UNIVERSIDADE D
COIMBRA

Rodrigo Miguel Pimenta Simões

Laboratório Didático Virtual sobre Comunicações por Fibra Ótica

Dissertação no âmbito do Mestrado Integrado em Engenharia Física, ramo de Instrumentação orientada pelo Professor Doutor António Miguel Lino Santos Morgado e apresentada à Faculdade de Ciências e Tecnologias da Universidade de Coimbra.

Outubro de 2021

1 2 9 0 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE D COIMBRA

Rodrigo Miguel Pimenta Simões

Laboratório Didático Virtual sobre Comunicações por Fibra Ótica

Tese submetida à
Universidade de Coimbra, para o grau de
Mestre em Engenharia Física

Dissertação com orientação científica do
Professor Doutor António Miguel Lino Santos Morgado,
Departamento de Física

Coimbra, Outubro de 2021

Agradecimentos

Começo por agradecer ao Professor Miguel Morgado, por me ter proporcionado a oportunidade de desenvolver este trabalho, que unifica o que mais me cativou durante o meu percurso académico, a engenharia da luz e a programação. Agradeço-lhe, também, pela disponibilidade praticamente incondicional que mostrou. Deu-me a liberdade de experimentar ideias e de cometer erros com a segurança de que na próxima reunião teria a orientação para seguir o melhor caminho. Neste último ano cresci como engenheiro e como pessoa, por seguir o seu exemplo.

Agradeço à minha mãe, que incansavelmente me levou aos treinos de futebol, de natação, às aulas de música, catorze vezes por semana. Que me leu as histórias que me fizeram sonhar e que me fez os pequenos almoços ao acordar. Durante 18 anos colocou-me antes de si. Deu-me mais que dezoito anos, deu-me quem sou e tornou-me muito fácil viver, porque a melhor decisão que posso tomar é sempre a que a deixar mais orgulhosa. Obrigado, mamã.

Agradeço ao meu pai, que me fez acreditar em magia. Mais tarde fez-me acreditar em algo ainda mais fantástico, que era a ciência e a física. Ensinou-me a jogar xadrez e a tomar decisões. Ensinou-me a adiar a recompensa e a máxima: "Treino duro, combate fácil!" Tudo o que fazia, eu só queria saber fazer igual. Agradeço por me perguntar sempre "que nota tiveste?", não me deixando distrair do caminho. Obrigado, papá.

Às minhas irmãs, agradeço porque são as miúdas mais fixes do mundo. Espero deixar--vos tão orgulhosas como me orgulham a mim.

À Gé, porque chegar aqui sem ela teria sido impossível. Foi minha avó sem o ser, minha companheira de casa e minha melhor amiga.

A todos os meus professores, os meus amigos de Coimbra e à Secção de Xadrez da Associação Académica de Coimbra,

À Sofia e à minha família,

Um eterno obrigado.

Resumo

O surgimento da pandemia impôs às instituições de ensino a transição para um paradigma de ensino à distância. O novo ambiente de lecionação motivou a procura de soluções que permitissem preservar a qualidade pedagógica. Laboratórios tecnológicos são um exemplo de um ambiente de lecionação difícil de substituir, por proporcionarem aos alunos a oportunidade de testar os conhecimentos na prática e adquirir experiência na manipulação do equipamento laboratorial, bem como as boas práticas de segurança e de condução de experiências.

Uma solução estudada é a criação de laboratórios virtuais. O laboratório virtual é um programa de computador que simula o equipamento laboratorial e oferece uma interface para utilização das simulações. Estudos quantificaram ganhos pedagógicos consideráveis também no recurso a esta solução como preparação para a utilização dos laboratórios físicos e como ferramenta de estudo.

A presente dissertação expõe o processo de desenvolvimento de um laboratório virtual didático sobre comunicações por fibra ótica. O professor e o aluno podem utilizar o laboratório por meio de uma interface gráfica que tenta replicar as características do equipamento laboratorial real. O desenvolvimento divide-se em objetos distintos, que são a interface gráfica e as simulações. O projeto foi desenvolvido em Java, com recurso às ferramentas Swing e AWT. Foram simulados um gerador de sinais com modulação digital e sinusoidal, um LASER, um LED, fibras óticas de diferentes comprimentos, um fotodetetor e um osciloscópio.

O equipamento laboratorial simulado é utilizado na disciplina de Instrumentação Optoeletrónica do Departamento de Física e inclui um kit desenvolvido pela OptoSci (OptoSci Ltd, Glasgow, Scotland, UK) e um osciloscópio. O kit foi desenvolvido com o propósito didático e inclui um manual de experiências a realizar. O principal requisito que o laboratório virtual teve de cumprir foi o de possibilitar a realização dessas experiências.

Palavras-chave: pedagogia, laboratório virtual, simulação, optoeletrónica, instrumentação.

Abstract

The emergence of the pandemic imposed the transition to a remote paradigm on educational institutions. The new teaching environment motivated the search for solutions that would preserve pedagogical quality. Technological laboratories are an example of a teaching environment that is difficult to replace, as they provide students with the opportunity to test their knowledge in practice and gain experience in handling laboratory equipment, as well as good safety practices and conducting experiments.

One studied solution is the creation of virtual laboratories. A virtual laboratory is a computer programme that simulates laboratory equipment and provides an interface for using the simulations. Studies have quantified considerable pedagogical gains also in the use of this solution as a preparation for the use of physical laboratories and as a study tool.

This dissertation exposes the development process of a virtual didactic laboratory about fibre optic communications. The teacher and the student can use the laboratory by means of a graphical interface that tries to replicate the characteristics of the real laboratory equipment. The development is divided into distinct objects, which are the graphical interface and the simulations. The project was developed in Java, using the frameworks Swing and AWT. A signal generator with digital and sinusoidal modulation, a LASER, an LED, optical fibres of different lengths, a photodetector and an oscilloscope were simulated.

The simulated laboratory equipment is used in the discipline of Optoelectronic Instrumentation of the Physics Department and includes a kit developed by OptoSci (OptoSci Ltd, Glasgow, Scotland, UK) and an oscilloscope. The kit was developed for teaching purposes and includes a manual of experiments to be performed. The main requirement that the virtual laboratory had to fulfil was to make it possible to perform these experiments.

Keywords: pedagogy, virtual laboratory, simulation, optoelectronics, instrumentation.

Conteúdos

Lista de Figuras

Lista de Tabelas

1

Introdução

1.1 Tema

A presente dissertação foca-se no desenvolvimento de instrumentação virtual de características didáticas para utilização no ensino superior de comunicações por fibra ótica.

1.2 Enquadramento e motivação

A comunicação por fibra ótica tem um peso cada vez mais relevante no mercado das comunicações à distância, o que a torna um objeto de estudo importante para as instituições de ensino. Apesar da relevância ascendente desta tecnologia, o custo monetário dos componentes necessários a uma investigação mais profunda e prática do tema é elevado, sendo um entrave para muitas instituições.

A instrumentação virtual representa uma interface para interação com um sistema simulado. Este tipo de instrumentação permite, hoje em dia, uma flexibilidade elevada na criação de instrumentos, sem acarretar, grandes custos além do de um computador pessoal. A flexibilidade e a expansibilidade desta abordagem permitem que se façam as melhorias ao sistema que se considerem adequadas pedagogicamente, quer ao nível do instrumento, quer ao nível do *software* de simulação. Uma aplicação muito popular dos sistemas de instrumentos virtuais é no projeto de produtos, em que é necessário simular as várias condições de operação dos produtos a ser projetados.

Um laboratório virtual pode definir-se como um laboratório numa plataforma virtual que apresenta uma interface e funcionalidade semelhante ao seu equivalente físico [1]. Pode ser constituído pela conjugação de vários instrumentos virtuais e *software* de simulação. Para as academias, a utilização de laboratórios virtuais pode revelar muita utilidade, proporcionando um leque de possibilidades para o ensino. Enumeram-se algumas das vantagens:

- Segurança: é possível para o estudante cometer erros que poderiam acarretar custos financeiros ou riscos para a saúde, se os tivesse cometido num laboratório físico;

- Escalabilidade: as limitações logísticas impostas no acesso aos laboratórios físicos, principalmente para grandes grupos de alunos, quer por escassez de material ou qualquer outro fator imposto pela natureza do laboratório, é uma dificuldade colmatada pelos laboratórios virtuais. Cada aluno pode ter acesso à sua cópia do *software* do laboratório virtual;
- Portabilidade: se cada aluno pode ter acesso a uma cópia, também tem mais flexibilidade para utilizar o laboratório remotamente, para estudar, ou num contexto de ensino à distância, como foi imposto pela pandemia de COVID-19;
- Adequação: numa experiência real estão presentes efeitos de ordem superior que podem diminuir a utilidade didática da experiência, por fugirem ao plano curricular ou, simplesmente, por desviar as atenções dos aspetos mais importantes numa dada fase de aprendizagem. Um laboratório virtual pode ser construído de forma a evitar essas situações.

Estas vantagens são suficientes para suscitar interesse na criação de um laboratório virtual didático sobre comunicações por fibra ótica.

1.3 Objetivo

Pretende-se desenvolver um laboratório virtual para a realização de experiências didáticas sobre comunicação com fibras óticas. O sistema deverá ter como referência o kit ED-COM, desenvolvido pela OptoSci (OptoSci Ltd, Glasgow, Scotland, UK) e utilizado nas aulas laboratoriais da disciplina de Instrumentação Optoeletrónica. O kit inclui um modulador de sinal comutável, sinusoidal e digital, um gerador de corrente, um LED, um LASER, um fotodíodo, fibras óticas multimodo GRIN de 1 m, 1km e 2km e conectores. É necessário simular também um osciloscópio de duplo canal.

2

Estado da arte

O conteúdo académico sobre laboratórios virtuais aplicados às comunicações por fibras óticas é escasso. No entanto, nota-se um aumento do conteúdo produzido sobre laboratórios virtuais didáticos para o ensino superior, em diversas áreas, durante a pandemia do vírus COVID-19. Este aumento é fruto de uma transição considerável para a aprendizagem à distância. Neste capítulo, será apresentada uma visão geral do paradigma atual do Ensino e Aprendizagem à Distância (ED e AD), que será particularizada para a AD nos cursos de ciências e cursos tecnológicos e a consequente criação de laboratórios virtuais. Será ainda explorado o estado da arte da simulação nas comunicações por fibra ótica, destacando um exemplo de um laboratório didático virtual, neste mesmo tema.

Os desenvolvimentos tecnológicos observados no século XXI, principalmente ao nível da computação, digitalização da informação e da acessibilidade quase global à internet, têm um impacto curricular forte no paradigma de aprendizagem. A AD é potenciada por estes avanços, que se traduziram na criação de inúmeras ferramentas e plataformas *online* [2].

Na última década, testemunhou-se um crescimento em número e importância destas plataformas de AD. A Coursera é um exemplo de uma plataforma em que várias instituições de renome oferecem formações certificadas, a troco de um preço. A Udemy é uma plataforma mais ligada a formadores em *free lancing* e também é extremamente popular. A acessibilidade e abrangência da oferta de formações propícias ao desenvolvimento de *soft-skills* que nem sempre podem ser exploradas a fundo no espaço universitário, por haver um programa curricular a cumprir, torna estas plataformas incontornáveis para muitos alunos universitários, que precisam de se adaptar para entrarem num mercado de trabalho cada vez mais dinâmico [3].

Além de todos os cursos *online* que é possível frequentar, são de salientar os Recursos Educacionais Abertos (REA), acessíveis de forma gratuita, com o objetivo de proporcionar uma amostra da qualidade de ensino, elevando o bom nome e contribuindo para a sustentabilidade da instituição. O MIT OpenCourseWare foi uma das iniciativas de REA mais bem sucedidas, levada a cabo pelo Massachussets Institute

of Technology. O MIT é das instituições de ensino mais reconhecidas no mundo e com esta iniciativa consegue simultaneamente expor-se, captar talentos, solidificar a sua posição nas tabelas de instituições de ensino e proporcionar materiais de qualidade para estudantes de todo o mundo, mesmo que inscritos noutras instituições [4].

Ainda além do mercado de formação de *soft-skills* e da abordagem das iniciativas de REA, existem instituições de ensino superior cuja principal atividade é o ED. Em Portugal, existe uma instituição de ensino superior público à distância, a Universidade Aberta. O ensino superior tem um papel importante na concretização dos objetivos definidos para 2030 de educação para o desenvolvimento sustentável pelas Nações Unidas. O ED é uma prática cada vez mais corrente e está bem posicionado no mercado, mas mesmo os líderes de quatro das maiores instituições de ensino superior à distância da Europa divergem sobre o rumo deverá seguir o ED, numa ótica de sustentabilidade [5].

Desta forma, até mesmo antes da pandemia do Covid-19, o ensino presencial continuava a ser o estado da arte do ensino e o mais em linha com as metas definidas pelas Nações Unidas, sendo também a principal abordagem desde o ensino elementar até ao ensino superior.

O surgimento e rápido alastramento da pandemia forçou a mudanças drásticas nos mecanismos fundamentais da sociedade. Se os contactos presenciais são o principal fator de contágio, o ensino presencial passa a ser um fator contra a sustentabilidade. É necessária a criação de infraestruturas pedagógicas capazes de transições ágeis entre ensino presencial e ensino à distância [6]. Já eram estudadas as dificuldades de introdução de tecnologias de aprendizagem à distância no ensino superior antes da pandemia [2], de onde se poderá extrair informação, mas também já há estudos sobre os obstáculos à qualidade e soluções para este tipo de ensino durante a pandemia [7] [8].

