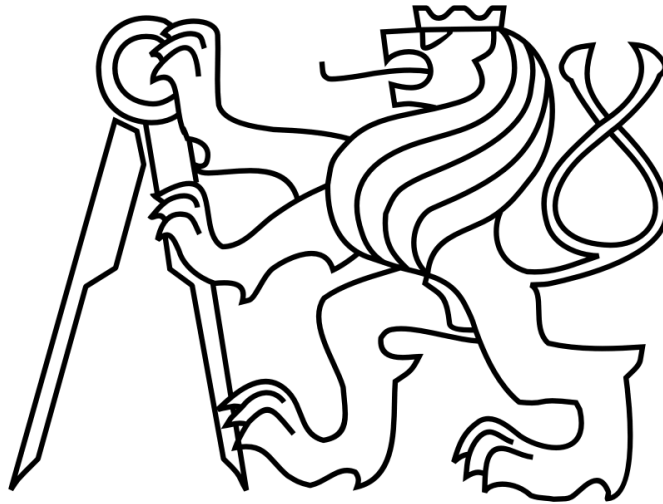


CZECH TECHNICAL UNIVERSITY in PRAGUE

Faculty of Electrical Engineering



Designing DB for Neurofeedback Lab Management

BACHELOR THESIS

Bachelor Project Supervisor: Ing. Václav Gerla

Study programme: Cybernetics and Robotics

Specialisation: Robotics

2012

Radek Procházka

BACHELOR PROJECT ASSIGNMENT

Student: Radek Procházk a

Study programme: Cybernetics a Robotics

Specialisation: Robotics

Title of Bachelor Project: Designing DB for Neurofeedback Lab Management

Guidelines:

1. Enhance your knowledge of neurofeedback and available technologies, especially SW.
2. Design a database structure for SW destined to administer a NFB lab, that works with: patient EEG recording and manual diagnostics - plannig NFB therapies with various therapy protocols - check-up diagnostic sessions - evaluation of patient's improvement within one session - evaluation of patient's improvement within entire therapy - patient list with patient details.
SW application, that will run on designed DB, will allow for: graphic outputs - timetables for therapists and patients - www booking with therapists confirmations/acknowledgements - automatic reminders (email, sms) with records of sent items and recipients - data export - interconnection with existing software for EEG monitoring used in the BioDat group - functioning with multiple diag. and therap. HW devices (able to identify, save different parameter sets) - one PC function, multiple PC function.
Even though this SW application is not part of the assignment, it is necessary to count on it's aspects during the database design process.
3. The DB design will be also available in graphical form with clearly defined and marked structure with descriptions of each table and list of reasons for choices of relations, keys, etc.
4. Implement in any scripting language for some SQL mutation of your choice.

Bibliography/Sources:

- [1] Catherine M. Ricardo: Databases Illuminated. Jones & Bartlett Learning 1 edition (April 16, 2004)
- [2] Coben Evans, James R. Evans: Neurofeedback and Neuromodulation Techniques and Applications. Academic press is an imprint of Elsevier 32 Jamestown Road, London NW1 7BY, UK First edition 2011
- [3] Mark S. Schwartz, Frank Andrasik: Biofeedback – A Practitioner's Guide 2003 The Guilford Press A Division of Guilford Publications, Inc. 72 Spring Street, New York, NY 10012

Bachelor Project Supervisor: Ing. Václav Gerla

Valid until: the end of the winter semester of academic year 2012/2013


prof. Ing. Vladimír Mařík, DrSc.
Head of Department




prof. Ing. Pavel Ripka, CSc.
Dean

Prague, January 9, 2012

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Radek Procházka
Studijní program: Kybernetika a robotika (bakalářský)
Obor: Robotika
Název tématu: Návrh DB pro řízení neurofeedback centra

Pokyny pro vypracování:

1. Prohlubte své znalosti o fungování neurofeedbacku a dostupných nástrojů, zejména SW.
2. Navrhněte strukturu databáze pro SW určený k organizaci NFB centra, hlavně:
vstupní vyšetření EEG - plánování NFB terapie s různými protokoly pro terapii - kontrolní (diagnostické) měření - hodnocení výkonu pacienta a jeho pokroku v rámci jednoho sezení – hodnocení výkonu pacienta a jeho pokroku v rámci celé terapie - evidence pacientů.
SW aplikace, pro kterou je DB určena, bude umožňovat: grafické výstupy - časové rozvrhy pro terapeuty i pacienty - objednávání přes www s potvrzováním ze strany terapeutů - automatické oznamování událostí (email, sms) s evidencí, co bylo komu odesláno - export dat - provázání s existujícím softwarem pro sledování EEG používaným ve skupině BioDat - práce s více diag. a terap. HW zařízeními (možnost rozlišení, potřeba uložení odlišných parametrů) - práce na jednom PC, práce na více PC.
Ačkoliv SW aplikace není součástí tohoto zadání, je třeba s těmito skutečnostmi během návrhu DB počítat.
3. Návrh DB bude mít i grafickou podobu s jasně definovanou a vyznačenou strukturou, popisem jednotlivých tabulek a zdůvodněním zvolených relací, klíčů atd.
4. Proveďte implementaci ve skriptovacím jazyce pro některou z variant SQL.

Seznam odborné literatury:

- [1] Catherine M. Ricardo: Databases Illuminated. Jones & Bartlett Learning 1 edition (April 16, 2004)
- [2] Coben Evans, James R. Evans: Neurofeedback and Neuromodulation Techniques and Applications. Academic press is an imprint of Elsevier 32 Jamestown Road, London NW1 7BY, UK First edition 2011
- [3] Mark S. Schwartz, Frank Andrasik: Biofeedback – A Practitioner's Guide 2003 The Guilford Press A Division of Guilford Publications, Inc. 72 Spring Street, New York, NY 10012

Vedoucí bakalářské práce: Ing. Václav Gerla

Platnost zadání: do konce zimního semestru 2012/2013

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry



prof. Ing. Pavel Ripka, CSc.
děkan

I would like to thank Ing. Václav Gerla for consulting and leading this thesis, and I would also like to thank Ing. Radim Bělobrádek for many useful tips, which helped me to finish this project successfully.

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 29.5.2012


.....
podpis autora práce

Anotace

Tato práce rozšiřuje existující software pro neurofeedback terapie o možnost použití databáze pro ukládání záznamů o pacientech a jejich výsledcích v jednotlivých terapeutických sezeních, stejně tak i celkových výsledcích za delší časové období. Dále poskytuje přehled vybavení a personálu neurofeedback centra a o plánovaných sezeních.

Annotation

This project extends the software currently used for neurofeedback therapy by incorporating a database for the filing of the patients' portfolios, their results in individual therapeutic sessions, as well as their general results monitored over a longer time period. It further offers a general overview of the equipment and staff of the neurofeedback center, and of the planned sessions.

Table of Contents

1 Introduction.....	9
1.1 Motivation.....	9
2 Theory.....	9
2.1 Biofeedback.....	9
2.2 EEG biofeedback.....	10
3 Technologies.....	12
3.1 MySQL.....	12
3.2 Swing.....	13
3.3 JFreeChart.....	13
4 Application.....	14
4.1 Signal processing.....	14
4.1.1 Fourier transformation (FT).....	14
4.1.2 Discrete Fourier Transformation (DFT).....	15
4.1.3 Short time Fourier transformation (STFT).....	15
4.1.4 FFT.....	16
4.1.5 Radix-2 algorithm.....	17
4.2 Database architecture.....	21
4.2.1 Patients and therapies.....	21
4.2.2 Sessions.....	21
4.2.3 Rounds.....	22
4.2.4 Settings.....	22
4.2.5 Personnel.....	23
4.2.6 Diagnoses.....	23
4.3 Application architecture.....	25
4.3.1 Main window.....	25
4.3.2 Personnel.....	26
4.3.3 Patients.....	26
4.3.4 SelectPatient.....	27
4.3.5 Session.....	28
4.3.6 Database.....	28
4.3.7 Credentials.....	28
4.3.8 DataHolder.....	29
4.3.9 GameData.....	29
4.3.10 Settings.....	30
4.3.11 Game.....	31
4.4 Developing a game.....	31
4.4.1 Declaration.....	32
4.4.2 Constructor.....	32
4.4.3 prepareGame.....	32
4.4.4 startRound.....	32
4.4.5 endRound.....	33
4.4.6 run.....	33
4.4.7 paint.....	33
4.4.8 Game.....	34
5 Results.....	35

5.1.1 Database.....	35
5.1.2 Application.....	35
6 Conclusion.....	36
Reference	37
User manual.....	38
1.Connection to the database.....	38
2.Add and edit the patient.....	38
3.Add and edit personnel.....	39
4.Assign a therapy to the patient.....	40
5.Plan a session	42
6.Start planned session.....	42
7.Start unscheduled session.....	43

Illustration Index

Illustration 1: EEG signal and spectrum.....	15
Illustration 2: Preparation EEG signal for segmentation.....	18
Illustration 3: EEG segmentation.....	19
Illustration 4: Overlapping windows.....	20
Illustration 5: Database diagram.....	24
Illustration 6: Main window.....	25
Illustration 7: Add/edit personnel form.....	26
Illustration 8: Add/edit patients form.....	27
Illustration 9: Logging form.....	28
Illustration 10: Example of the Game.....	34
Illustration 11: Database information.....	38
Illustration 12: Menu selection - patient list.....	38
Illustration 13: Add/Edit patient.....	39
Illustration 14: Menu selection - add/edit personnel.....	40
Illustration 15: Add/Edit personnel.....	40
Illustration 16: Menu selection - Select patient.....	41
Illustration 17: Add new therapy.....	41
Illustration 18: Menu - sessions.....	42
Illustration 19: Sessions planning.....	42
Illustration 20: Start new session.....	43

Index of Tables

Table 1: Standard frequency bands.....	11
Table 2: States of session.....	22
Table 3: Methods inherited from abstract Game.....	31

1 Introduction

1.1 Motivation

Neurofeedback is a therapy method with a large spectrum of applications. Currently, however, it is not as frequently used as it could be. The main problem is that this method isn't covered by insurance (in the Czech Republic). Another problem is the price of the equipment required for providing this therapy. And the cost of the neurofeedback software is an integral part of this overall price.

The software for executing the neurofeedback therapies is developed by the Department of Cybernetics. There is a future plan to run a neurofeedback center which would be offering therapies for free in exchange for the agreement to use the acquired data for further research. A connection to the database was one of the missing features of this software.

