD-15 SD-51
SRS-018	SD-04
SRS-019	SD-04 SD-31 SD-22 SD-39
SRS-020	SD-04 SD-34 SD-36
SRS-021	SD-04 SD-33
SRS-022	SD-04 SD-32
SRS-023	SD-04 SD-30
SRS-024	SD-04 SD-23
SRS-025	SD-04 SD-40 SD-21
SRS-026	SD-04
SRS-027	SD-04 SD-27
SRS-028	SD-04 SD-37
SRS-029	SD-04 SD-29
SRS-030	SD-04 SD-25
SRS-031	SD-04
SRS-032	SD-04 SD-35
SRS-033	SD-04 SD-24
SRS-034	SD-04 SD-28
SRS-035	SD-04 SD-26
SRS-036	SD-04 SD-35

SRS	SD
SRS-037	SD-04 SD-43
SRS-038	SD-53 SD-54
SRS-039	SD-38
SRS-500	SD-01 SD-02
SRS-501	SD-07
SRS-502	SD-07
SRS-503	SD-07
SRS-504	SD-07
SRS-505	SD-10
SRS-506	SD-07
SRS-507	SD-07
SRS-508	SD-07
SRS-509	SD-07
SRS-510	SD-01 SD-02
SRS-511	SD-01 SD-02
SRS-512	SD-01 SD-02
SRS-513	SD-01 SD-02
SRS-514	SD-01 SD-02
SRS-515	SD-01 SD-02
SRS-516	SD-01 SD-02
SRS-517	SD-04
SRS-518	SD-04
SRS-519	SD-08
SRS-520	SD-11 SD-20
SRS-521	SD-12
SRS-522	SD-08 SD-20
SRS-523	SD-09 SD-06 SD-03 SD-04
SRS-524	SD-06 SD-03 SD-04 SD-05
SRS-525	SD-03 SD-04 SD-05 SD-49
SRS-526	SD-05 SD-49 SD-50
SRS-527	SD-49 SD-50 SD-48
SRS-528	SD-50 SD-48
SRS-529	SD-48
SRS-530	SD-09
SRS-531	SD-09
SRS-532	SD-09
SRS-533	SD-09

11. Revision History

REVISIONS			
REVISION LEVEL	**DESCRIPTION**	**AUTHOR**	**DATE**
1	Creation of document	Jerome Chabrol, Charles Garraud, Jozef Legeny	13/11/2015