Num estudo levado a cabo em universidades de vários países árabes, onde foi feito um inquérito a 100 professores e 300 alunos, algumas das categorias de obstáculos encontrados são os obstáculos pessoais, os pedagógicos e os organizacionais. Ao nível pessoal, existe uma desmotivação dos alunos perante aulas *online*, que se pode dever à dificuldade de compreensão sem a interação da sala de aula. Pedagogicamente, notou-se a dificuldade em levar a cabo trabalho orientado remotamente e em aprender disciplinas aplicadas. Organizacionalmente, observou-se uma deficiência no treino para a utilização de tecnologias e também dificuldades relacionadas com o ambiente doméstico em que os alunos e professores tentavam trabalhar, que não tinha as características da sala de aula como espaço orientado à lecionação. Neste estudo, uma

proposta à eliminação destes problemas foi a criação de cursos *online* de elevada qualidade que estejam acessíveis nos *websites* das universidades [7].

Esta é uma solução corroborada por outros estudos, que concluíram que os alunos sentem uma autonomia superior na gestão do tempo quando têm cursos permanentemente disponíveis *online* [2]. Especificando para os cursos de ciências e engenharias, é destacada, mais do que a criação de cursos *online*, a importância de criar o conteúdo que lhes permita ter as aulas laboratoriais à distância. Os artigos sugerem e estudam a pertinência da criação de laboratórios virtuais que permitam desempenhar essas tarefas num ambiente de simulação virtual [8][9][10][11][12].

Alguns destes estudos referem os laboratórios virtuais não apenas como auxiliares para a aprendizagem à distância, mas como ferramentas de estudo e complementares aos laboratórios físicos, por exemplo, para a realização de atividades pré-laboratoriais para preparação dos alunos [12][13]. Um estudo [12] investigou o impacto da utilização de laboratórios virtuais com a avaliação de três parâmetros: a metacognição (M; definida como o processo de desenvolvimento de autoconsciência de aprendizagem), raciocínio analógico (A; adaptação do conhecimento a cenários desconhecidos) e transferência de conhecimento (T; utilização num contexto diferente daquele em que o conhecimento foi adquirido). O método envolveu a criação de dois grupos de 145 alunos de uma disciplina de primeiro ano de engenharia. O primeiro grupo tinha uma aula teórica, uma laboratorial num laboratório físico e uma sessão num laboratório virtual. O segundo grupo trocava a ordem do laboratório físico e do laboratório virtual. No fim de cada aula era feita uma avaliação dos parâmetros M, A e T. Uma das análises dos resultados que foram destacadas mostrava que o grupo que foi primeiro ao laboratório físico teve uma melhoria global da classificação, à altura do último teste, de 70%, enquanto o grupo que utilizou primeiro o laboratório virtual teve uma melhoria global na classificação de 83,3%. A aula laboratorial era sobre a determinação do índice de refração de um prisma. A conclusão tirada foi que a integração de laboratórios virtuais nos programas das disciplinas pode melhorar o desempenho dos alunos, sendo sugerido que se façam estudos ainda mais abrangentes.

Especificamente para simulação em ótica e fotónica, a realidade virtual é utilizada principalmente em *software* de design profissional e científico[14][15][16], mas não com objetivo didático. Estas aplicações são construídas partindo do pressuposto que os seus utilizadores têm o domínio necessário das matérias, não permitindo que os estudantes aprendam sobre as matérias, pelo que não são ideais para utilização em sala de aula. Um laboratório virtual para este efeito teria de ser construído sobre uma

base curricular [13] e será neste estudo de Hayes *et al.* que se concentrarão os próximos parágrafos.

No estudo de Hayes, o *software* é composto por um módulo de prototipagem e um módulo de avaliação fluida. O primeiro módulo dá total liberdade ao estudante de manipular uma série de parâmetros, de forma a tirar as suas conclusões sobre a dependência entre eles e familiarizar-se com o equipamento laboratorial. Este módulo contém um laboratório virtual 3D. O laboratório é composto por modelos do díodo laser, uma lente biconvexa e uma fibra ótica. São ainda representados graficamente cones luminosos para a luz emitida pelo laser, a luz refratada pela lente e o cone de aceitação da fibra ótica. Os parâmetros que se podem manipular são:

-as distâncias entre o laser e a lente e entre a lente e a fibra;

-a potência, o ângulo de dispersão e a área da abertura do díodo laser;

-a distância focal da lente;

-o diâmetro e a abertura numérica da fibra.

O segundo módulo consiste numa avaliação por etapas: três níveis em que a progressão se dá quando o aluno responde corretamente às perguntas que constituem o nível e que avaliam a compreensão dos mecanismos de perda nos acoplamentos de um sistema simples formado por um laser, uma lente e uma fibra ótica. O insucesso num nível obriga o estudante a repetir esse nível. Cada nível ativa ou desativa a manipulação de certos parâmetros, dependendo do acoplamento em estudo. O primeiro nível é sobre as perdas no acoplamento devido à incompatibilidade entre as áreas da abertura do laser, da lente, da secção do feixe depois da lente e da entrada da fibra ótica. O segundo nível serve para o estudo da perda no acoplamento devida à incompatibilidade entre as aberturas numéricas do laser, da lente e da fibra. O terceiro nível agrega todos os fatores de perda em estudo, sendo que o estudante pode manipular todos os parâmetros. As respostas dos alunos são gravadas.

Antes e depois da atividade laboratorial virtual, os alunos são submetidos a uma avaliação diagnóstica e a uma avaliação final, respetivamente. A proporção de respostas corretas no primeiro teste foi de 0,45±0.15 e no último foi de 0.56±0.15. O teste de T-Student confirma que o aumento é estatisticamente significativo, com $t(12) = -3.386$, e o valor p a valer 0.005.

Ainda no mesmo estudo, foi concluído que os alunos consideraram muito útil a flexibilidade na manipulação dos parâmetros, que lhes conferiu um nível de controlo sobre o sistema que seria impossível num laboratório físico. O estudo admite ainda o interesse em desenvolver o módulo, aumentando a complexidade e expandindo a sua abrangência, podendo ainda implementar-se num curso *online*. Os alunos consideraram

que seria benéfico ter este tipo de laboratório implementado de forma ainda mais abrangente no seu currículo.

Outros exemplos de temas onde foi explorada a simulação no ramo da fotónica são uma fórmula melhorada para a perda devida à curvatura de uma fibra ótica [17], tanto mono como multimodal, a simulação de um amplificador de fibra ótica utilizando fibras curtas equivalentes [18] e a simulação de uma fibra ótica multimodal utilizando o meio de simulação OptiBPM [19] (seria pertinente explorar o artigo mais a fundo, mas a língua em que foi escrito constitui uma barreira para a sua compreensão). O livro de Le Nguyen Binh “Optical Fiber Communication Systems with MATLAB and Simulink Models” [20] é uma bibliografia amplamente compreensiva sobre todas as etapas de simulação de um sistema de comunicação por fibra ótica, baseando-se em modelos MATLAB e Simulink. Apesar de este livro pretender ser um manual para construção de simulações virtuais a um nível profissional, também pode ser utilizado como apoio à construção de um sistema mais simples, com intuito pedagógico.

Resumindo, os laboratórios didáticos virtuais já provaram oferecer um ganho pedagógico considerável, quer em regime de ensino à distância, quer como complemento aos laboratórios físicos. Apesar de não haver muitas publicações sobre este tipo de laboratórios dedicados à comunicação por fibra ótica, é reconhecida a sua necessidade e existe intenção de desenvolvimento do tema.

3

Metodologia

Há duas tarefas principais associadas à criação de um laboratório virtual: o desenvolvimento de uma interface e o desenvolvimento de toda a funcionalidade necessária. A interface exige a criação de um ambiente virtual, que consiste num programa de computador. Para garantir a funcionalidade, será necessário simular o sistema físico, que no caso é o de um sistema de comunicação por fibra ótica e um osciloscópio – a análise detalhada dos requisitos será feita mais adiante, no subcapítulo **3.1**. O carácter multidisciplinar do projeto a desenvolver, bem como a dimensão prevista de cada componente, torna-o uma tarefa de dimensões consideráveis, pelo que a execução deve ser planeada com antecedência e o mais detalhadamente possível.

Se o laboratório virtual for interpretado em abstração como um instrumento, o planeamento fica imediatamente simplificado, porque pode herdar o processo clássico de desenvolvimento de um instrumento, com uma primeira fase de conceção e uma segunda fase de projeto, propriamente dito.

Na fase de conceção, é definido o conceito do instrumento, por definição do seu objetivo e construção de um conjunto de requisitos, levada a cabo após análise dos sistemas a simular.

Na fase de projeto, dá-se a execução do processo de desenvolvimento, seguindo as linhas do diagrama da figura 1, que ilustra a metodologia a empregar.

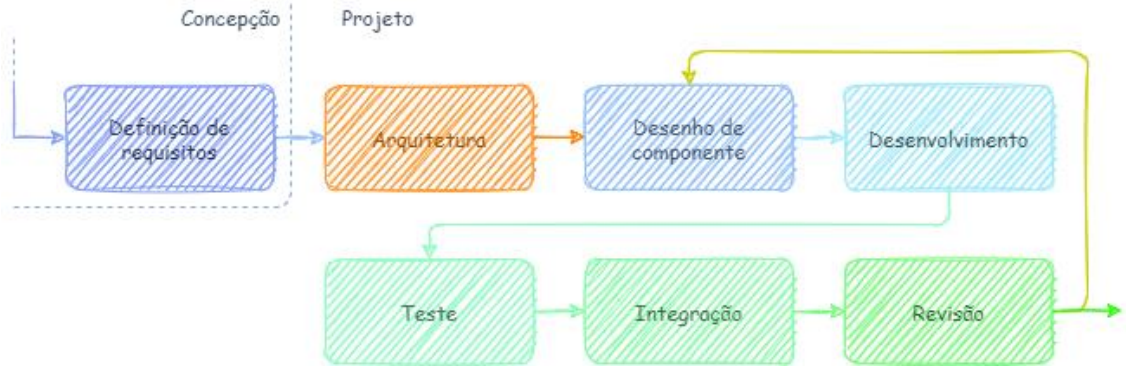

Figura 1. Metodologia de desenvolvimento

A definição de requisitos é a última etapa da fase de conceção.

Relativamente ao projeto:

- Arquitetura: estruturação do sistema de *software* pelas suas componentes e relações entre componentes [21];
- Desenho de componente: descrição de uma proposta de solução para a componente ou funcionalidade respetiva, que faça cumprir os requisitos que lhe estão associados;
- Desenvolvimento: execução do desenho para a sua forma mais concreta;
- Teste: verificação do bom funcionamento da solução;
- Integração: quando a solução da componente ou funcionalidade passa os testes com sucesso, é integrada na solução global em desenvolvimento;
- Revisão: avaliação do comportamento da solução global em desenvolvimento com a adição da nova componente.

É seguida uma linha desde o desenho à revisão. Nesta última etapa, se for apontado algum defeito ou detetada alguma correção conceptual à componente adicionada, pode ser necessário voltar à etapa de desenho. Este ciclo representa o processo iterativo de desenvolvimento, tanto para cada componente, como para a solução global em desenvolvimento, que é o resultado da integração das várias componentes e funcionalidades.

3.1 Conceção

Quer-se desenvolver um laboratório virtual didático sobre comunicações por fibra ótica, que tenha por base o kit ED-COM da OptoSci.

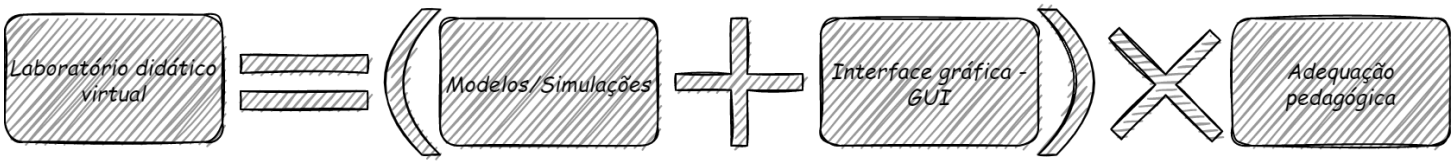

Figura 2. Conceito do laboratório virtual

O laboratório será composto por modelos e simulações, que replicam o ambiente físico que é objeto de estudo, e uma GUI – *Graphical User Interface* – que permite ao utilizador do laboratório interagir com esse ambiente físico através de construções gráficas que sejam claras em funcionalidade e relativamente ao instrumento físico a que se referem. Este conjunto deve ser controlado por uma matriz de pesos de forma a garantir a adequação pedagógica da solução final. O facto de estar a ser criado um laboratório virtual possibilita a adaptação de algumas características do laboratório

físico, no sentido de reforçar os efeitos que se pretendem estudar e ensinar e suprir os que fogem ao conteúdo programático das atividades laboratoriais.

3.1.1 Conceitos de comunicações por fibras óticas

Antes de proceder à descrição do kit e do aparato experimental, pode ser útil uma breve explicação dos conceitos de comunicações por fibras óticas envolvidos nas atividades laboratoriais.

Num sistema de comunicação digital por fibras óticas, um díodo laser é modulado por uma corrente elétrica com a codificação da informação em bits. O díodo laser envia os 1s digitais na forma de pulsos óticos para a fibra ótica. A fibra ótica nestes sistemas pode ter muitos quilómetros de comprimento – veja-se o exemplo da rede Fibre-optic Link Around the Globe (FLAG), com cerca de 28 000 km de comprimento em fibra ótica, obviamente não contínuos [22] – o que degrada a qualidade do sinal recebido na outra extremidade da fibra, pelo fotodetetor, devido aos mecanismos de atenuação e dispersão. Durante a propagação dos pulsos na fibra, há, por um lado, perdas de energia, que atenuam a intensidade dos pulsos. Por outro lado, a dispersão causa um aumento da duração de cada pulso. A combinação destes fatores faz com que quanto maior o comprimento da fibra, maior a probabilidade de haver um erro na leitura da informação contida no sinal.