2 Theory

2.1 Biofeedback

Experiments with the phenomenon of people controlling their body functions with their own minds were conducted since end of the 19th century. But real biofeedback had to wait for Norbert Wiener's discovery of the cybernetics theory and feedback itself.

Biofeedback is widely used in many different ways and relates to many different problems, such as:

Electromyograph for treating chronic pain, headache, back pain and many more. Electrocardiograph therapy used to help patients with depressions, asthma, heart disease and abdominal pain. Photoplethysmograph can also help with chronic pain, headache, anxiety and stress. The Pneumograph is used to inform the patient about his/her respiratory rate and expansion and contraction of chest. It is often used with other methods such as the electrocardiograph or photoplethysmograph to train heart rate variability. Pure pneumograph biofeedback is used to treat asthma, but also anxiety disorders, panic attacks, and stress.

2.2 EEG biofeedback

Also known as neurobiofeedback or neurofeedback, it is one of the youngest biofeedback methods, but already it has wide areas of application. Starting with sleep problems, hyperactivity and attention-deficit disorders, through to learning disabilities and bipolar disorders, as well as, last but not least, potential epilepsy treatment. Epilepsy can indeed be detected just from the measured EEG signal. [1]

This method consists in measuring the EEG signal. The most common way to do that is to use electrodes on the head of the patient (there have been some experiments with subdermal needles and implanted electrodes, but they are not widely used. [2] The first measurement of human EEG was performed while using platinum wires placed under the skin as electrodes). Usually some sort of cap with electrodes sewn on is used to ensure the proper placement. The signal from the electrodes goes through the differential amplifier and the filter, which is designed to suppress disturbance from the power grid (50 or 60 Hz) and from muscle movement (usually all below 0.67 Hz).

Most application use 19 electrodes (plus one or two for measuring zero potential) in well-known 10-20 system, although low cost devices can use less channels, sometimes only one.

On the other hand there is more of a system with more electrodes. For example, in 1985 a 10-10 system with 74 electrodes was proposed. Than a 10-5 system allowing the use of 345 electrodes. There are other systems, designed to increase the electrode count, as to cover a larger part of the head. Sometimes additional electrodes are placed on the face to transmit additional information [3]. Still, spatial resolution of EEG is very low, compared to CT or MRI. However EEG excels in time resolution.

Measured signals are composed of small electrical potentials generated by neurons, and it doesn't provide any information about behaving of one single neuron. However it provides lots of information about the state and behavior of the whole brain and its parts under corresponding electrodes. Some information can be discovered from the plain EEG signal. For example sleep spindles, epileptic spikes, and the eye-blink artifact can be easily observed. Trained specialists are even able to detect potential epilepsy only by looking at an EEG record.

But the most common approach here is to use short term Fourier transformation or wavelet transformation to acquire distribution of energy in frequency bands. This distribution corresponds to the brain's state and its activity.

There are four basic and most often used frequency bands: (according to [4] other sources can use slightly different values)

Band	From [Hz]	To [Hz]
Delta	0.1	3.5
Theta	4.0	7.5
Alfa	7.5	13
Beta	14	30

Table 1: Standard frequency bands

With the development of methods able to measure higher frequencies there is the growing importance of the Gama band (30 – 100+ HZ), according to short term memory activity and cross modal sensor processing.

The ratio of energies in these bands reflects the mental state of the patient. So, if there is a therapy designed for a patient with an attention-deficit disorder, the goal is to get the patient in a state where his brain waves (or more precisely his distribution of energy across frequency bans) are similar to the brain waves of a regular person paying attention to something.

To achieve this, a simple feedback mechanism is used. The patient's brain activity is rated with a score and this score is reflected as progress in a simple game, which is then shown to the patient. As he is trying to improve his game success he improves also his brain activity.

More precisely his ratio of energy in each frequency band, compared with total energy, is closer to desired values. He is rewarded for this success with a better game score/result, and he perceives this as a positive situation. When this training is repeated, the patient becomes able to achieve better results easier. For example, when he is trained for values which correspond to attention, it becomes easier for him to pay attention to something, even out of the therapy center, in real life situations. That can lead to great improvements in his study results, as well as in his personal life.

3 Technologies

3.1 MySQL

Today, there are many data storage systems available under different sorts of a free license. The followings systems were considered :

PostgreSQL is an object-relational database system developed by a community under an MIT license. It is available for many platforms, such as GNU/Linux, MS Windows and Mac OS X. [5] The developers of this database claim that it is the most advanced open source database system.

SQLite is not a database server but only a library providing an interface to store and access data in .dbm files (also known as embed database). Although its only a small library almost all requirements of SQL-92 are implemented.

Firebird is a database engine based on Borland's InterBase 6.0 now developed by a broad community with a significant amount of former Borland employees.

MySQL was created by the Swedish company MySQL AB but now is owned by Oracle Corporation. With PostgreSQL claiming to be the “most advanced open source database” MySQL can be called the “most popular open source database”. It is often a developer's first choice of database for web applications due to its simplicity of use and its speed. (MySQL was designed to achieve better performance at the cost of missing features, like stored procedures, triggers and views. But these features were added later, as they started to be more used and more desired. Licensing here was not so simple. MySQL is provided under the GPL license but Oracle also offers a paid license.

Some **ODBMS** (object database management system) were considered too. They are now far behind laboratory experiments and proof-of-concept state. Actually, the whole concept of storing objects is a great supplement to object-oriented programming.

Unfortunately ODBMS are not quite common today and this application should be as simple as possible and mainly opened. When the word 'database' is mentioned, most people picture some sort of RDBMS, or some of its implementations. (Well, most people's imaginations picture some sort of library, but I meant people who understood computers). Resulting from these conditions, I decided

to pick the most used database - MySQL. Despite limitations, like no triggers on views and triggers limited to one action only, this database is one people are most familiar with. Which should make future development of this application easier.

3.2 Swing

Formerly, Java used Abstract Window Toolkit (AWT) to create GUI. But the AWT was platform-dependant and not so well designed. As a result, Swing framework was developed. It significantly extends the possibilities of Java GUI. For starters, SWING uses model-view-controller architecture, so it helps to keep data apart from the graphic form of every component. So, it is quite easy to change the look of components with the same behavior as the original component had. Or the other way around, to keep the component's look, and totally change the function as well as the handled data.

Event handling is improved also by, among other things, the Observer pattern. It means that anyone who implements the `ActionListener` interface can subscribe to a component and subsequently be notified every time an event occurs on this component.

AWT used system to draw components (known as “heavyweight”) which caused an inconsistent look trough different platforms. Swing however, comes with lightweight GUI, where every component is responsible for it's own rendering. That (besides other advantages like transparency support, double buffering, handling Z-index and better accessibility) allows changing the look of the entire application with a custom “Look and feel”.

3.3 JFreeChart

It is a library distributed under LGPL which provides an easy way to represent data in graphic form. More than fifty different charts are prepared for easy used. All that developers have to do is create an instance of `DataSet` and then use `ChartFactory` to create a chart. The Observer pattern is used here too, so the chart(s) are automatically updated as soon as the source `DataSet` is changed.

4 Application

4.1 Signal processing

Biofeedback therapy is based on comparing the energy within several frequency bands. In order to be able to do that, there must be a way to compute it. The most common approach to do this is to use the Fourier transformation.

4.1.1 Fourier transformation (FT)

Is defined by a complex integral:

$$S(\omega) = \int_{-\infty}^{\infty} s(t) e^{-i\omega t} dt$$

Where variable t represents time in seconds and variable ω represents frequency in hertz. The result of this calculation for every ω is the complex number S , where amplitude corresponds with the amplitude of the sine wave with a frequency of ω and a phase angle of S matches its phase angle.

If we would be able to perform this calculation for every real ω (actually every positive ω is enough, because the frequency spectrum of a real function is symmetric) we would receive a complex function $S(\omega)$ which represents amplitudes and the phase angles of all frequencies. But this exact calculation is not possible in finite time. Therefore some simplifications are necessary. First, it is not necessary to compute all frequencies. In conventional EEG biofeedback (generally all EEG measurements with scalp electrodes) frequencies over 50 Hz are insignificant because the human skull and skin damps higher frequencies.

This gives a frequency spectrum over the entire interval of time, from minus infinity to infinity. For controlling games and monitoring the patient's performance, it is necessary to be able to find changes of frequency in the spectrum within the context of time. One way to achieve that is by using wavelet transformation, or the second, used in this project is to use short time Fourier transformation.

4.1.2 Discrete Fourier Transformation (DFT)

The Conventional Fourier Transformation is computed for continuous time, but the measured data here are in discrete time and a discrete frequency spectrum is needed. So, using the discrete Fourier transformation is necessary.

The resulting frequency spectrum is symmetric around half of the sampling frequency (for the input signal without the imaginary part), except in d. c. component. The example is on Illustration 1.