Também num sistema analógico, quanto maior for o comprimento da fibra, maior a atenuação e menor será a Relação Sinal-Ruído (SNR – Signal to Noise Ratio). Já a dispersão, criará componentes de frequências que irão causar interferências destrutivas no sinal transmitido, também contribuindo para a diminuição da SNR.

O desempenho de um sistema de comunicações está limitado pelas características das fibras óticas utilizadas. Em particular, a atenuação e a dispersão limitam o comprimento máximo entre cada ligação, a taxa de transmissão de dados e a largura de banda do sistema.

3.1.2 Descrição do kit e das atividades laboratoriais a simular

O kit ED-COM foi desenvolvido pela OptoSci, uma empresa de soluções para a fotónica fortemente ligada ao Departamento de Engenharia Eletrónica e Eletrotécnica da Universidade de Strathclyde, Glasgow.

Os objetivos das atividades experimentais que o kit permite realizar em laboratório incluem a caracterização de todos os principais componentes de um sistema de comunicações por fibra ótica, a construção de uma ligação ponto a ponto simples e avaliar experimentalmente o desempenho dessa ligação.

O kit é constituído por materiais e tecnologias menos avançadas do que as realmente utilizadas nos sistemas comerciais típicos de comunicações por fibra ótica.

Este compromisso é benéfico de duas formas, pois diminui o custo do produto para as universidades, bem como amplifica os efeitos de atenuação e dispersão a ser estudados nas atividades laboratoriais, tornando-os mais evidentes para os alunos, o que traz vantagens pedagógicas.

As fontes de luz utilizadas emitem próximo dos 800 nm, para obter valores de dispersão e atenuação superiores aos observados nos sistemas de comunicação ótica, que utilizam fontes de luz com emissão nos 1300 nm ou nos 1550 nm. Para compatibilidade com o osciloscópio de cerca de 50 MHz normalmente utilizado nos laboratórios didáticos, que tem uma banda larga relativamente baixa comparativamente à dos sistemas de comunicação comerciais, a fibra ótica fornecida é construída com índices de refração não ideais, para que a dispersão seja suficientemente elevada.

3.1.2.1 Aparato experimental

Em baixo é apresentada a lista de equipamento da figura 3. Este é todo o equipamento necessário para a concretização das atividades experimentais. Exceto o osciloscópio, todo ele é fornecido pela OptoSci:

- Um LED com pico de emissão nos 850nm, com corrente de controlo (excitação) ajustável e modulável;
- Um díodo LASER com pico de emissão nos 785 nm, corrente de controlo ajustável e modulável;
- Dois rolos de fibra ótica multimodo (rolos #1 e #2), de índice gradual (graded index) de aproximadamente 1km e 2km de comprimento, núcleo de 62.5um, abertura numérica (NA) de 0.275 e índice de refração de 1.497;
- Um cabo de fibra do mesmo tipo, mas apenas com um metro de comprimento;
- Um recetor fotodíodo de silício;
- Um gerador de ondas que pode comutar entre uma onda quadrada de 4MHz e uma sinusoidal de frequência variável entre 1 MHz e 28 MHz. Tem uma tensão pico a pico máxima de 10 V e um botão de ligar e desligar – permitindo também a operação das fontes de luz em regime contínuo (CW – conctinuous wave);
- Uma fonte de alimentação integrada;
- Um osciloscópio de dois canais com uma largura de banda mínima de 50MHz e uma escala temporal de pelo menos 5 ns/div.

Figura 3. Equipamento OptoSci e osciloscópio [24]

3.1.2.2 Atividades laboratoriais

Este subcapítulo é uma recolha dos pontos de interesse da descrição das experiências que se pode encontrar no manual Fibre Optic Communications Educator Kit (ED-COM) Student Manual. Para mais detalhe, sugere-se a consulta do próprio documento.

1. **Comparação entre as características do LED e do LASER díodo:**

1.1 Potência ótica de saída vs. Corrente de excitação:

O aluno deve conectar a fibra de 1 metro entre o LED e o fotorrecetor e anotar a potência ótica detetada em função da corrente de excitação. Deve fazer o mesmo para o LASER e obter a curva potência ótica de saída vs. corrente de excitação para as duas fontes luminosas.

1.2 Potência ótica:

O aluno deve anotar a potência ótica emitida pelo LED e pelo LASER com a fibra de 1 metro, nas respetivas correntes de polarização. A corrente de polarização será a corrente de operação a ser mantida para as restantes experiências.

2. Atenuação nas fibras óticas:

2.1 Perdas nos conectores:

O aluno faz duas medições da potência ótica no fotorrecetor. A primeira medição é com o rolo de 1 km de comprimento e a segunda é com a fibra de 1 m conectada ao rolo #1 por um conector ST. O procedimento é feito com o LED e repetido para o LASER.

2.2 Atenuação ao longo da ligação:

O aluno verifica as potências medidas na experiência 1.2. Depois, mede a atenuação total após propagação pelo rolo #1 e pelo rolo #2.

2.3 Determinação do comprimento da fibra:

O comprimento da fibra é estimado pela medição do atraso do sinal transmitido através da fibra, relativamente ao sinal sinusoidal vindo diretamente do gerador de ondas. Com os dois sinais visíveis no osciloscópio, numa base temporal adequada e com o *trigger* definido para um dado ponto do sinal de modulação, o aluno deverá variar a frequência deste sinal até conseguir detetar o ponto do sinal de saída em que a fase permanece inalterada. O atraso temporal entre os dois sinais é o indicado pela separação entre esses dois pontos no eixo horizontal do osciloscópio.

A partir da informação no osciloscópio e conhecendo o índice de refração das fibras, o aluno deve determinar o comprimento do rolo de fibra ótica e o coeficiente de atenuação por quilómetro da fibra ótica.

2.4 Determinação de comprimentos de ligação limitados pela atenuação:

O aluno deve conectar o díodo LASER diretamente ao fotodíodo com a fibra de 1m, aplicar a modulação de onda quadrada e medir a amplitude pico a pico do sinal ótico. Depois, sem sinal ótico, deve estimar a amplitude RMS do ruído, por inspeção do sinal detetado no osciloscópio. A partir da medição, deve determinar a sensibilidade do fotorrecetor e calcular o comprimento máximo de um sistema de comunicações limitado pela atenuação. A experiência deve ser repetida para o LED.

3. <u>Medições de largura de banda e de dispersão:</u>

3.1 Medições no domínio do tempo:

O aluno deve excitar o LED com modulação digital e determinar o tempo de subida dos pulsos à saída da fibra. A partir dessa medição, ele determina a largura de banda do sinal, o produto largura de banda – comprimento (BW.L) e o produto taxa de bits-comprimento (BR.L). Para extrair o tempo de subida dos pulsos na fibra, deve medir com o osciloscópio tanto o tempo de subida dos pulsos no sistema transmissor-recetor (utilizando apenas a fibra de 1m) como o tempo de subida dos pulsos com o sistema de comunicação completo, ou seja, com os rolos de fibra ótica.

A experiência deve ser repetida para o laser.

3.1.3 Requisitos

Conhecidas as atividades laboratoriais podemos elaborar a lista de requisitos, por seleção do equipamento a simular virtualmente e das funcionalidades a desenvolver.

GUI:

- Deve conferir uma utilização intuitiva do laboratório, com funcionalidade e elementos gráficos semelhantes aos encontrados no laboratório físico.

Osciloscópio:

- duplo canal;
- trigger;
- dois ponteiros no tempo;
- dois ponteiros em tensão;
- escala temporal mínima de 5 ns/div;
- escala de tensão mínima de mV e máxima tal que permita visualizar 10 V pico a pico no mostrador ;
- mostrador adicional para informações sobre os sinais, como intervalos entre ponteiros, frequências e tensões pico a pico.

Gerador de sinal:

- Botão on/off para a modulação;
- Modulação por onda quadrada com frequência fixa de 4 MHz;
- Modulação por onda sinusoidal de frequência variável entre 1 MHz e 25 MHz;
- Modelação de ruído;
- Tensão pico a pico máxima de 10 V.

LED:

- Pico de emissão nos 850 nm;
- Largura espectral rms de 30 nm;
- Controlo de corrente de excitação;
- Mostrador de corrente de excitação;
- Entrada de modulação;
- Saída ótica.

LASER:

- Pico de emissão nos 785 nm;
- Largura espectral rms de 1 nm:
- Controlo de corrente de excitação;
- Mostrador de corrente de excitação;
- Entrada de modulação;
- Saída ótica.

Fotorrecetor:

- Entrada ótica;
- Saída em tensão;
- Mostrador de potência em μW.

Fibras óticas:

- Multimodo 1 m;
- Multimodo 1 km;
- 1 m + conector + 1 km.

A partir desta lista e da descrição das atividades laboratoriais a simular, pode elaborar-se a lista das histórias do utilizador e dos casos de uso.

3.1.3.1 Histórias de utilizador (User Stories)

Uma história de utilizador é uma descrição informal de uma característica de um *software* escrita da perspetiva do utilizador. O objetivo é esclarecer como é que uma característica irá proporcionar valor ao utilizador [23], constituindo um complemento à lista de requisitos amplamente utilizado no desenvolvimento de *software*.

As histórias de utilizador constroem-se da seguinte forma:

"**Como** <sujeito>, **quero** <funcionalidade/característica>, (**para** <finalidade>)."

Histórias de utilizador sobre a simulação do gerador de sinal:

a) **Como** aluno, **quero** um gerador de sinal, **para** modular um LED e um LASER;
b) **Como** aluno, **quero** um botão on/off, **para** ativar ou desativar a modulação de sinal;
c) **Como** aluno, **quero** um manípulo **para controlar** a frequência de modulação;
d) **Como** aluno, **quero** um comutador de modulação, **para** alternar entre modulação digital (onda quadrada) e analógica (sinusoidal);
e) **Como** aluno, **quero** gerar ondas quadradas à frequência de 4 MHz;
f) **Como** aluno, **quero** gerar ondas sinusoidais de frequência variável;
g) **Como** aluno, **quero** uma saída, **para** controlar eletricamente um LED e um LASER.

Histórias de utilizador sobre a simulação do LED:

a) **Como** aluno, **quero** operar uma simulação de um LED de 850 nm, com desvio rms de 30 nm, **para** converter sinal elétrico em sinal ótico;
b) **Como** aluno, **quero** um manípulo, **para** controlar a amplitude da corrente de excitação;
c) **Como** aluno, **quero** uma entrada, **para** modular a corrente de excitação;
d) **Como** aluno, **quero** uma saída, **para** emitir sinal ótico, para uma fibra ótica;
e) **Como** aluno, **quero** que o módulo do LED tenha um mostrador, **para** ver a amplitude da corrente de excitação.

Histórias de utilizador sobre a simulação do LASER:

a) **Como** aluno, **quero** operar uma simulação de um LASER de 785 nm, com desvio rms de 1 nm, **para** converter sinal elétrico em sinal ótico;
b) **Como** aluno, **quero** as mesmas funcionalidades que o LED, bem como o mostrador de corrente de excitação.

Histórias de utilizador sobre a simulação de fibras óticas:

a) **Como** aluno, **quero** simulações de ligações por fibra ótica de 1 m, 1 km e 1 m + conector + 1 km, **para** transportar o sinal ótico do LED ou LASER para o fotorrecetor.

Histórias de utilizador sobre a simulação do fotorrecetor:

a) **Como** aluno, **quero** uma simulação de um fotorrecetor, **para** ver a potência ótica recebida **e** transmitir o sinal ótico convertido em elétrico num osciloscópio;
b) **Como** aluno, **quero** uma entrada, **para** receber um sinal ótico;
c) **Como** aluno, **quero** converter o sinal ótico num sinal elétrico;
d) **Como** aluno, **quero** medir a potência do sinal recebido;
e) **Como** aluno, **quero** uma saída em tensão, **para** fornecer o sinal a um osciloscópio.
f) **Como** aluno, **quero** que o módulo do fotorrecetor tenha um mostrador onde possa ver a potência do sinal recebido.

Histórias de utilizador sobre o osciloscópio:

a) **Como** aluno, **quero** uma simulação de um osciloscópio, **para** estudar o sinal transmitido;
b) **Como** aluno, **quero** que o osciloscópio seja de duplo canal;
c) **Como** aluno, **quero** que o osciloscópio tenha uma largura de banda mínima de 28 MHz;
d) **Como** aluno, **quero** que o osciloscópio tenha um trigger configurável;
e) **Como** aluno, **quero** que o osciloscópio tenha escalas ajustáveis de tensão e de tempo;
f) **Como** aluno, **quero** que o osciloscópio tenha dois ponteiros em tensão e dois ponteiros no tempo;
g) **Como** aluno, **quero** que o osciloscópio tenha um painel de informações **para** ler a frequência de cada sinal, a tensão pico a pico de cada canal e as informações dos ponteiros;
h) **Como** aluno, **quero** que o osciloscópio tenha um painel onde mostre os sinais detetados num gráfico de tensão em função do tempo.

A consulta desta lista permitirá organizar a fase de projeto de forma eficiente.

3.2 Projeto

3.2.1 Arquitetura

A fase de projeto inclui o desenho da componente, o desenvolvimento, o teste, a integração da componente no sistema e a revisão/avaliação. Para fazer a primeira ligação entre os requisitos e o desenho, deve fazer-se a arquitetura do sistema.

A arquitetura terá os seus próprios requisitos, a partir dos quais se faz o desenho da mesma [25].

Tabela 1. Requisitos da arquitetura

Atributo Qualitativo	**Requisito de Arquitetura**
Desempenho	O desempenho da aplicação deve proporcionar uma taxa de atualização do osciloscópio indistinguível a olho humano da de um osciloscópio real.
Segurança	Não há requisitos de segurança.
Gestão de recursos	O sistema deve funcionar sem atrasos num computador pessoal típico moderno (que tem normalmente 8GB de RAM) e sem comprometer outras funcionalidades dessa máquina.
Usabilidade	O sistema deve ser utilizável sem conexão à internet e a interface gráfica deve precisar de uma variedade de componentes.
Compatibilidade/portabilidade	O sistema deve ser compatível com computadores independentemente do sistema operativo ou do hardware.
Confiabilidade	O sistema deve ser robusto de forma a não perder desempenho nem com o decorrer do tempo nem com a acumulação de instruções.

Modificabilidade	A arquitetura deve suportar a adição de novas funcionalidades de forma facilitada.

Para garantir o desempenho, a gestão de recursos e a confiabilidade, o ponto crítico a considerar é a simulação. Quer simular-se um sistema ótico praticamente indistinguível daquele em laboratório, mas é impraticável num computador comercial comum fazer uma amostragem fotão a fotão, por muito eficiente que seja o algoritmo. Deve encontrar-se um meio termo que não comprometa nenhum dos atributos qualitativos referidos.

Para a usabilidade, compatibilidade e modificabilidade, o ponto crítico passa a ser a linguagem de programação e as tecnologias utilizadas. Para alguém com pouca experiência de programação e com apenas três linguagens no reportório, é preferível analisar primeiro se alguma delas cumpre os requisitos de arquitetura especificados. As linguagens são Java, Python e MATLAB, mas MATLAB não é gratuito, pelo que apenas Java e Python serão considerados.

Tabela 2. Comparativo Java vs. Python

Atributos Qualitativos	**Python**	**Java**
Desempenho	Para proporcionar maior flexibilidade ao desenvolvedor, como tipagem dinâmica e programas mais curtos, python é uma linguagem interpretada [26], mas apesar de utilizar tecnologias desenvolvidas de compilação dinâmica (*just-in-time compilation*) [27], continua a sacrificar eficiência durante a execução do programa.	É simultaneamente compilada e interpretada, no sentido em que o programa é primeiro compilado para *bytecode* e depois interpretado na *Java Virtual Machine* (JVM). Isto permite que o interpretador seja mais pequeno e eficiente, proporcionando uma boa eficiência durante a execução, mais próxima de linguagens compiladas como C++ [28].

Usabilidade	Tanto Python como Java têm disponíveis ferramentas de desenvolvimento de interfaces gráficas flexíveis e bem documentadas.	
Compatibilidade/ portabilidade	Tem uma excelente portabilidade. Existem compiladores que possibilitam que o código python possa ser executado em sistemas operativos diferentes, mas esses compiladores têm que ser instalados de acordo com cada plataforma.	Java tem uma portabilidade ligeiramente melhor que o python, simplesmente por apenas precisar da JVM para que o código seja compreendido pelo sistema operativo e atualmente a JVM já vem instalada na maioria das plataformas. Em 2016, 15 mil milhões de dispositivos corriam código Java e esta linguagem de programação é ainda hoje a mais utilizada por desenvolvedores em todo o mundo [29].
Modificabilidade	Não é tão dependente da linguagem, mas do desenho da arquitetura em si. Uma arquitetura de micro-serviços proporciona excelente modificabilidade porque torna mais fácil implementar e implantar as atualizações. Como o objetivo da aplicação em questão não é um desenvolvimento industrial, basta uma boa implementação de programação orientada por objetos para garantir que as alterações possam ser facilmente localizadas. Ambas linguagens suportam o paradigma de OOP (*Object Oriented Programming*).	

É de acrescentar que ambas as linguagens estão bem documentadas e são das mais populares do mundo, pelo que se espera haja na internet muito conteúdo útil para o desenvolvimento.

A linguagem de programação escolhida é o Java, principalmente pelo melhor desempenho que oferece e que é importante para a simulação.

3.2.1.1 Desenho da Arquitetura

A aplicação será modular e inteiramente contida numa única plataforma.

A aplicação contará com dois componentes principais, o osciloscópio e o ED-COM, sendo que este último refere-se a todo o equipamento do sistema de comunicações por fibra ótica.

No osciloscópio, haverá um painel que apresenta o gráfico dos sinais recebidos. Esse painel terá uma taxa de atualização constante em que haverá uma comunicação síncrona (isto é uma análise *a priori* e ainda não é evidente quem serão os intervenientes na comunicação) com as demais componentes do sistema, que definirá os parâmetros para desenhar os gráficos.

Toda a funcionalidade relacionada com a interação com o utilizador, no sentido de obter as suas instruções, terá um funcionamento assíncrono, típico de uma arquitetura baseada em eventos. Um evento é qualquer ocorrência significante ou mudança de estado no *software* ou *hardware* de um sistema [30], por isso cada interação do utilizador com os componentes da aplicação constitui um evento.

Uma arquitetura de barramento por eventos (*Event Bus*) contem criadores de eventos, um intermediário (ou mais) e subscritores. O diagrama na figura 4 constitui um exemplo ao nível de conceito.

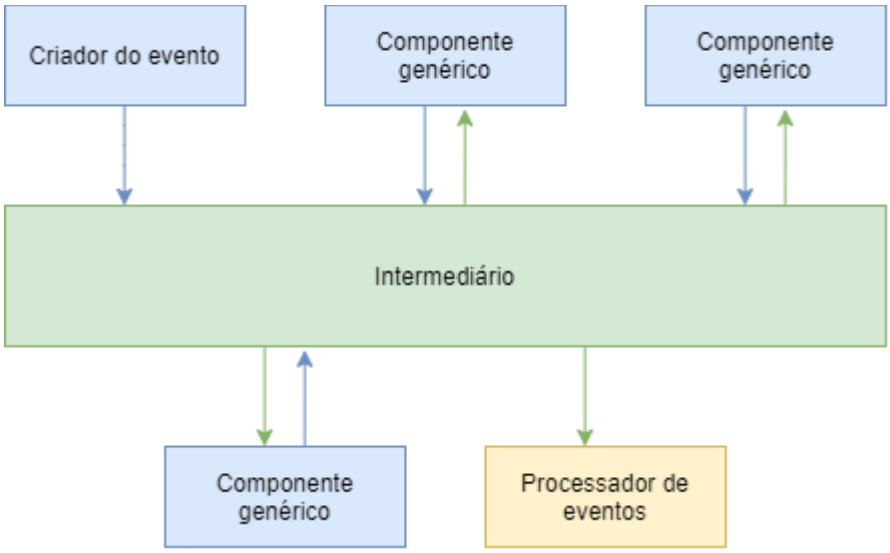

Figura 4. Conceito da arquitetura de barramento por eventos

Este é um padrão de arquitetura distribuída assíncrona capaz de criar aplicações altamente escaláveis e reativas [31]. A escalabilidade não é um requisito do projeto, mas é uma clara mais-valia para trabalhos futuros. Não o é, neste caso, ao nível de poder albergar mais utilizadores, mas por permitir, por exemplo, adicionar um LASER com outro comprimento de onda, ou fibras óticas diferentes, se pretendido.

A figura 5 mostra desenho do que seria a arquitetura na aplicação em desenvolvimento. Porque o sistema é totalmente contido na mesma plataforma, pode

implementar-se o mesmo conceito de arquitetura de forma mais simplificada, como mostra o desenho na figura 6. A arquitetura desenhada é uma adaptação da arquitetura mostrada na figura 5. Todos os componentes são objetos (instanciações das respetivas classes) instanciados no EDU_COM. Nesta implementação, num paradigma de orientação a objetos, o EDU_COM pode definir nos objetos os próprios processadores de eventos, que comunicam então as instruções adequadas aos outros objetos, porque mesmo sem se conhecerem entre si, conseguem comunicar-se. Como deixa de ser necessário um intermediário (um *broker*) para as comunicações – que são, na realidade, instruções – esta adaptação deve traduzir-se numa ligeira melhoria em eficiência.

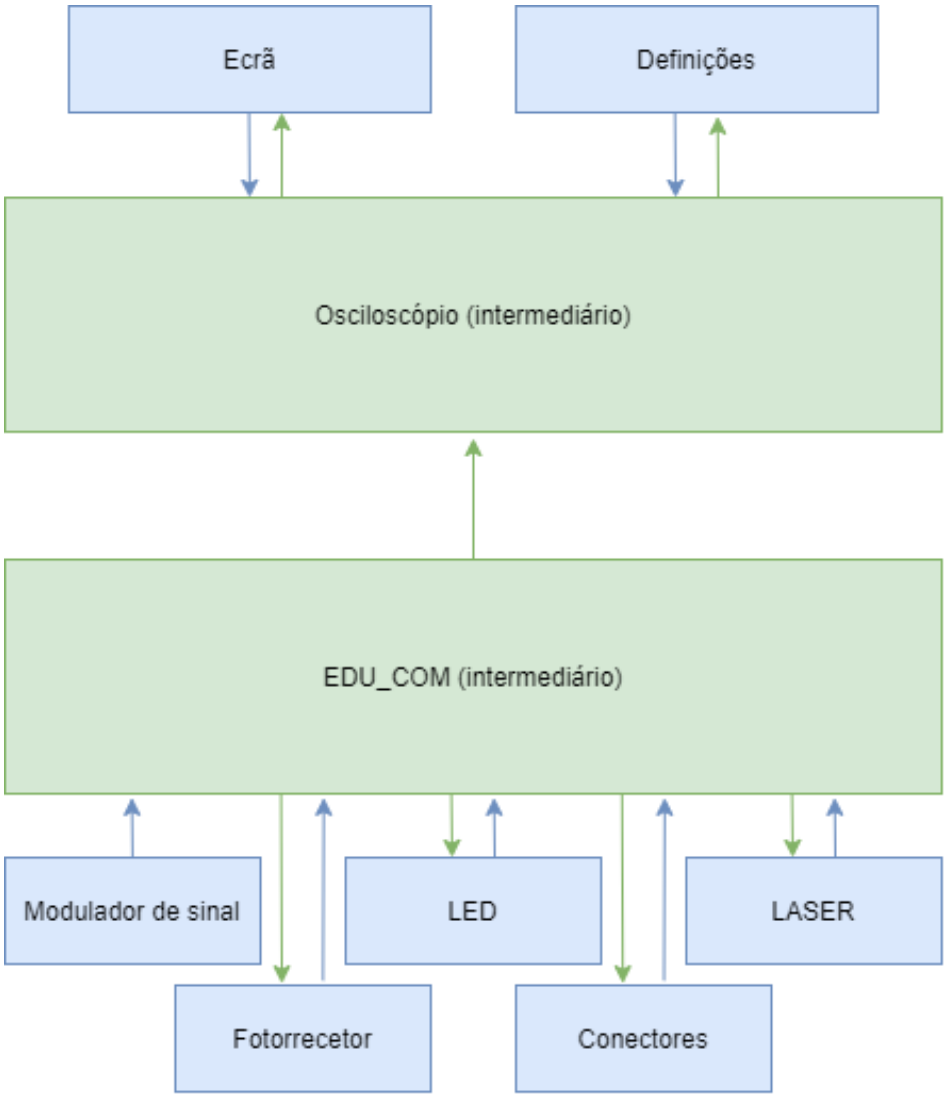

Figura 5. Conceito da arquitetura da aplicação

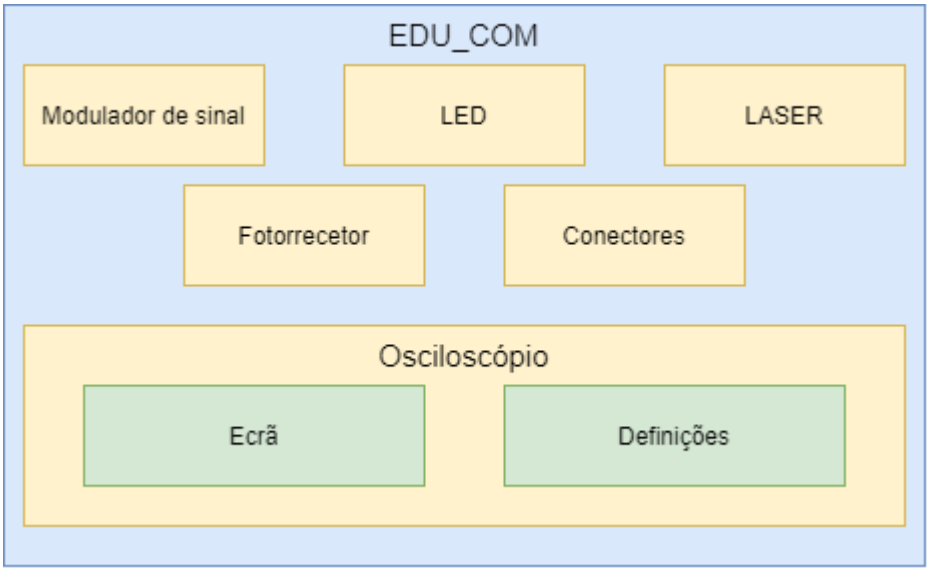

Figura 6. Arquitetura final a implementar

4

Desenvolvimento

4.1 Código

A aplicação será composta por duas vistas, como ilustrado na figura 7. A primeira vista é a do laboratório, sem o osciloscópio. A segunda vista força a sobreposição do osciloscópio de forma a garantir que toda a informação necessária está disponível de forma prática e compacta, permitindo que o aluno se foque apenas no essencial à realização da experiência. A aplicação foi desenvolvida com recurso ao *framework* para interfaces gráficas Swing do Java. A principal referência para o desenvolvimento em Swing foi o curso *online* intitulado de *Java Swing (GUI) Programming: from Beginner to Expert*, de John Purcell, na plataforma Udemy [32]. O curso forneceu uma abordagem prática para a aprendizagem das várias rotinas do Swing e guiou o desenvolvimento de uma aplicação-tipo completamente funcional que validou a utilidade da ferramenta para o projeto do laboratório virtual.