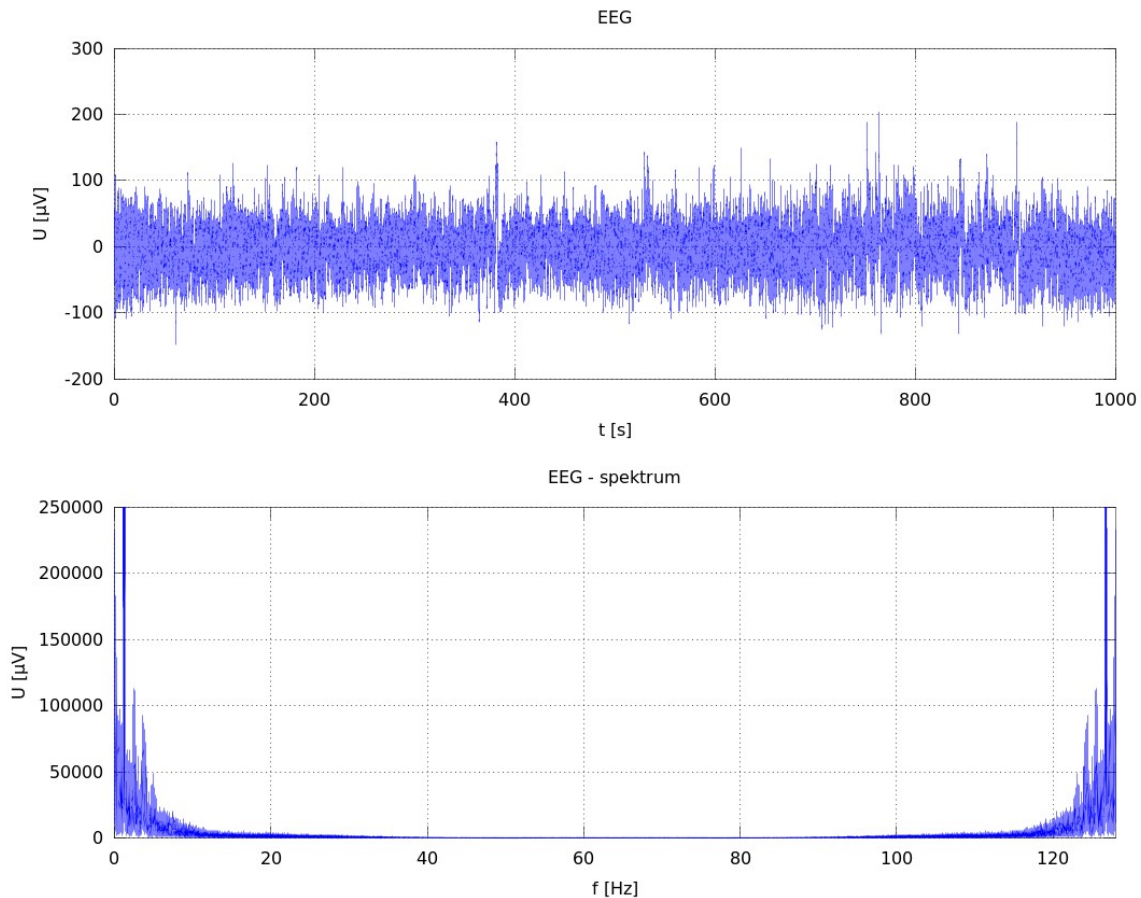


Illustration 1: EEG signal and spectrum

4.1.3 Short time Fourier transformation (STFT)

The idea behind the short time Fourier transformation (or short term Fourier transformation) is to cut source signal to segments, each of the same lengths, and then compute a Fourier transformation

for each segment separately. Usually some window function is used to do the cutting. The most common are rectangular, Hamming, Hanning, Blackman and Kaiser windows. But tapered windows should be preferred to reduce the amount of ripple [6]. Please see the example on Illustration 2 (The values of Hanning window are here set to demonstrate the principle. During real application the edges of the window would be sharper) which shows the signal before and after using the Hanning window, and Illustration 3 with separated segments and corresponding frequency spectrum (zoomed to the frequencies used in neurofeedback).

In an effort to get more accurate results “overlapping” may be used. Overlapping windows are shown on Illustration 4.

For each segment the following formula is used:

$$S(\Omega) = \sum_{k=0}^{N-1} s(k) e^{-i(2\pi k \Omega / N)}$$

4.1.4 FFT

The conventional method of computing the discrete Fourier transformation are quite expensive in terms of computer time. $O(N^2)$ to be concrete. But J. W. Cooley and J. W. Tukey introduced the algorithm to perform this computation in $O(N \cdot \log(N))$, which led to huge improvements (not only) in signal processing. Today, there are several different algorithms for FFT, which all have their different limitations.

Radix-2 algorithm, requires N to be to the power of 2. There are some other modification such as a prime-factor algorithm, requiring N to be a product of two co-primes. And variants without these limitations are also available. Although they are not so simple to implement, or even understood.

All FFT algorithms are based on dividing sequences of samples to subsequences and computing FFT from these subsequences.

4.1.5 Radix-2 algorithm

According to [6]: With N being an even number, then the sum used to compute the frequency spectrum can be divided into two smaller parts:

$$S(\Omega) = \sum_{k=0}^N s(k) e^{-i(2\pi k \Omega/N)} = \sum_{k=0,2,4,\dots}^N s(k) e^{-i(2\pi k \Omega/N)} + \sum_{k=1,3,5,\dots}^N s(k) e^{-i(2\pi k \Omega/N)}$$

after substitution $k=2m$ for the part where even indexes are used and $k=2m+1$ for the part with odd indexes:

$$S(\Omega) = \sum_{m=0}^{N/2-1} s(2m) e^{-i(2\pi 2m \Omega/N)} + \sum_{m=0}^{N/2-1} s(2m+1) e^{-i(2\pi(2m+1) \Omega/N)}$$

with another substitution $g(m)=s(2m)$ and $h(m)=s(2m+1)$ the equation can be rewritten as

$$S(\Omega) = \sum_{m=0}^{N/2-1} g(m) e^{-i(2\pi 2m \Omega/N)} + \sum_{m=0}^{N/2-1} h(m) e^{-i(2\pi(2m+1) \Omega/N)}$$

$$S(\Omega) = \sum_{m=0}^{N/2-1} g(m) e^{-i(4\pi m \Omega/N)} + e^{-i(2\pi \Omega/N)} \cdot \sum_{m=0}^{N/2-1} h(m) e^{-i(4\pi m \Omega/N)}$$

with the final substitutions $W_k = e^{-i(4\pi m \Omega/N)}$ (W_k is often referred as the twiddle factor)

$$G(\Omega) = \sum_{m=0}^{N/2-1} g(m) e^{-i(4\pi m \Omega/N)}$$

$$H(\Omega) = \sum_{m=0}^{N/2-1} h(m) e^{-i(4\pi m \Omega/N)}$$

comes the final equation

$$S(\Omega) = G(\Omega) + W_k \cdot H(\Omega)$$

Instead of one set with N samples there are now two sets each with $N/2$ samples, which lowers the time complexity from N^2 to $2 \cdot (N/2)^2 = N^2/2$. Assuming N is the power of two, this process can be recursively repeated to achieve complexity $N \cdot \log(N)$.