Das características que tornam o Swing adequado para o objetivo, destacam-se:

- Leque vasto de componentes (botões, menus, etc…);
- As componentes são independentes da plataforma (Windows, Linux, iOS);
- As componentes são leves;
- As componentes baseiam-se numa arquitetura *Model View Controller* (MVC), onde o modelo (*Model*) representa os dados que modelam a componente, a vista (*View*) é a apresentação da componente e o controlador (*Controller*) representa a interface entre o modelo e a apresentação da componente;
- A maioria das componentes são extensões da mesma classe abstrata, JComponent (que implementa o MVC), o que lhes confere várias funcionalidades comuns, facilitando o desenvolvimento;
- Possibilidade de criação de componentes de raiz, extendendo também a classe JComponent e herdando um *back-end* comum às restantes componentes;
- *Layouts* altamente flexíveis e customizáveis.

O código será desenvolvido num paradigma de programação orientada a objetos. A programação orientada a objetos assenta em quatro princípios [33]:

1. Encapsulação: o objeto pode manter o seu estado privado. Outros objetos não conseguem aceder a esta informação a menos que esteja de alguma forma disponibilizada por métodos públicos;

2. Abstração: é uma extensão da encapsulação. Refere-se ao processo de seleção de dados de um conjunto maior, de forma a obter apenas os detalhes relevantes para o objeto;

3. Herança: a capacidade de um objeto adquirir propriedades de outro traz vantagens em termos de reusabilidade de código;

4. Polimorfismo: permite a utilização de uma classe da mesma forma que a sua classe mãe, mas com a sua própria implementação dos métodos. Contribui para a abstração.

O diagrama da figura 8 traduz o esquema da figura 7 numa representação da organização dos painéis em classes.

O diagrama é a implementação da arquitetura na figura 6. Repare-se que a classe EDU_COM cria os vários painéis instanciando-os como objetos. Os painéis têm métodos que o EDU_COM pode chamar, pelo que o EDU_COM pode ser o mediador de toda a informação no sistema. O osciloscópio tem, por sua vez, dois painéis próprios e o incremento em complexidade confere-lhe maior autonomia em comparação com os restantes painéis instanciados pelo ED-COM

O código da classe App (App.java) inicializa a aplicação EDU_COM, tal como na figura 9. Esta primeira *thread* invoca o método javax.swing.SwingUtilities.InvokeLater, que agenda a tarefa de criação da GUI e retorna (termina). Esta *thread* apenas agenda a tarefa, porque na GUI quase todo o código que interage com componentes do Swing tem de correr numa *thread* de despacho de eventos [34]. Esta é uma restrição imposta pelo facto de a maioria das componentes do Swing não serem robustas na operação de múltiplas *threads*. Isto é, na ocorrência de eventos que fossem processados em mais que uma *thread*, pode haver interferências entre as *threads* ou erros de inconsistência de memória (as *threads* não são consistentes na forma como vêm os dados). Por esta razão, foi definido que todos os eventos são processados numa mesma *thread* de despacho de eventos [35].

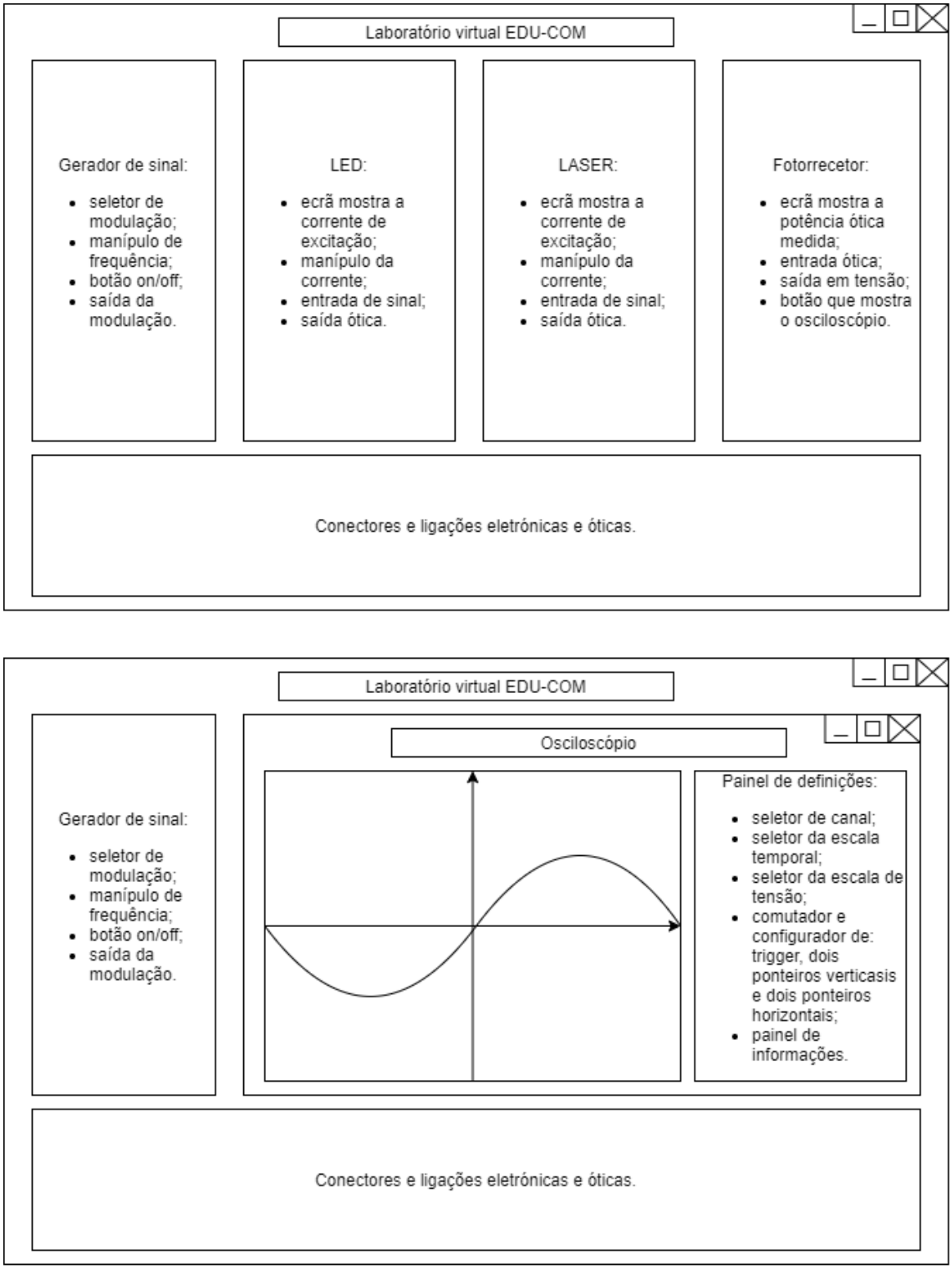

Figura 7. Abstração das telas a desenvolver

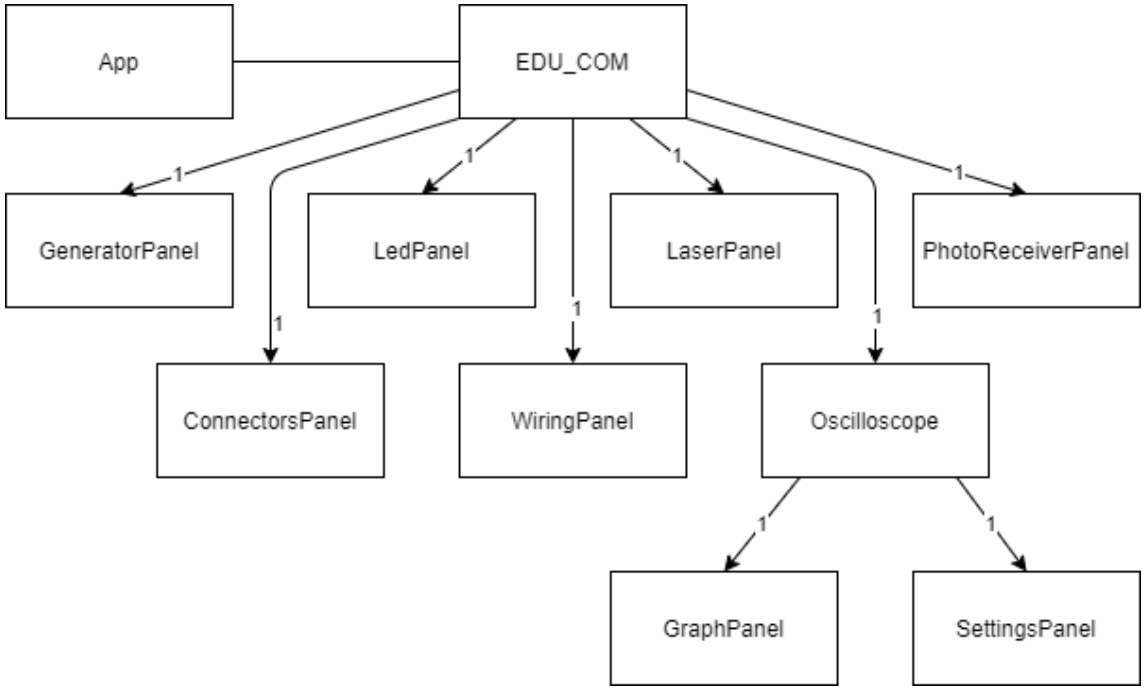

Figura 8. Estruturação básica do código

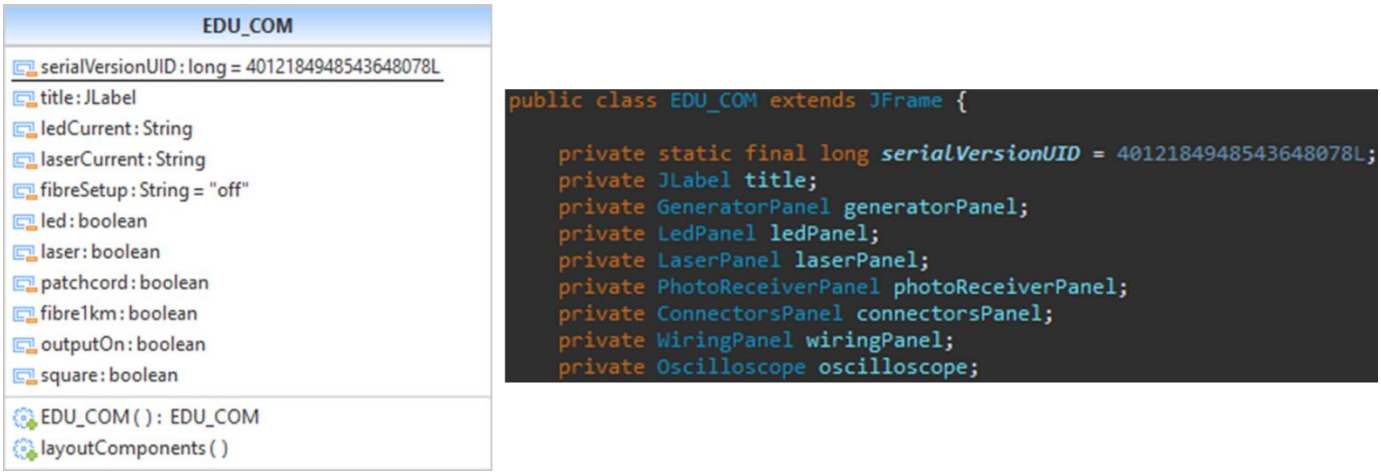

Figura 9. À esquerda, bloco gerado pela ferramenta UML Lab, no Eclipse, com os atributos que reconhece por serem intrínsecos ao JDK – Java Development Kit. À direita, os objetos instanciados de classes extrínsecas, criadas especificamente para o projeto em questão

```
3  public class App {
4
5      public static void main(String[] args) {
6              SwingUtilities.invokeLater(new Runnable() {
7              public void run() {
8                  try {
9                      new EDU_COM();
10                 } catch (Exception e) {
11                     e.printStackTrace();
12                 }
13             }
14         });
15     }
16 }
```

Figura 10. Classe App

A classe *EDU_COM*, na figura 9, contém o código para a vista principal do laboratório virtual, na janela superior da figura 7. Há variáveis que estão definidas como atributos, apesar de as outras classes serem praticamente independentes dos dados nesses registos. Estas variáveis são declaradas fora do construtor *EDU_COM(),* porque são instanciadas e acedidas em métodos anónimos que implementam os processadores de eventos, como é exemplificado com o processador dos eventos ocorridos no painel do gerador de sinal – *GeneratorPanel* – na figura 11. Vale a pena analisar este exemplo com algum detalhe, porque todos os painéis foram construídos de forma análoga.