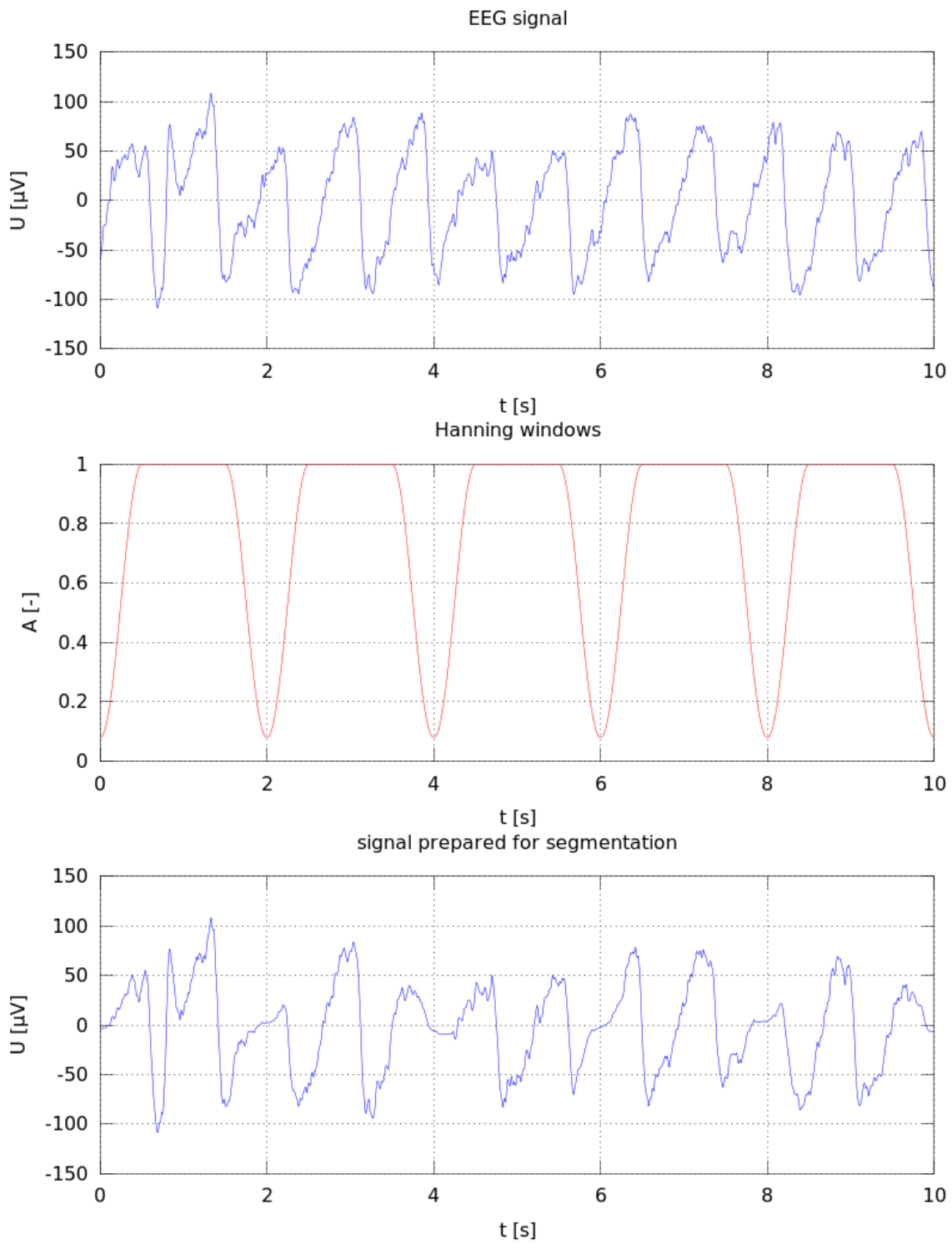


Illustration 2: Preparation EEG signal for segmentation

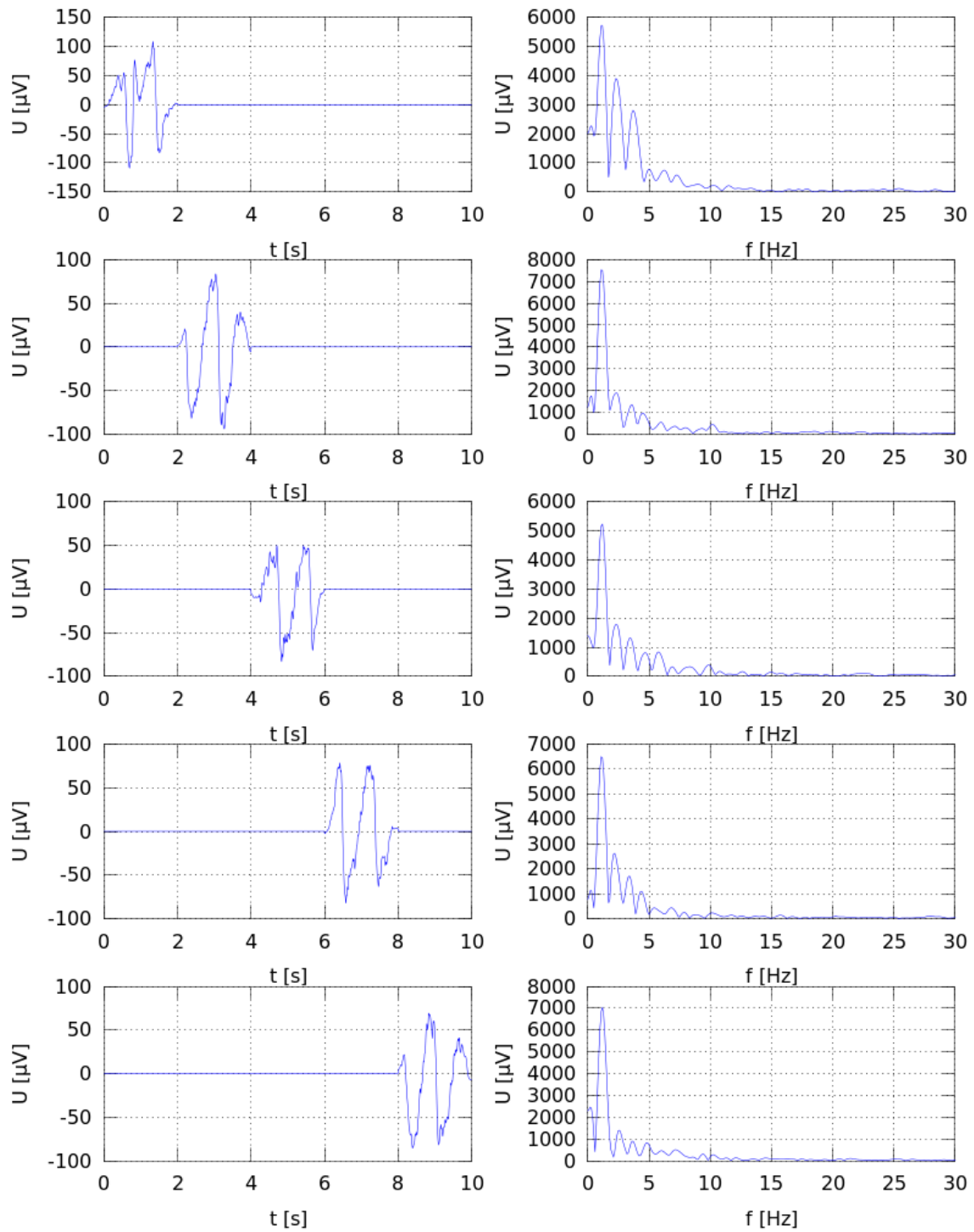


Illustration 3: EEG segmentation

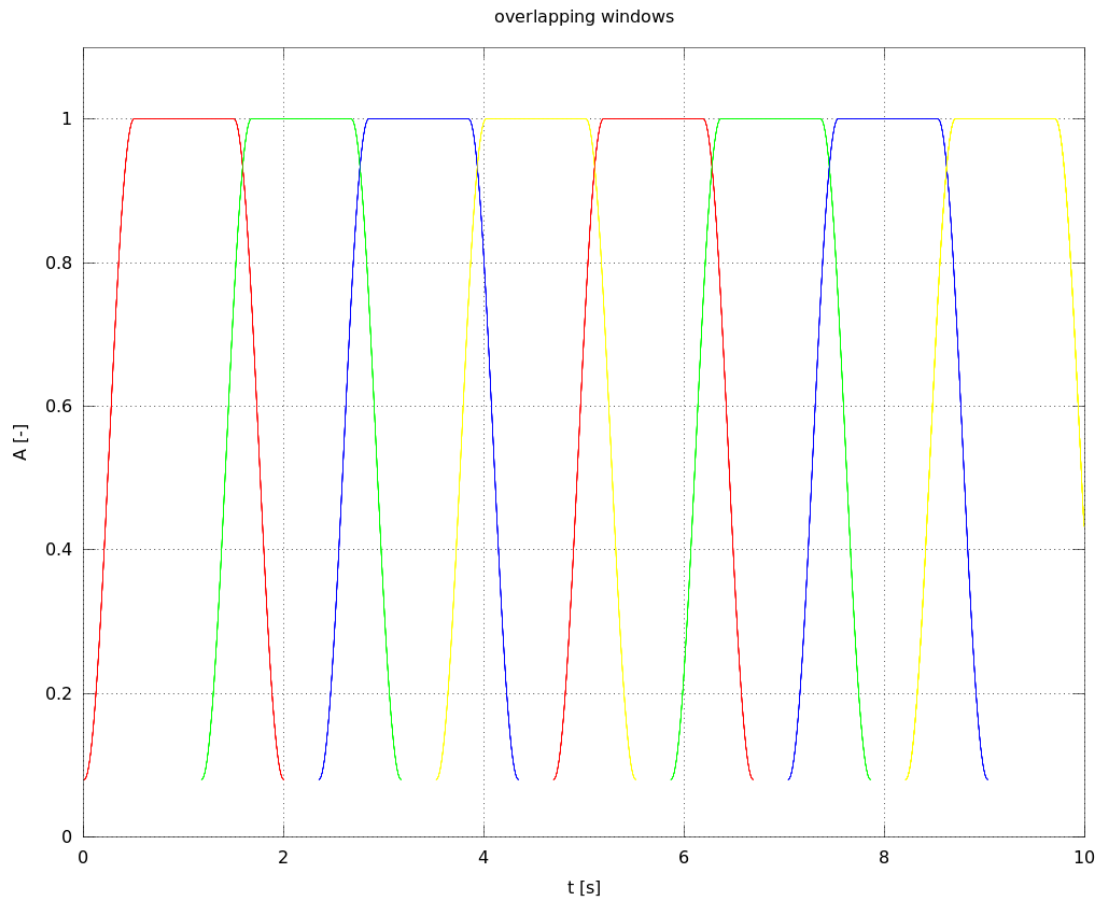


Illustration 4: Overlapping windows

4.2 Database architecture

The purpose of this work is to extend the existing application for neurofeedback within the database necessary for running the neurofeedback center. The main requirements are to store and make easy accessible data about patients, their therapies, their progress and their overall results.

There are also future plans to enrich this project by implementing an expert system, which would help therapists with diagnosis, or at least provide them with estimated probabilities of plausible diagnoses and thus help them select the proper treatment/therapy. Another planned extension is a web service for planning and booking therapy sessions.

4.2.1 *Patients and therapies*

The whole application is designed to help patients, so the `Patient` table is included. Beside storing information like names and phone numbers, it should provide access to the patient's therapy. There is the possibility of the patient having more than one therapy. It's not unusual when patient, several months after finishing one therapy, starts another, but now focused to different problem. This results in 1:N relationship with the `Therapy` table.

4.2.2 *Sessions*

While attending a therapy, the patient repeatedly visits the neurofeedback center. These visits are called 'sessions'. There is table `Session`, which was originally intended only to store records about passed sessions, but it was modified in order to simplify the future implementation of a sessions-planning web service. The modification lies in the possibility of registering the state of every session. The possible states are in Table 2.

state	description
planned	patient ordered session (over phone, mail, or in future by sessions planning web service)
scheduled	session was confirmed by attending therapist
running	this state is for session which was started by therapist (not by reaching planned time)
done	for finished sessions
voided	session canceled by patient or therapist

Table 2: States of session

There is also room for filling out where the session was, or where it is planned to be, the device used/planned to be used, and who the attending therapist was or will be.

4.2.3 Rounds

Each session usually lasts from thirty minutes up to two hours top. During that time the patients play one or more games. The number and the duration of rounds depends on the therapists decision. The patient's score is stored during every round into the dedicated `Score` table. Than, with the end of the current round, the corresponding record in the `Round` table is updated with the average score.

4.2.4 Settings

The patient's score depends on the settings of the currently played game. And these settings can be changed while the game is running. Game settings consist of the `Band Settings` which provide information about the frequency scope of each channel (which can slightly change with the desired effect of therapy [or rather literature used for therapy]) and the `Channel Settings` which store channel settings in terms of game difficulty and the channel's effect on game. This allows therapists to adapt games not only to therapy goals, but to the patient's current results as well. Because there are seven channels, table `Settings` is used to bond the corresponding records in both previously mentioned tables together, and also to store information about the method used to compute the score. Now the square sum of differences between the desired and the achieved energy in each channel is used, although it can be changed.

Table `Set` keeps information about which settings were used and when. But there is also the table `Preset` which contains prepared settings for the most common therapies.

4.2.5 Personnel

There are several roles which the personnel can fulfill. The main role is of the therapist, who performs therapies themselves. Some neurofeedback centers can also have their own doctor (or even doctors) to perform the initial entrance examination. And the last role is of the assistant who coordinates and plans therapies, as well as chooses the therapists, the rooms to be used, and the devices.

By storing them in one table, there is significant simplification in the case where some, or all, roles are carried out by one person.

4.2.6 Diagnoses

This table is prepared for future extension with an expert system to provide probabilities of diagnoses and offer a corresponding therapy. Please note that the data in the `Diagnose` table are only for purposes of demonstration.

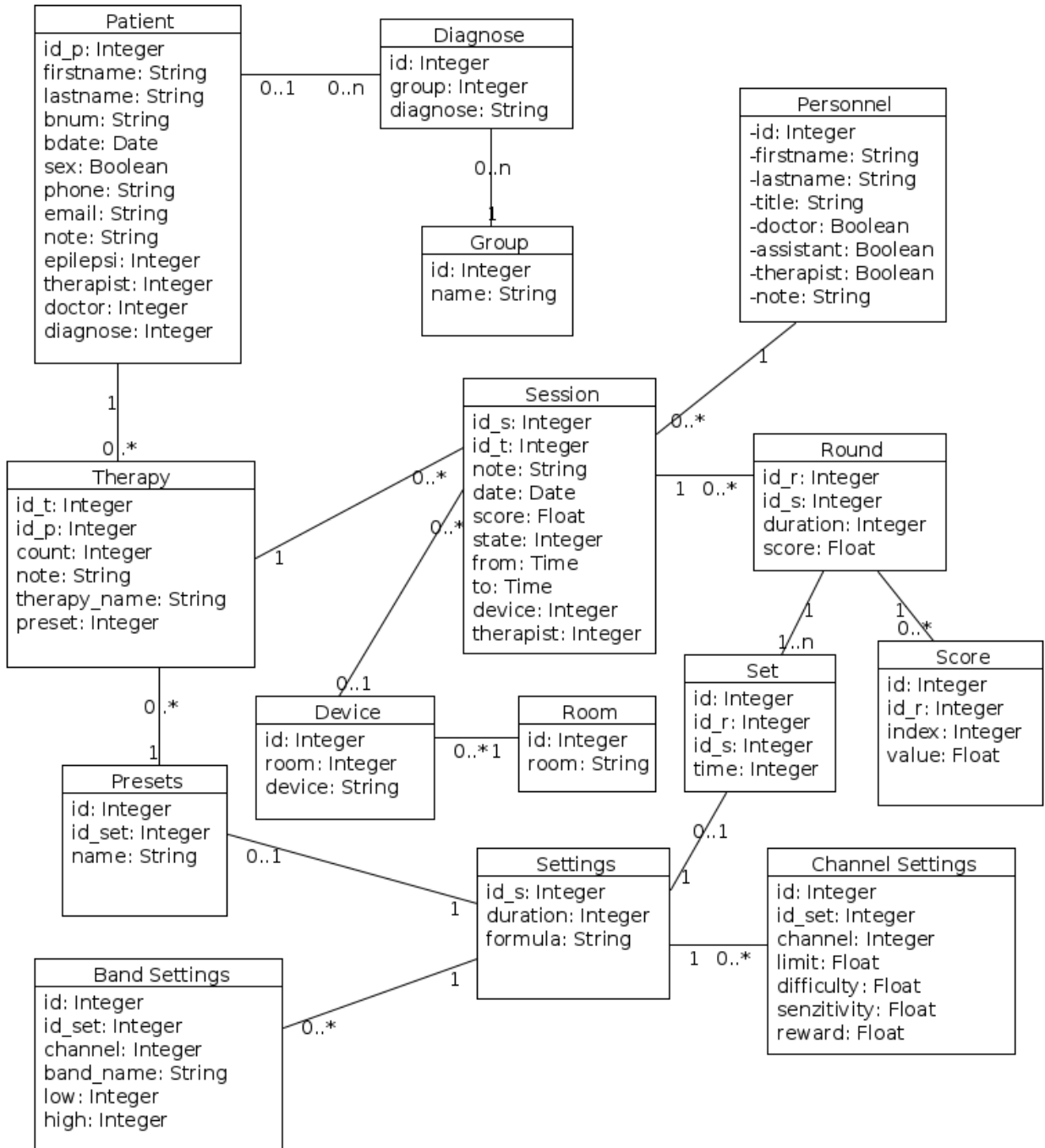


Illustration 5: Database diagram

4.3 Application architecture

4.3.1 Main window

The main window, which shows itself after the application has started, is in the class EEG. This window provides a menu to control the application, and also shows several charts. The first is the standard EEG signal (only four probes). Then, there is a long term spectrogram, which was formerly used in a previous version of this application – a version for long-term monitoring. However, it can still be helpful to have a look at the patient's state during the entire session. The Last chart here shows the patient's performance in a played game. The overall score (computed by the formula shown in chapter 4.3.9) for the actual rounds is shown here.

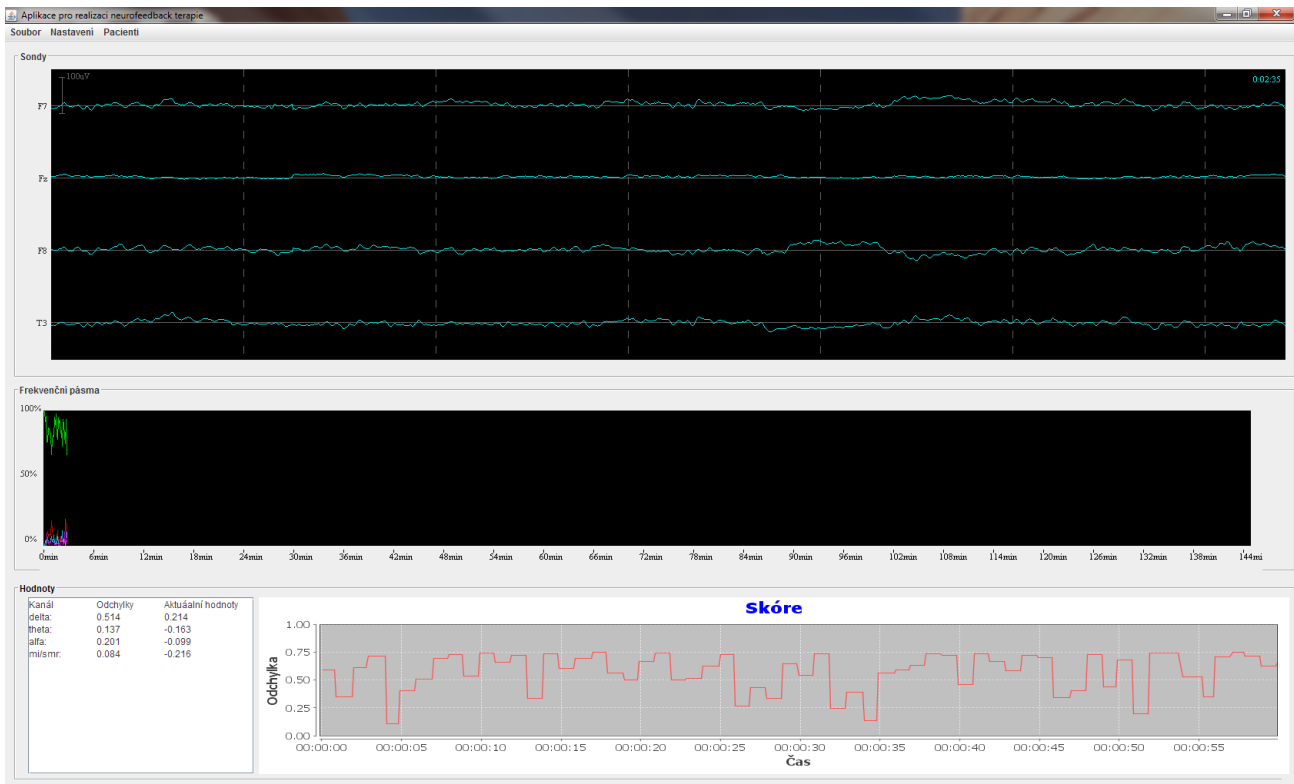


Illustration 6: Main window

4.3.2 Personnel

This class handles all personnel present in the neurofeedback center. Three main roles are mentioned and explained in chapter 4.2.5. There is a simple form to add new, or edit the current personnel.

Illustration 7: Add/edit personnel form

4.3.3 Patients

Similar to the personnel settings, there is the possibility to add new, or modify the existing, records of current patients. The `Patients` class provides the form to do this. An example of this form is shown on Illustration 8. There is the diagnosis record, which is now selected by the doctor performing the initial entrance examination, or by the assistant or therapist who must then respect the results of the initial entrance examination. There are also plans to extend this application with an expert system to determine the diagnosis (or rather probabilities of potential diagnoses) from the patient's symptoms.

Pacient	Křestní jméno	Příjmení	Datum narození	Rodné číslo	Pohlaví	Telefon	Email	Epilepsie	Terapeut	Ošetřující lékař	Diagnóza	Poznámka
Jan Novák	Jan	Novák	31.12.2001	124123414	<input checked="" type="radio"/> Muž <input type="radio"/> Žena	+420 721 985 443	testpatient@fel.cvut.cz	nezjištěno	Tomáš Černý	Tomáš Černý	ADHD	

Nový pacient Uložit

Illustration 8: Add/edit patients form

4.3.4 *SelectPatient*

This class provides the form and methods to watch the patient's results in his therapies (there is a chart showing the average score for every passed session within the selected therapy) as well as the possibility of assigning a new therapy to the patient. This new therapy should be based on one of the preset therapies, but there are currently only test-data in the database, so consultation with an experienced therapist is necessary before this application starts to be used to provide neurofeedback therapies to any real patients.

4.3.5 *Session*

For every session with the patient in the neurofeedback center, there is a corresponding record in the `session` table and every session is also reflected by one instance in the `Session` class. And that instance has a form to control the run of the session. Specifically, all of the channel settings can be changed here as well as the widths of the frequency bands. But the main purpose of this window is to prepare the game and launch the individual rounds.

4.3.6 *Database*

The database class handles all manipulation with data in the database. It also cares about credentials used for logging into a database. The database logging form is shown on Illustration 9: Logging form. This form is shown every time the database connection fails or when the user uses the “Database settings” from the “Settings” menu.

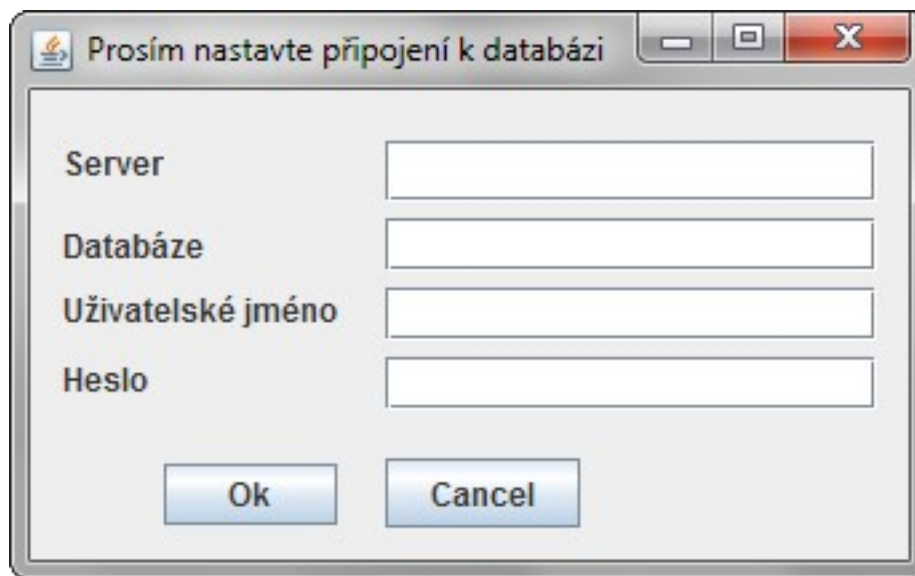
The image shows a standard Windows-style dialog box with a title bar that reads "Prosím nastavte připojení k databázi". The dialog contains four text input fields, each with a label to its left: "Server", "Databáze", "Uživatelské jméno", and "Heslo". At the bottom of the dialog, there are two buttons: "Ok" and "Cancel". The dialog has a light gray background and a thin border.