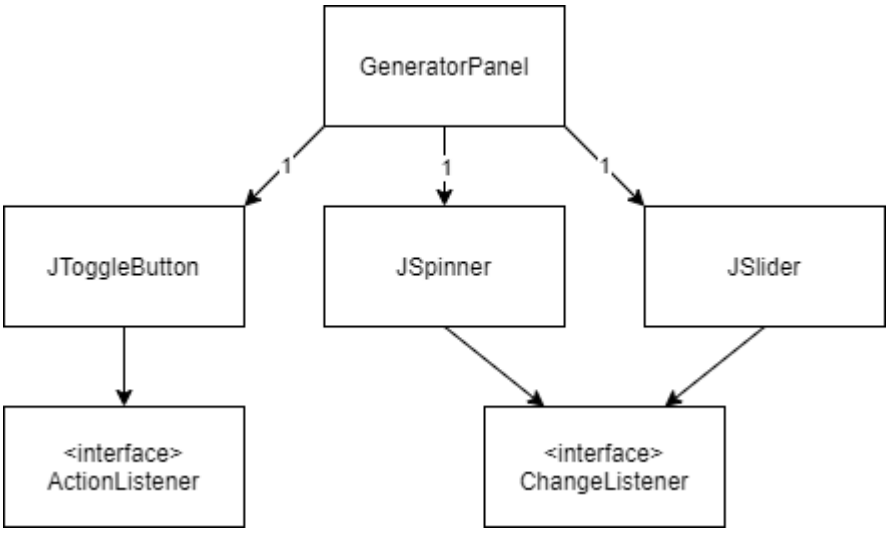

Figura 11. Componentes do GeneratorPanel

No *GeneratorPanel* o objeto da classe *JToggleButton* é responsável por ligar e desligar a simulação. Quando está desligado, a corrente de excitação das fontes de luz será direta. Este objeto implementa a interface *ActionListener*. Uma interface é uma das construções mais importantes no paradigma da programação orientada a objetos, porque alcança um dos conceitos fundamentais, que é a abstração. Uma interface é uma classe completamente abstrata, que agrupa um conjunto de métodos, sem corpo. Por exemplo, a interface MouseListener contem os métodos *MouseClicked(MouseEvent e), MouseEntered(MouseEvent e), MouseExited(MouseEvent e), MousePressed(MouseEvent e)* e *MouseReleased(MouseEvent e)*, onde *MouseEvent e* é o argumento passado para os métodos. Cada método é invocado em situações diferentes, como clicando no rato, libertando o botão do rato, entre outras. O que a interface *MouseListener* permite é a implementação diferenciada desses métodos por outras classes, com a palavra-chave *implements* – figura 12. Classes diferentes podem implementar a mesma interface de forma diferente, porque a interface para a interação com o rato, no caso, é abstraída.

```
public class WiringPanel extends JPanel implements MouseListener, MouseMotionListener {
```

Figura 12. Exemplo de classe que extende a superclasse JPanel e implementa MouseListener e MouseMotionListener

No caso do *GeneratorPanel*, o processador de eventos *ParametersListener* fica à escuta de alterações nos parâmetros definidos pelo utilizador para o sinal de entrada do sistema ótico: o tipo de modulação, a frequência de modulação e a corrente de excitação. Esses valores são alterados com recurso à manipulação de componentes gráficas, como o *JSlider*, que é um manípulo deslizante. Quando o *EDU_COM* instancia o *generatorPanel*, este último instancia todas as componentes gráficas e atribui-lhes, quando necessário um processador de eventos. Com as componentes gráficas pode haver vários tipos de interações. Uma interação possível com todas elas é o simples sobrepor do rato com a componente, mas nem sempre essa interação é relevante. Com o *JSlider*, por exemplo, a interação relevante é arrastar o manípulo para alterar a frequência. É então implementado um processador de eventos – figura 13 – que deteta a nova posição do manípulo e cria, por sua vez, um novo evento, do tipo *ParametersEvent*. Repare-se que uma implementação mais eficiente podia ter sido feita, como boa prática. No código vê-se a criação do evento do tipo *ParametersEvent* sempre que há um evento do tipo *ChangeEvent* no *frequencySlider* (que é uma instanciação do *JSlider*). Seria mais eficiente criar o evento ParametersEvent apenas no caso em que o *ChangeEvent* é relevante, ou seja, quando há o arrasto do manípulo da frequência. No excerto de código apresentado a implementação não está otimizada, uma vez que não compromete os requisitos do sistema.

```
frequencySlider.addChangeListener(new ChangeListener() {
    public void stateChanged(ChangeEvent e) {
        int freq = ((JSlider)e.getSource()).getValue()*1000000;
        frequencyValueLabel.setText(String.valueOf(freq/1000000)+" MHz");
        ParametersEvent ev = new ParametersEvent(this, freq, "frequency");
        if(parametersListener != null) {
            parametersListener.parametersEventOccurred(ev);
        }
    }
});
```

Figura 13. Implementação da interface ChangeListener() pelo manípulo deslizante de frequência, por meio da criação de método anónimo

Esse evento é então detetado no *EDU_COM* por um outro processador de eventos, o *ParametersListener*, que atua sobre todos os eventos a ocorrer no *GeneratorPanel* (eventos do tipo *ParametersEvent*).

```
// generatorPanel setup
generatorPanel.setParametersListener(new ParametersListener() {
    public void parametersEventOccurred(ParametersEvent e) {
        if(e.getChangedVariable().contentEquals("waveform")) {
            square = e.isSquare();
            if(oscilloscope!=null) {
                oscilloscope.getGraphPanel().setWaveform(square);
            }
        }
        else if(e.getChangedVariable().contentEquals("frequency")) {
            oscilloscope.getGraphPanel().setFreq(e.getFrequency());
            oscilloscope.getSettingsPanel().setF(e.getFrequency());
        }
        else if(e.getChangedVariable().contentEquals("modulation")) {
            oscilloscope.getGraphPanel().setModulated(e.getModulation());
            //System.out.println(e.getModulation());
        }
    }
});
```

Figura 14. Implementação da interface ParametersListener, para o painel do gerador de ondas

Para cada interação com o sistema há, portanto, dois processadores de eventos a ser utilizados. Há um primeiro mais próximo da componente gráfica (*front-end*), que está do lado dos painéis (GeneratorPanel, LEDPanel, etc…) e um segundo dedicado à distribuição da informação pelo sistema. O EDU_COM é o mediador dessa informação entre todos os painéis entre si e o osciloscópio. É certo que o osciloscópio é instanciado pelo EDU_COM, mas é mais complexo que os restantes painéis, porque ele próprio instancia os seus painéis (SettingsPanel e GraphPanel) e funciona como mediador da informação entre eles. Tem o mesmo papel que o EDU_COM, mas para efeitos diferentes.

```
waveformSpinner.addChangeListener(new ChangeListener() {
    public void stateChanged(ChangeEvent e) {
        JSpinner spinner = (JSpinner) e.getSource();
        String waveform = (String)spinner.getValue();
        ParametersEvent ev;
        if(waveform.contentEquals("square wave")) {
            ev = new ParametersEvent(this, true, "waveform");
            frequencySlider.setValue(4);
            frequencySlider.setEnabled(false);
        }
        else {
            ev = new ParametersEvent(this, false, "waveform");
            frequencySlider.setEnabled(true);
        }

        if(parametersListener != null) {
            parametersListener.parametersEventOccurred(ev);
        }
    }
});
```

Figura 15. Implementação da interface ChangeListener para o selecionador de modulação

O objeto *waveformSpinner* é uma instanciação da classe *JSpinner* e, tal como o *JSlider*, implementa a interface *ChangeListener* – figura 15. Repare-se que a abstração proporcionada pelas interfaces facilita a implementação de diferentes funcionalidades para o mesmo tipo de eventos. O *waveformSpinner* é responsável por permitir a definição do tipo de modulação a aplicar ao sinal de excitação. Se a modulação for por onda quadrada, o manípulo da frequência é automaticamente movido para a posição de 4 MHz e bloqueado, como definido nos requisitos. Se a modulação for sinusoidal, o manípulo deve ser movível. É então disparado um evento do tipo *ParametersEvent*, que permite ao *EDU_COM* saber que houve alterações nestes parâmetros e encaminhar a informação para as outras construções, nomeadamente o osciloscópio.

Apesar de serem instanciações de classes diferentes, o *ledPanel* e o *laserPanel* são na realidade muito semelhantes e podiam ter sido implementados como instanciações diferentes da mesma classe, uma vez que acabou por se centralizar a parte de simulação numa classe apenas e não nas classes dedicadas (a simulação do led no *ledPanel* e a do laser no *laserPanel*).

Ambos os painéis instanciam um objeto da classe *JScreen*, que será o mostrador da amplitude da corrente de excitação, em miliampere. Instanciam também um objeto da classe *JKnob*, que é o manípulo rotativo que permite controlar essa amplitude. A classe *JKnob* implementa duas interfaces de interação com o rato, para as suas funcionalidades internas e ainda implementa a interface *PropertyChangeListener*, mas, à semelhança das interfaces *ChangeListener* e *ActionListener* no *generatorPanel*, são o *ledPanel* e o *laserPanel* que definem como o objeto deverá implementar os métodos da interface *PropertyChangeListener*.

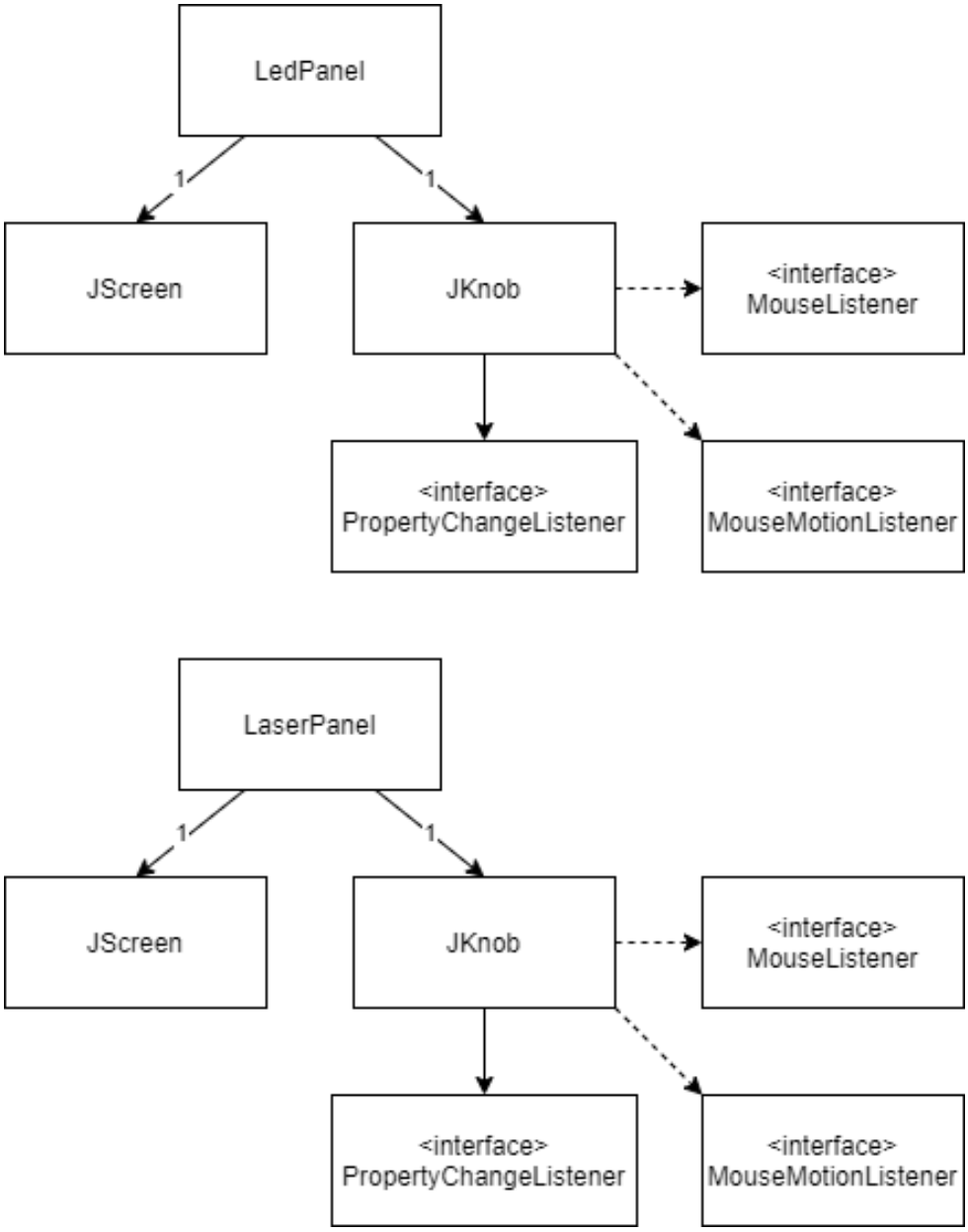

Figura 16. Painéis do LED e do LASER

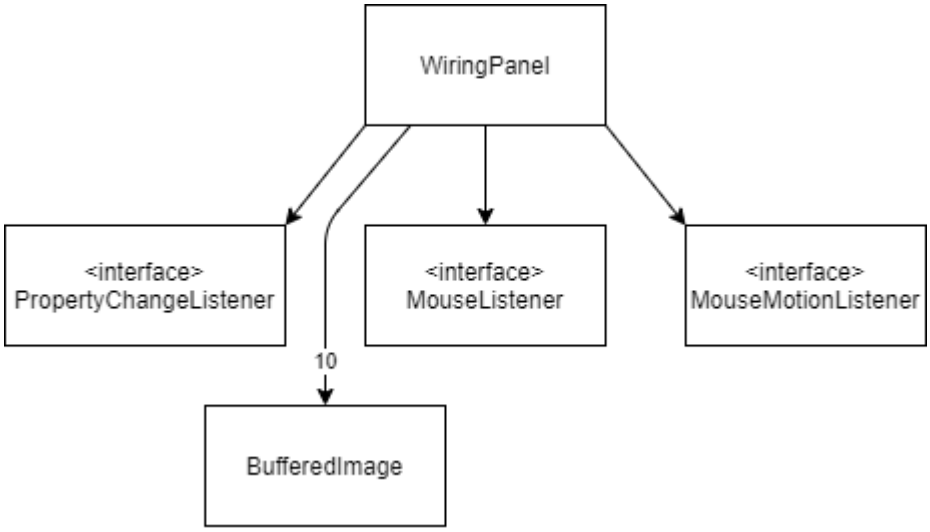

Figura 17. WiringPanel

O *wiringPanel* – figura 17 – é o painel onde se fazem as conexões das fibras óticas e dos cabos de sinal elétrico. Para isso, é necessário criar uma interface de operação dos conectores. São utilizadas fotografias dos conectores, acoplados e desacoplados. Para carregar as fotografias, instanciam-se objetos do tipo *BufferedImage* para cada imagem. O *wiringPanel* implementa três interfaces. Tal como a maioria dos restantes painéis, implementa a *PropertyChangeListener*, que processa os eventos relevantes para o sistema. As interfaces *MouseListener* e *MouseMotionListener* são as relevantes para a operação dos conectores. Existem os conectores I/O e os conectores dos cabos. Começa por colocar-se as imagens dos conectores I/O no painel, com a mesma disposição do equipamento que está a ser simulado. Pressionando o botão principal do rato, recorrendo à interface *MouseListener*, são desenhados o conectores do cabo – cabo selecionado no *connectorsPanel* – na coordenada em que está o rato. Arrastando o rato, recorrendo à interface *MouseMotionListener* e à classe *java.awt.geom.Path2D*, desenham-se curvas de forma a simular o cabo suspenso entre os dois conectores. Para isso, calcula-se o ponto médio entre os dois extremos do cabo e faz-se uma translação desse ponto de forma ortogonal à linha que une os extremos do cabo – são eles o ponto em que se pressionou o rato pela primeira vez e o ponto que está a ser pressionado no momento ou deixou de ser pressionado – de 10% do comprimento dessa linha. Com a classe *java.awt.geom.Path2D* calcula-se uma spline que une esses três pontos – as características da spline não são muito relevantes, desde que seja rapidamente calculada e tenha uma curvatura que represente realisticamente o efeito que se pretende simular. A curva resultante tem apenas um pixel de espessura, pelo que se concatenam dez (para a fibra ótica) ou quinze (para o cabo elétrico) iterações deste

processo, com a cor a incrementar gradualmente os parâmetros RGB pintando o cabo com um gradiente de cinzentos que lhe confere um efeito tridimensional. Quando o rato é pressionado sobre o conector I/O, as imagens desse conector e do conector esquerdo do cabo são substituídas pela imagem dos conectores acoplados. Quando o rato é pressionado e simultaneamente movido sobre um conector I/O compatível com o conector do cabo, as duas imagens também são substituídas pela imagem dos conectores acoplados. Para mudar de cabo, basta selecioná-lo no *conectorsPanel* e repetir o processo. Para reposicionar o cabo, também é necessário repetir o processo.