Illustration 9: Logging form

Information from this form is stored in the `Credentials` class.

4.3.7 *Credentials*

The `Credentials` class is used to store and manage database logging information. Instance of `Credentials` receive them in the constructor and can store them with the `save` function. Or the constructor without parameters can be used to gain instance without this information and then get

them with `load` function. Logging information is now stored in a java property file, and only one set of logging information can be stored. But this can easily be changed, for example if there is a need for some kind of database-connection manager who would be able to handle more than one database. Despite this possibility it is not likely to happen, because neurofeedback centers are not supposed to maintain more than one database.

This object provides functions useful for logging into a database. The first method is `getUrl` to get a url with all the needed pseudo protocols. Both the server and the selected database name are included in this url. Along with the methods `getUsername` and `getPassword` all information necessary to connect to a database is provided.

4.3.8 *DataHolder*

It is a component which stores and handles basically all data related to EEG. This component was designed for an older application for long-term EEG monitoring, which was lately modified to neurofeedback application. Besides long-term related data, the `DataHolder` provides two different ways of getting data for the neurofeedback games.

The first is a set of methods `getActualAlphaBand()`, `getActualBetaBand()`, `getActualDeltaBand()`, `getActualThetaBand()`. These methods return the last value in a long-term spectrogram. This spectrogram, however, reflects only four basic frequency bands and thus a new, more detailed, division of frequency bands is required. Also, the borders of these new bands can vary. So `DataHolder` was extended with `bandBorders` and `bands`. `BandBorders` is a list of settings for every frequency band. Method `setBandBorders()` and `getBandBorders()` are intended for manipulating these settings. And better `getBands()` is used to return the array of energies (relative to the total energy) in the frequency bands. Data for this array are computed in `SpectrumThread`.

4.3.9 *GameData*

`GameData` holds all settings and values needed for playing the neurofeedback games. This includes the settings of each channel, which consists of several items. First is the frequency band used for this channel. This is the main difference between the new and the the old version of this application, which didn't allow to change used bands or even disable some bands in the application settings. If it

was necessary to do this, it had to be changed by modifying the source code of the game for which this change was intended.

The next settings affect the results in each channel directly. There is `limit`, which the patient is trying to achieve, `sensitivity` – basically a gain of the patient's energy in the frequency band according to this channel. Then `difficulty` to scale the difference between the desired limit and the patient's actual performance (modified by `sensitivity` coefficient). The last one is `reward`, determining the threshold for additional motivation.

The setting of `sensitivity` is quite controversial. When it is not set to one for each channel, it changes calculations, so `limit` can't be used as the desired ratio of the whole energy in the appropriate channel. But it was required, so `limit` of a used channels can be set to any number instead of being restricted by the sum of limits equal to one.

For each channel, a value called “channel score” is computed. With channel score c , difficulty d , sensitivity s , limit l and the actual energy in the current channel v :

$$c = d * (v - s * l)$$

and the overall score S is computed as a sum of squares of all the channel scores:

$$S = \sum_{i=1}^4 c_i^2$$

4.3.10 Settings

There is the class `GameSet` to store `ChannelSettings` used by the `GameData` class as well as `BandSettings` used by the `DataHolder`. These classes directly match tables with corresponding names and their function is described in chapter 4.2.4. Using two separate structures for settings might seem redundant, but this is intentional. There are future plans to extend this application with heart rate monitoring and, possibly, blood pressure monitoring, so separation of the EEG-specific setting (frequency range) to one place, and the settings which are the same for every possible feedback (target value, difficulty,...) is necessary.

4.3.11 Game

Game is an abstract class, which should be extended by any neurofeedback game. There are three main methods which every game should have.

Method	Description
prepareGame	Serves to prepare the game window and all necessary resources before the game has started.
startRound	After this method commences, the game should start to respond to the patient's EEG.
endRound	This method commences when the time set by the therapist is reached. There is no need of computing the overall score in this method, because this is automatically done by <code>GameData</code> class.

Table 3: Methods inherited from abstract Game

All methods mentioned in Error: Reference source not found are called by the Session class either when the therapist commences them, or when the time dedicated for the current round expires. Method `prepareGame` should be commenced once, at the start of the session, or when the therapist decides to change a game which the patient is currently playing. Methods `startRound` and `endRound` can be commenced repeatedly, after the game has been prepared.

4.4 Developing a game

This chapter demonstrates the process of programming a simple game. This specific game is not intended to be used in real neurofeedback therapy, its purpose is only to show how to create such a game.

Every game should extend the abstract class `Game`. There are three main methods which must be implemented:

4.4.1 Declaration

```
public class GamePad1 extends Game implements Runnable {
    public boolean run = false;
    boolean play = false;
    public Thread t;
    Float[] f = {0f, 0f, 0f, 0f};
    GameData gd;
```

4.4.2 Constructor

```
public GamePad1() {
    initComponents();
    gd = GameData.getInstance();
    t = new Thread(this);
    System.out.println("GP: konstruktor");
}
```

This piece of the code create an instance of `gameData` (which is singleton, so there is only one instance shared through the whole application) and will prepare thread which will later handle the acquisition of the game data. Although this game uses standard Java threads, the `Game` class doesn't implement a runnable interface, so other games can use any other mechanism to gather data and prepare a scene to be drawn.

4.4.3 *prepareGame*

```
@Override
public void prepareGame() {
    this.setVisible(true);
}
```

This game does not use any additional graphics, sounds or data, but, if it would do so, this method is ideal for loading them, as it avoids delay by loading them when the first round starts.

4.4.4 *startRound*

```
@Override
public void startRound() {
    play = true;
}
```

This method just set the `play` variable, which is used in the thread to determine if the game is indeed running.

4.4.5 *endRound*

```
@Override
public void endRound() {
    System.out.println("END ROUND");
    play = false;
}
```

Similarly to the `startRound` method, there is only the `play` variable unset.

4.4.6 *run*

```
@Override
public void run() {
    while (run) {
        if (play) {
            f = gd.getGameData();
            validate();
            repaint();
        }
        try {
            Thread.sleep(50);
        } catch (InterruptedException ex) {
            ex.printStackTrace();
        }
    }
}
```

This thread reads data from `gameData` class and saves them for calculation performed in the `paint` method.

4.4.7 *paint*

```
@Override
public void paint(Graphics g) {
    super.paint(g);
    float val = 150;
    int[] xpoints = {100, 250, 400, 250};
    int[] ypoints = {250, 400, 250, 100};
    int[] xxpoints = xpoints.clone();
    int[] yypoints = ypoints.clone();
    xxpoints[0] = (int) (xpoints[0] - f[0] * val);
    xxpoints[2] = (int) (xpoints[2] + f[2] * val);
    yypoints[1] = (int) (ypoints[1] + f[1] * val);
    yypoints[3] = (int) (ypoints[3] - f[3] * val);

    g.setColor(Color.getHSBColor(0.29f, 0.5f, 0.95f));
    g.fillPolygon(xxpoints, yypoints, 4);
}
```

```

g.setColor(Color.green);
g.drawPolygon(xpoints, ypoints, 4);
g.setColor(Color.black);
g.drawPolygon(xpoints, ypoints, 4);
g.drawLine(20, ypoints[0], 480, ypoints[2]);
g.drawLine(xpoints[1], 20, xpoints[3], 480);
g.drawString("kanál 1: " + f[0], 20, 50);
g.drawString("kanál 2: " + f[1], 20, 70);
g.drawString("kanál 2: " + f[2], 20, 90);
g.drawString("kanál 2: " + f[3], 20, 110);
}

```

In this method all the calculations which transform the patient's effort to the visual representation are executed. Also target values are marked as a black rectangle. Then a green polygon is painted and the distance between corners of this polygon and the previously painted rectangle are equivalent to the difference between the desired values and the values measured on the patient.

4.4.8 Game

There is an example of the game running on

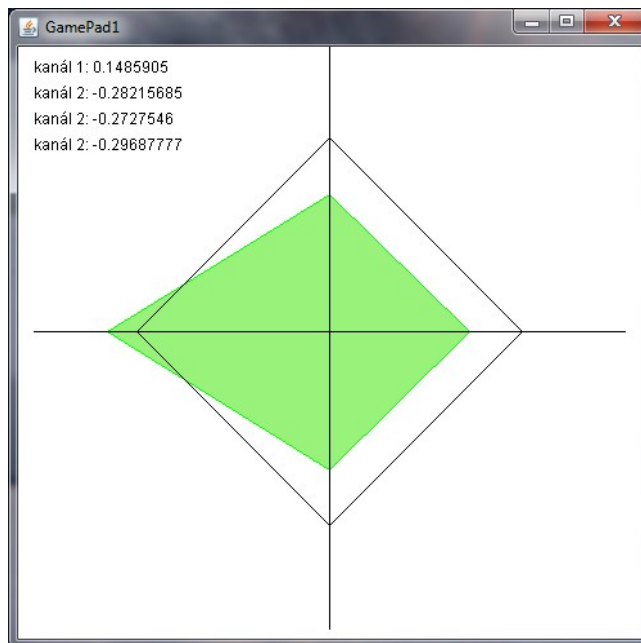


Illustration 10: Example of the Game

This game is only for demonstration. A real application would require at least the use of double buffering to prevent blinking while the content of the window is repainted. Also, the resizing of this window is not handled, so the game will still keep its dimensions regardless of the window size.

5 Results

5.1.1 Database

Main purpose of this database is to store information about patients, so there the `Patient` table. There is a foreign key to the table `Diagnoses`, to specify the patient's diagnosis. And this table has its own foreign key pointing to the `Group` table, to assign this diagnose to the proper category.

For every patient, there can be one or more therapies and every therapy has its own record in the `Therapy` table. This table stores id of corresponding patients and also through the `Preset` table default values for this therapy (or, more precisely id of record in the `Settings` table with default value for this therapy).

The `Settings` table stores only the duration of the round and the way used to compute patient's score. The rest of the settings are in tables `Channel Settings` and `Band Settings`, which both have a reference to a row in the `Settings` table.

A patient's visit during a therapy is called a session and every visit is stored in the `Session` table. There is a foreign key used to assign session to the patient's therapy. Also there are links to the tables `Personnel` and `Device`, to save information about which therapist performed this session and which device was used.

During every session, several rounds of a game are played. The table `Round` is used to keep information about this rounds. Its primary key serves as the foreign key to the `Score` table, which, as its name says, serves to store patient's score. This primary key is also used by the table `set`, which saves details about which settings were used and when.