O desenho dos componentes é feito com recurso a uma API (Application Programming Interface) do AWT (Abstract Window Toolkit) chamada *paint* e é implementado por sobreposição do método *java.awt.Component.paint()* com a implementação pretendida. Além da lógica já descrita no último parágrafo, é ainda necessária a lógica de disparo dos eventos que informam o sistema sempre que há uma alteração na configuração da cablagem, para que o sistema saiba com que parâmetros executar o algoritmo de simulação. Toda essa lógica é implementada no método paint() que é redefinido no wiringPanel.

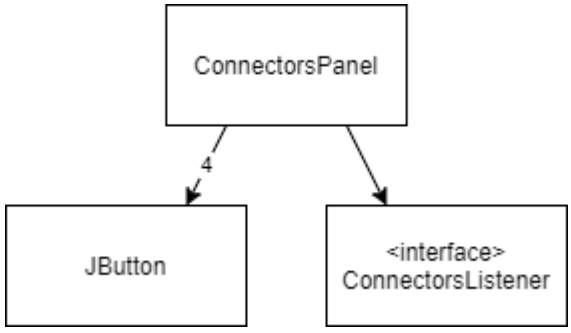

Figura 18. ConnectorsPanel

O *connectorsPanel* tem a função de selecionar o cabo e tipo de conector a utilizar. Oferece a possibilidade de escolher entre o cabo elétrico, a fibra ótica de um metro, a fibra ótica de um quilómetro e a fibra de um metro conectada à fibra de 1 quilómetro. São desenhados os botões comutadores do tipo *JtoggleButton*, com a interface *ActionListener* são detetadas e processadas as interações com esses botões. Quando um botão é selecionado passa ao estado de verdadeiro lógico e aquele que estivesse selecionado previamente passa ao estado de falso lógico. Ainda no processamento desses eventos, são disparados eventos do tipo *ConnectorsEvent*. O conceito é análogo ao do *ParametersEvent*, mas neste caso o *EDU_COM* implementa uma interface diferente, como está visível no excerto de código do *EDU_COM* na figura 15. A razão para a qual foi utilizado outro tipo de evento e não o *ParametersEvent* que

foi utilizado até agora, é que este evento tem um objetivo diferente. A informação no evento do tipo *ParametersEvent* vai influenciar a forma como é vista a simulação, já a informação do evento *ConnectorsEvent* vai apenas influenciar o tipo de cabo e conector que vai ser utilizado na próxima interação com o *wiringPanel*. Ou seja, o *ConnectorsEvent* não tem a palavra final sobre qual o tipo de simulação que vai ocorrer. Ainda é necessário que no *wiringPanel* a conexão tenha sucesso. E aí sim, a simulação será informada.

O *photoReceiverPanel* é onde está alojada a simulação ao nível de potência ótica, que é simplesmente uma recolha de dados laboratoriais com máxima resolução, ou seja, todos os pares de corrente de excitação – potência ótica possíveis de registar com o equipamento físico. Os dados estão organizados em ficheiros .txt que são lidos linha a linha com a otimização da classe *BufferedReader* [36], assim que o painel é instanciado. À medida que a leitura decorre, os dados são escritos e disponibilizados em variáveis próprias do tipo String[], uma vez que a leitura sob demanda é muito mais eficiente numa variável deste tipo do que consultando um ficheiro.

Quando há uma alteração na corrente de excitação (por rotação do manípulo ou reconfiguração da cablagem), o painel *EDU_COM* capta esse evento e chama o método *PhotoReceiverPanel.forceUpdate(String current),* que encontra o valor da potência ótica correspondente e o atualiza no ecrã via o método *Jscreen.setText(String s)*.

```
connectorsPanel.setConnectorsListener(new ConnectorsListener() {
    public void ConnectorsEventOccurred(ConnectorsEvent e) {
        if(e.getChangedVariable().contentEquals("rfConnector")) {
            wiringPanel.setRfConnector(true);
            wiringPanel.setFibre(false);
        }
        else if(e.getChangedVariable().contentEquals("patchCord")) {
            wiringPanel.setRfConnector(false);
            wiringPanel.setFibre(true);
            wiringPanel.setConnectedFibre("patchCord");
            patchcord = true;
            fibre1km = false;
            if(oscilloscope != null) {
                if(outputOn) {
                    if(led) {
                        oscilloscope.getGraphPanel().setFibreSetup("led1m");
                    }
                    if(laser) {
                        oscilloscope.getGraphPanel().setFibreSetup("laser1m");
                    }
                }
            }
        }
        else if(e.getChangedVariable().contentEquals("fibre1km")) {
            wiringPanel.setRfConnector(false);
            wiringPanel.setFibre(true);
            wiringPanel.setConnectedFibre("fibre1km");
            fibre1km = true;
            patchcord = false;
            if(oscilloscope != null) {
                if(outputOn) {
                    if(led) {
                        oscilloscope.getGraphPanel().setFibreSetup("led1km");
                    }
                    if(laser) {
                        oscilloscope.getGraphPanel().setFibreSetup("laser1km");
                    }
                }
            }
        }
        else if(e.getChangedVariable().contentEquals("patchCord1km")) {
            wiringPanel.setRfConnector(false);
            wiringPanel.setFibre(true);
            wiringPanel.setConnectedFibre("fibre1001m");
        }
        else if(e.getChangedVariable().contentEquals("fibre2km")) {
            wiringPanel.setRfConnector(false);
            wiringPanel.setFibre(true);
            wiringPanel.setConnectedFibre("fibre2km");
        }
        else if(e.getChangedVariable().contentEquals("patchCord2km")) {
            wiringPanel.setRfConnector(false);
            wiringPanel.setFibre(true);
            wiringPanel.setConnectedFibre("fibre2001m");
        }
    }
});
```

Figura 19. Implementação da interface do ConnectorsPanel

O painel *photoReceiverPanel* tem ainda um botão do tipo *JButton* que torna visível a janela do osciloscópio. O botão foi escolhido para o propósito que serviriam as saídas em tensão do *EDU_COM* e as entradas do osciloscópio e fornece uma interface mais amigável para o utilizador.

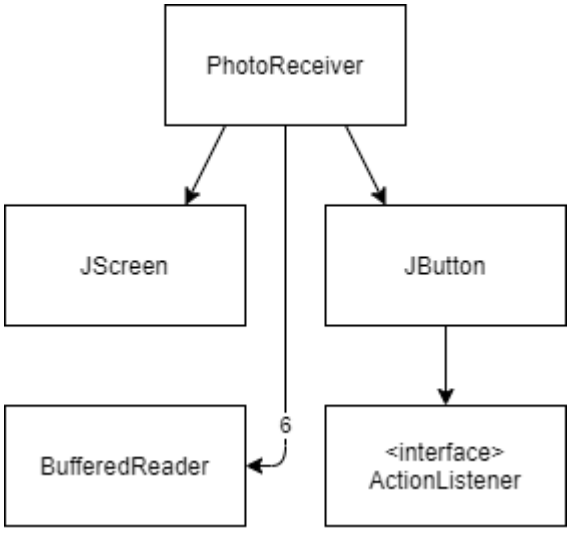

Figura 20. Classe do fotodetetor: PhotoReceiverPanel

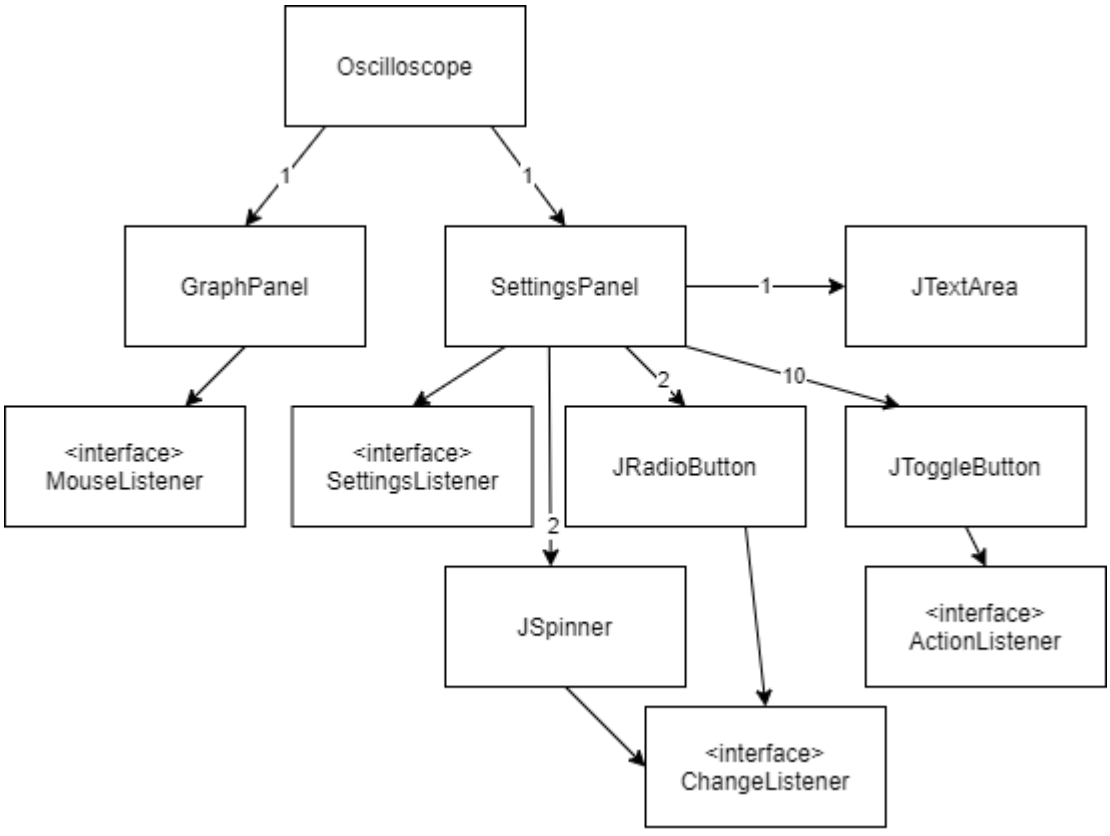

Figura 21. Classe do osciloscópio: Oscilloscope

O osciloscópio é inicializado no *EDU_COM* na mesma etapa que todos os outros painéis. Enquanto não está visível, por não ter sido ainda clicado o botão *Oscilloscope*

no painel *photoReceiverPanel*, continua a ser informado pelo *EDU_COM* de todas as configurações relevantes para o seu subsistema, sem que desempenhe as tarefas que consomem mais recursos, que são as gráficas. É composto por dois painéis. O oscilloscope cria uma thread específica que atualiza os gráficos e textos do osciloscópio a cada 10 milissegundos, quando o osciloscópio está visível. Nesta nova janela é ainda criado um painel de definições à direita, onde se pode configurar a escala temporal e as escalas de tensão para os dois canais, bem como os ponteiros verticais e horizontais e o *trigger*. A interface para as escalas utiliza duas instâncias da classe *JRadioButton* – para escolher entre os canais 1 e 2 – bem como dois objetos do tipo *JSpinner*, para alterar os valores da escala respetiva. A interface para os ponteiros e o *trigger* é construída com um sistema de vários *JPanel* dedicados e botões do tipo *JToggleButton*. Quando o botão "Change" está ativo, a interface *MouseListener* no painel dos gráficos fica à escuta de cliques do rato na sua área, e definirá a nova abcissa ou ordenada do ponteiro respetivo.