5.1.2 Application

The main application window was extended with the panel showing a chart of patients score during every game round. It was causing unexpected exceptions due to a race condition, but it was fixed by using synchronized block as suggested in [7].

There were added dialogs to manipulating the patients and staff lists, both using same principle with

patient/personnel list on the left side and form for editing or adding new on the right side, which should be more user friendly than a sequence of window to choose a patient and than new window to edit previously selected patient.

Then there is a new window with session planning, which shows planned sessions and allows manipulating with them. There is also a window to view the patient's achievements during the entire therapy and assigning therapies to the patient.

6 Conclusion

In this thesis the database structure for the neurofeedback center was designed and implemented using MySQL. There is possibility of planning therapies based on one of the pre-defined (by psychiatrist or neurologist) protocol and planning sessions within such therapy. From the planned session, schedule for patients, therapists, rooms, or devices can be generated. The patient's results during sessions are stored and these data can be used to create a general overview on the patient's progress. The database is also prepared for being extended with another type of biofeedback due to the separation of common settings to one table, and using a second table for EEG-specific settings.

Although it was not part of the project assignment, the neurofeedback application was heavily modified to be able to work with the database that was designed and implemented in this thesis. There is a graphic interface to modify the patients list, as well as the personnel. A component to plan sessions was added, though it is showing schedule by date only. There is a chart, generated from the database, to show patient's results during a therapy.

This application was tested only with a pre-recorded EEG signal, but it was working well, except minor difficulties on Linux with desktop environment using accelerated graphics.

As of now, it is probably a first open-source neurofeedback application with the support for patients database. And it might be, in the future, used to run neurofeedback therapy under the auspices on the BioDat group on the Cybernetics Department.

Reference

[1]	SCHWARTZ, Mark S a Frank ANDRASIK. <i>Biofeedback: a practitioner's guide</i> . 3rd ed. New York: Guilford Press, c2003, 930 s. ISBN 15-723-0845-1.
[2]	USAKLI, Ali Bulent. Improvement of EEG Signal Acquisition: An Electrical Aspect for State of the Art of Front End. <i>Computational Intelligence and Neuroscience</i> . 2010, roč. 2010, s. 1-7. ISSN 1687-5265. DOI: 10.1155/2010/630649. Available at: http://www.hindawi.com/journals/cin/2010/630649/
[3]	Nobre, A. C., and G. McCarthy, "Language-Related Field Potentials in the Anterior-Medial Temporal Lobe: II. Effects of Word Type and Semantic Priming," <i>J. Neurosci.</i> , Vol. 15, 1995, pp. 1090–1098.
[4]	TONG, Shanbao a Nitish Vyomesh THAKOR. <i>Quantitative EEG analysis methods and clinical applications</i> . Boston: Artech House, c2009, 421 s. Artech House engineering in medicine. ISBN 15-969-3204-X.
[5]	"What is PostgreSQL?". <i>PostgreSQL 9.0.0 Documentation</i> . PostgreSQL Global Development Group. Retrieved 2010-09-20. Available at: http://www.postgresql.org/docs/current/static/intro-whatis.html
[6]	MANDAL, Mrinal Kr a Amir ASIF. <i>Continuous and discrete time signals and systems</i> . New York: Cambridge University Press, 2007, 865 s. ISBN 05-218-5455-5.
[7]	ECKEL, Bruce. <i>Thinking in Java</i> . 3rd ed. Upper Saddle River, N.J.: Prentice Hall, c2003, 1119 p. ISBN 01-310-0287-2.
[8]	SEGARAN, Toby a Jeff HAMMERBACHER. <i>Beautiful data: [the stories behind elegant data solutions]</i> . 1st ed. Sebastopol, CA: O'Reilly, c2009, xv, 364 p., [32] p. of plates. ISBN 978-059-6157-111.
[9]	SINHA, Rakesh Kumar. EEG power spectrum and neural network based sleep-hypnogram analysis for a model of heat stress. <i>Journal of Clinical Monitoring and Computing</i> . 2008, roč. 22, č. 4, s. 261-268. ISSN 1387-1307. DOI: 10.1007/s10877-008-9128-x. Available at: http://www.springerlink.com/index/10.1007/s10877-008-9128-x
[10]	MATTHEW, Neil. <i>Linux programujeme profesionálně</i> . Vyd. 1. Praha: Computer Press, 2001, 1079 s. ISBN 80-722-6532-6.

User manual

1. Connection to the database

When the application has been started it tries to get at the database logging information from the property files. If there is no such information, or if it is no longer valid, the following form is shown to set (and save) the new logging information.

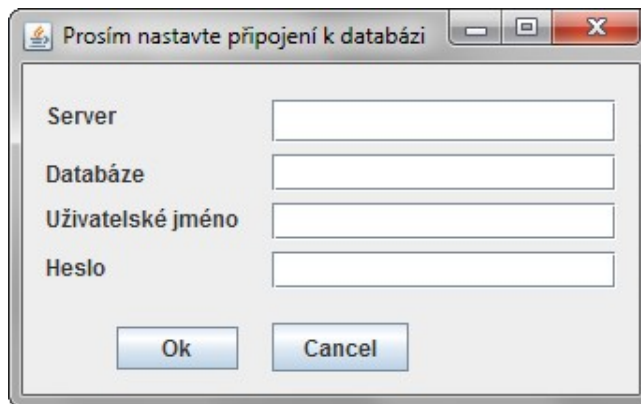


Illustration 11: Database information

2. Add and edit the patient

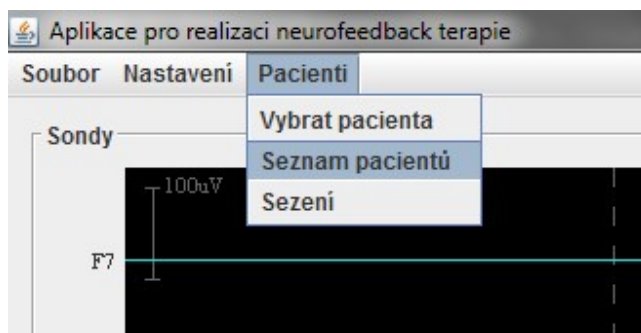


Illustration 12: Menu selection - patient list

Start with selecting the item “Seznam pacientů” from the “Pacienti” menu, as is shown on Illustration 12. Then the form (Illustration 13) will pop up.

Illustration 13: Add/Edit patient

Select a patient from the left column, or press the “Nový pacient” button to add a new patient, fill in the proper information and then save with the “Uložit” button.

3. Add and edit personnel

Select “Personál” from the “Nastavení” menu. Then pick one of the current persons from the left menu, edit the information in the form on the right and save the changes with the “Uložit změny” button. Or press the “Nový pracovník” button, fill in the information about the new person and save it to the database by pressing the “Přidat” button.



Illustration 14: Menu selection - add/edit personnel

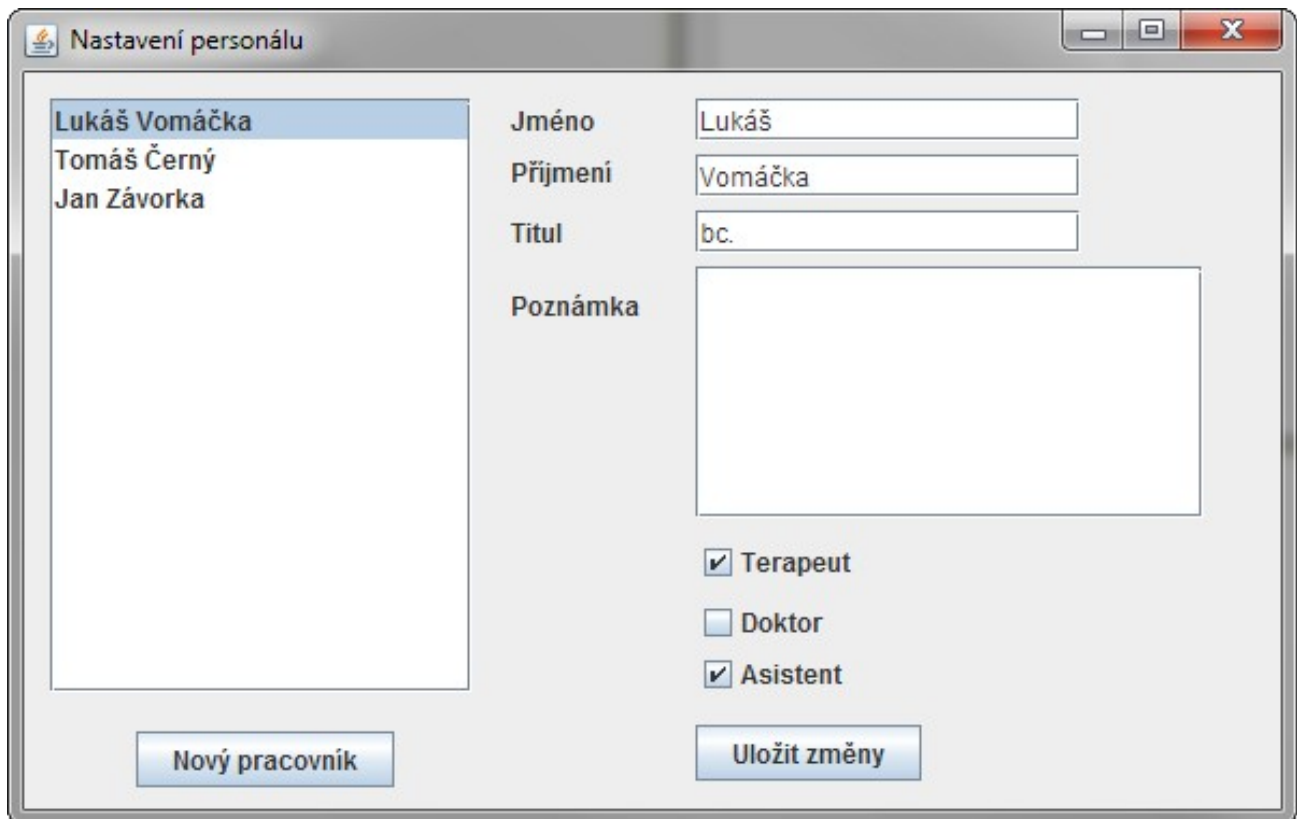


Illustration 15: Add/Edit personnel

4. Assign a therapy to the patient

Press "Vybrat pacienta" from the "Pacienti" menu (Illustration 16). In the following form select "patient" and in the "Přidat terapii" box select the preset used for this therapy, fill in the name of the new therapy, its description (Illustration 17) and the estimated number of sessions, then save with the "Přidat" button.

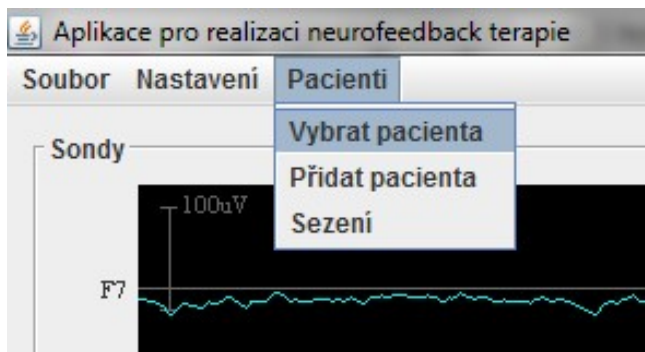


Illustration 16: Menu selection - Select patient

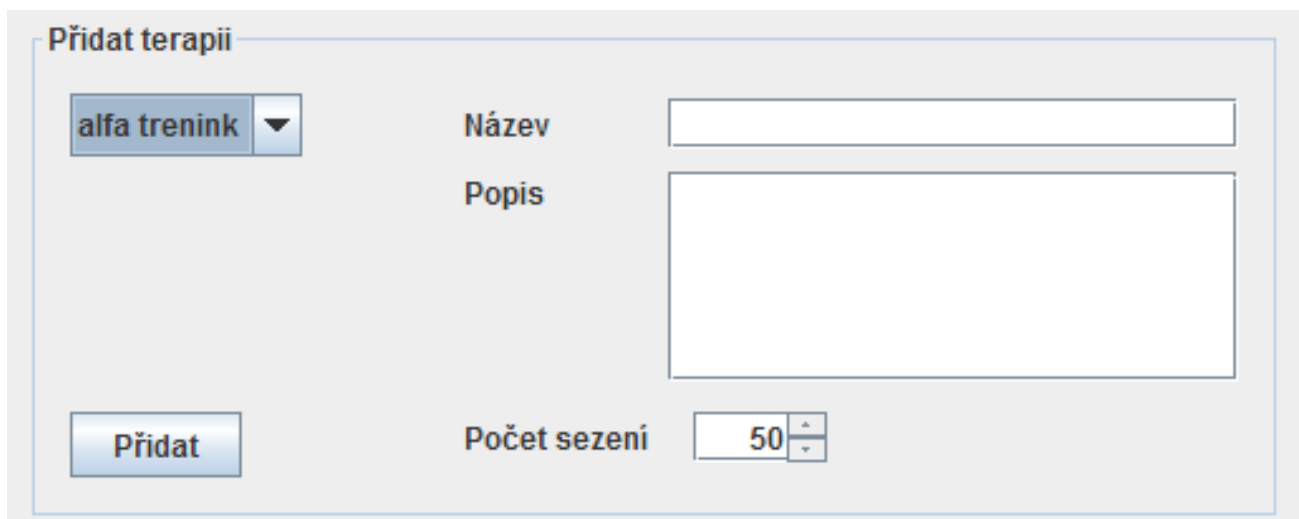


Illustration 17: Add new therapy

5. Plan a session

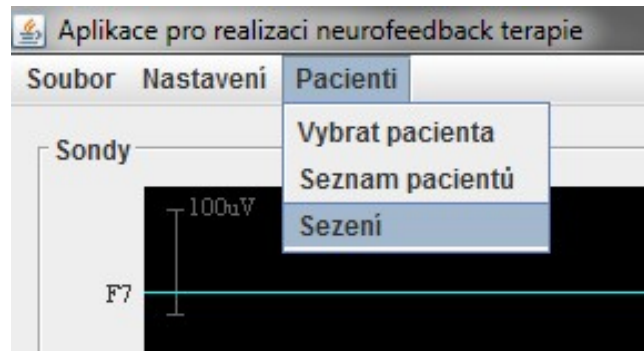


Illustration 18: Menu - sessions

Select “Sezení” from the “Pacienti” menu and then fill in the information and add the new session by pressing the “Přidat” button, or click on some existing session to select it and change some of its details.

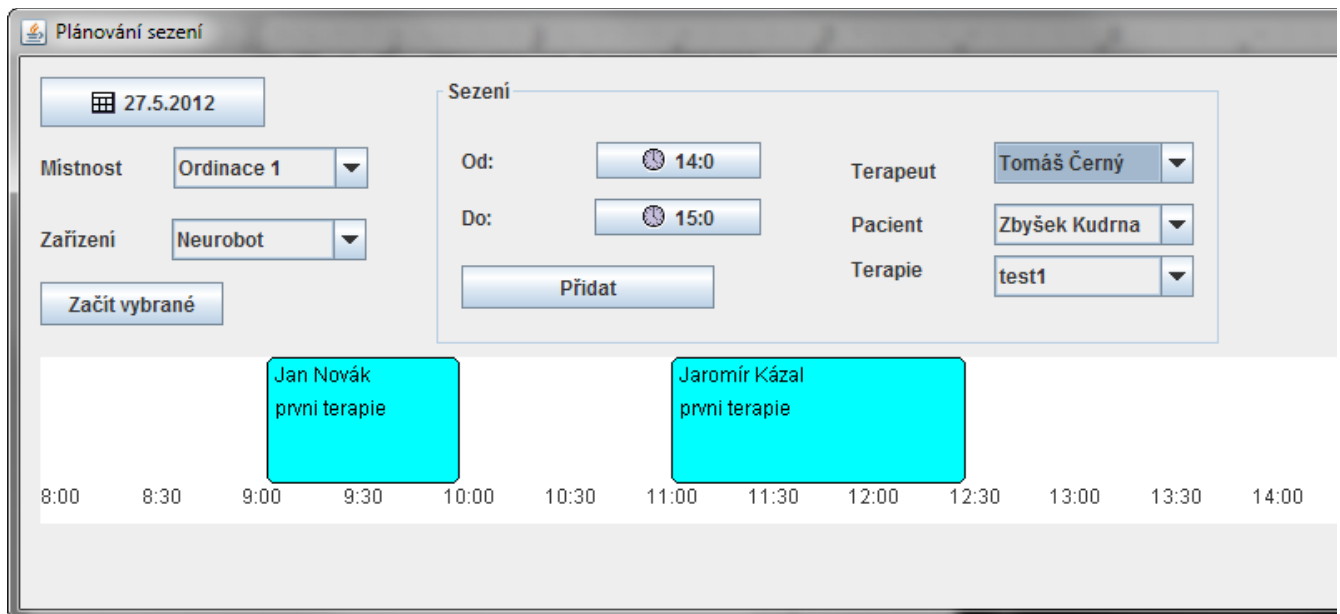


Illustration 19: Sessions planning

6. Start planned session

Open the same form as in , select a session and click the “Začít vybrané” button. All changes made before the session has started must be saved by using the “Uložit změny” button.

7. Start unscheduled session

Use the same menu as in (“Vybrat pacienta” from “Pacienti” menu). Then, instead of adding a new therapy, select the patient and the therapy which he attends, and click “Začít nové sezení” to start a new session.

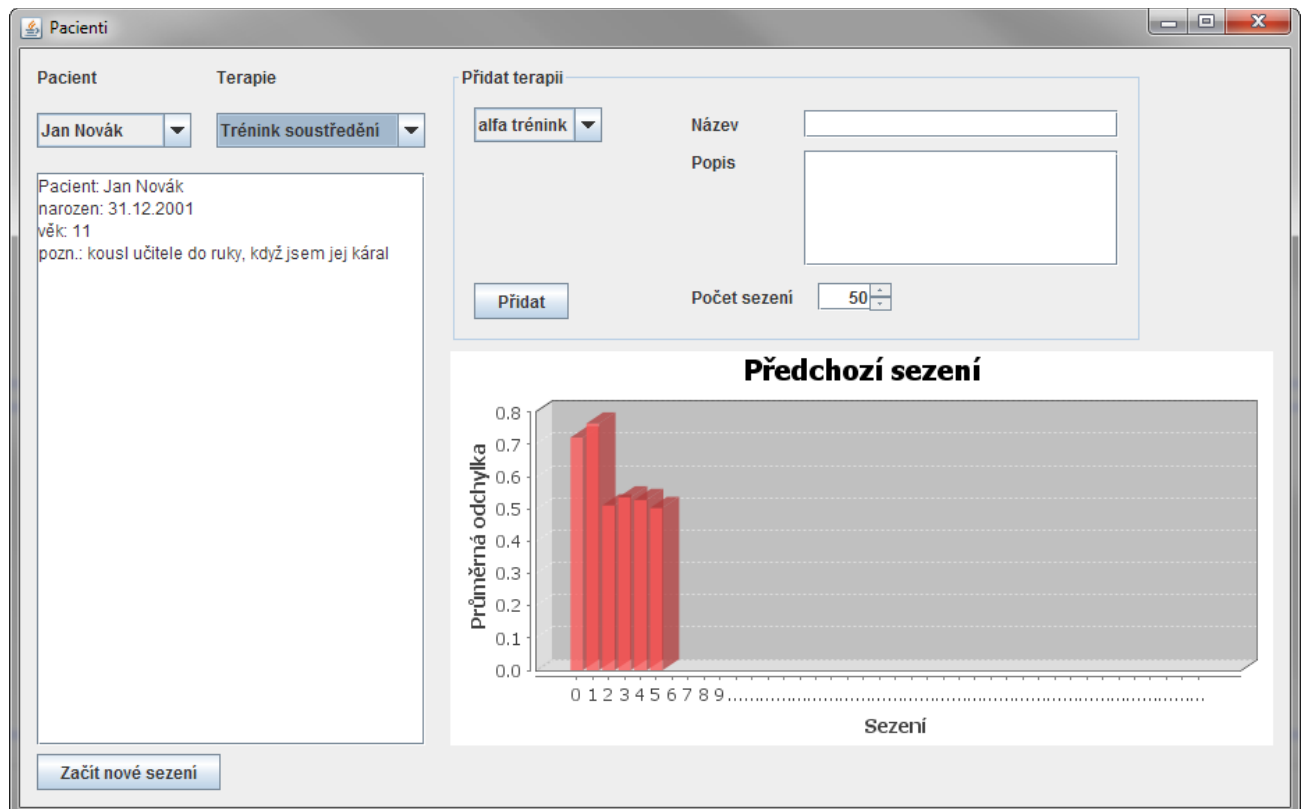


Illustration 20: Start new session