Os eventos que ocorrem no painel de definições são do tipo *SettingsEvent* e os eventos que ocorrem no painel dos gráficos são do tipo *PropertyChangeEvent.* A classe *PropertyChangeEvent* é intrínseca ao JDK e não é necessário implementá-la. Na figura 22 é possível notar quais os atributos do evento *SettingsEvent* e a forma como se chama o seu construtor. O evento é disparado pela chamada do seu construtor, com o valor da variável formatado num inteiro e o nome da variável numa *String*. A interface *SettingsListener* permite ao oscilloscope aceder aos métodos públicos *getters* do evento que foram implementados para cada variável e encaminhar as informações para o *graphPanel* que, a cada dez milissegundos, atualizará o gráfico.

```
import java.util.EventObject;

public class SettingsEvent extends EventObject {

    private static final long serialVersionUID = -6558986381093456818L;
    private String changedVariable;
    private int channel;
    private int timeScale;
    private int voltageScale;
    private int triggerEn;
    private int triggerCh;
    private int pointerV1En;
    private int pointerV1Ch;
    private int pointerV2En;
    private int pointerV2Ch;
    private int pointerH1En;
    private int pointerH1Ch;
    private int pointerH2En;
    private int pointerH2Ch;

    public SettingsEvent(Object source, int value, String variable) {
        super(source);
```

Figura 22. SettingsEvent

A arquitetura da classe e subsistema do *Oscilloscope* é análoga à do *EDU_COM,* adaptada à realidade de operação com uma thread dedicada ao gráfico. Ao contrário do *EDU_COM,* o oscilloscope apenas encaminha as informações para o *graphPanel* e é o próprio painel quem processa a informação periodicamente. O *EDU_COM*, regra geral, comunica instruções, mais do que dados. As informações relativas às características do sinal relevantes para a experiência serão escritas numa componente do tipo *JTextArea*, no *SettingsPanel.*

Em Java não existe uma função do estilo *plot()* como em MATLAB ou Python. Por isso é necessário implementar esse método de raiz, que consistirá num algoritmo de pintura pixel por pixel das curvas dos sinais detetados no osciloscópio

A simulação gráfica ocorre no GraphPanel e será discutida no próximo capítulo.

4.2 Simulação

No desenvolvimento do laboratório virtual, a simulação é um ponto fulcral do projeto. Deve ser profunda na quantidade certa para o propósito pedagógico e ao mesmo tempo leve, no que toca a necessidade de processamento, não comprometendo os requisitos de portabilidade e compatibilidade da aplicação. Não sendo à partida evidente qual a abordagem da simulação mais adequada ao seu propósito, foi utilizado um método iterativo até alcançar a abordagem pretendida.

O gráfico do osciloscópio tem uma taxa de atualização de 100 Hz, pelo que a simulação deve consistentemente levar menos de 10 milissegundos a correr, já integrada em todo o sistema, que também tem as suas necessidades de processamento. Se for necessário, pode baixar-se a taxa de atualização do osciloscópio para os 50 Hz, por exemplo. Dadas as experiências que o aluno deve conseguir realizar com recurso ao laboratório, a simulação deve compreender três fatores em especial:

- Potências de entrada e de saída do sistema nas suas várias configurações;
- Desfasamento temporal entre sinal de entrada e sinal de saída;
- Tempos de subida na modulação digital.

Como etapa zero no processo de construção da simulação, para testar o ambiente, fez-se uma simulação fotão a fotão. Claro que é uma simulação destinada ao fracasso, dada a impossibilidade de simular individualmente o comportamento de um número tão elevado de partículas. Repare-se que 500 μW de potência ótica de uma fonte perfeitamente coerente nos 785 nm de comprimento de onda são quase 2×10^{15} fotões num segundo, pela relação de Planck, na equação 1, com E a energia do fotão, h a constante de Planck, c a velocidade da luz no vácuo e λ o comprimento de onda:

$$E = h\frac{c}{\lambda} \tag{1}$$

O processador AMD Ryzen 9 5900X custa tanto como muitos computadores completamente funcionais e tem um relógio que chega aos 4.8 GHz. Ou seja, se simular o percurso completo de um fotão fosse tão simples como fazer uma soma, este CPU levaria praticamente 6 minutos e 57 segundos a simular um fenómeno que dura 1 segundo. Esta simulação não produziu resultados.

A primeira simulação levada a cabo, consistiu no agrupamento de fotões em comprimentos de onda discretos, em que a distribuição dos comprimentos de onda é feita num histograma com cada vez menos colunas, até se encontrar o menor compromisso de resolução espetral pela viabilidade temporal da simulação. É certo que há uma perda em resolução, mas a amostragem preserva os parâmetros das distribuições espetrais das duas fontes luminosas – o laser segue uma distribuição Lorentziana e o led segue uma distribuição gaussiana [37].

As figuras 23 e 24 mostram as distribuições cumulativas das abundâncias relativas expectáveis para o LASER e o LED, respetivamente, de acordo com as equações 2 e

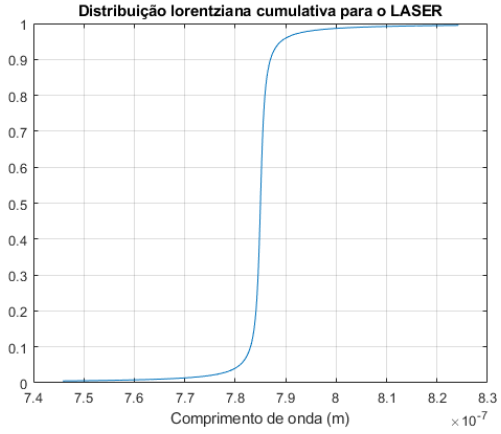

Figura 23. Lorentziana cumulativa por comprimento de onda

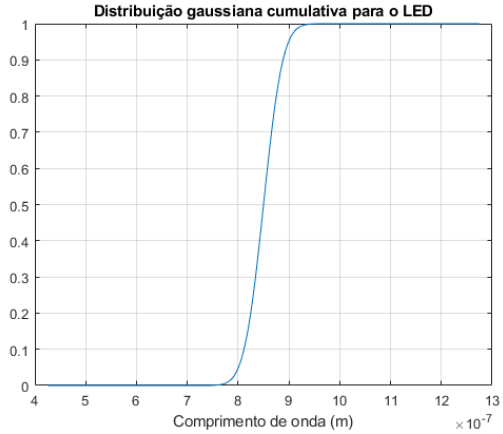

Figura 24. Gaussiana cumulativa por comprimento de onda

3. As amostragens são feitas para 1001 pontos, com comprimentos de onda compreendidos no intervalo $[\bar{\lambda} - \left(\frac{\#pontos}{2} - 1\right) \times \frac{s\bar{\lambda}}{\#pontos}; \bar{\lambda} + \left(\frac{\#pontos}{2} + 1\right) \times \frac{s\bar{\lambda}}{\#pontos}]$ em passos de $\frac{s\bar{\lambda}}{\#pontos}$, onde $\bar{\lambda}$ é o comprimento de onda médio da respetiva fonte de luz (785nm para o LASER e 850nm para o LED), e s é o desvio relativo máximo, em relação a $\bar{\lambda}$.

$$L_{\bar{\lambda},FWHM}(\lambda) = 0.5 + \frac{1}{\pi}\arctan\frac{\lambda - \bar{\lambda}}{FWHM} \tag{2}$$

$$L_{\bar{\lambda},rms}(\lambda) = 0.5 + \frac{1}{\sqrt{\pi}}\int_0^{\frac{\lambda-\bar{\lambda}}{rms\sqrt{2}}} e^{-t^2}dt \tag{3}$$

As figuras 21 e 22 mostram a distribuição resultante em potência, recorrendo à relação de Planck (equação 1) para determinar a contribuição em energia por cada comprimento de onda e refletindo uma potência integral, para o LASER, de 500 μW e para o LED de 90 μW, que são as potências óticas típicas de saída medidas para as respetivas correntes de polarização.

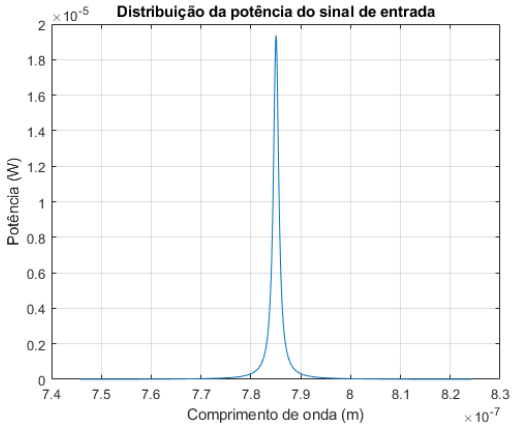

Figura 25. Distribuição de potência do LASER por comprimento de onda

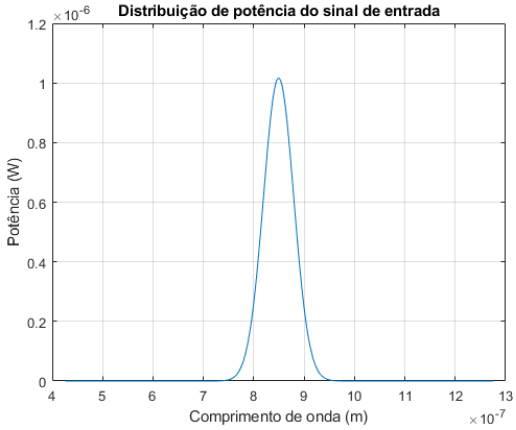

Figura 26. Distribuição da potência do LED por comprimento de onda

As consequências de a luz produzida por estas fontes luminosas não ser perfeitamente monocromática são um dos objetos de estudo que o laboratório virtual deve proporcionar. A velocidade de propagação da luz no núcleo da fibra ótica depende do seu comprimento de onda. Em comunicações por fibra ótica, em que as ligações podem ter vários quilómetros de comprimento, o espetro de velocidades resulta numa dispersão temporal do sinal, denominada dispersão cromática, que é um fator determinante na especificação de redes de comunicação. Wolfgang von Sellmeier formulou uma equação para o cálculo do índice de refração em função do comprimento de onda de materiais óticos transparentes, naquela a que chamamos a fórmula de Sellmeier [38]:

$$n(\lambda) = \sqrt{1 + \sum_j \frac{A_j \lambda^2}{\lambda^2 - B_j^2}} \tag{4}$$

A_j e B_j são coeficientes e são geralmente obtidos por um método de mínimos quadrados. Este modelo é especialmente preciso na região do infra-vermelho próximo. A equação de Sellmeier para a sílica fundida, com os comprimentos de onda em micrómetros, é:

$$n(\lambda) = \sqrt{1 + \frac{0.6961663\,\lambda^2}{\lambda^2 - 0.0684043^2} + \frac{0.4079426\lambda^2}{\lambda^2 - 0.1162414^2} + \frac{0.8974794\lambda^2}{\lambda^2 - 9.896161^2}} \quad (5)$$

Num laboratório físico real, mede-se a dispersão temporal pela medição do tempo de subida do sinal de saída em modulação digital, que é detetável num osciloscópio com o *trigger* configurado. Na simulação virtual que está a ser construída, não há a dificuldade em detetar um pulso único, pelo que não é necessário implementar a modulação digital para testar a dispersão temporal. Para isso, simula-se a dispersão de um pulso instantâneo de 500 µJ para o LASER e de 90 µJ para o LED ao longo de uma fibra com 1 km de comprimento, modelada pela fórmula de Sellmeier. Para a simulação, é utilizada a equação 5, uma vez que a sílica fundida é um material muito comum em núcleos de fibras óticas. Os resultados das simulações são apresentados nas figuras 27 e 28.

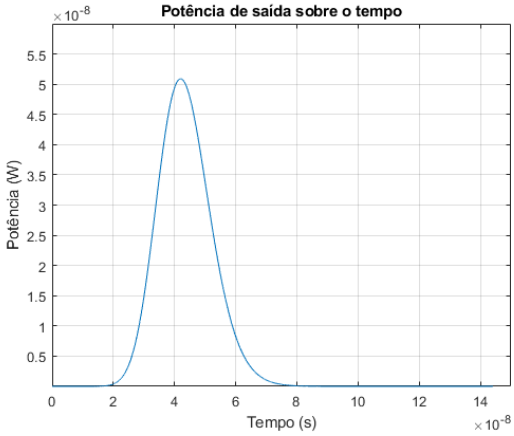

Figura 27. Dispersão temporal de um pulso LED

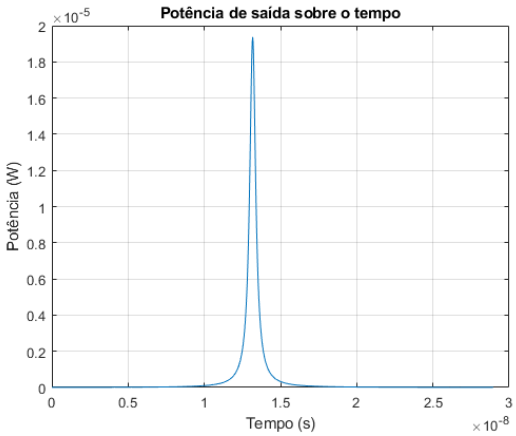

Figura 28. Dispersão temporal de um pulso LASER

Desprezando a parte dos sinais abaixo de 2.5% da altura da curva respetiva, o sinal LASER dispersa-se sobre 2.93 ns e o sinal LED tem uma largura de 46 ns. Estes valores são apenas relativos à dispersão cromática. De acordo com o Manual do Instrutor do kit da OptoSci, valores do tempo de subida na fibra são de 7,4 ns para o LASER e de 17,9 ns para o LED. No manual, é considerado que a dispersão cromática para o laser é desprezável, pelo que o tempo de subida apenas depende de dispersão modal. No LED, existem as duas contribuições, que sendo estatisticamente independentes são somadas em quadratura. Pelo método de simulação empregado, que ignora a dispersão modal, os tempos de subida simulados são cerca de metade da largura de cada pulso, o que está longe dos valores do manual. A avaliação quantitativa tem pouco significado, porque a simulação não foi feita para replicar com exatidão os

valores do sistema físico. Isso seria praticamente impossível sem uma caracterização mais detalhada do sistema. O objetivo, recorde-se, é simular um sistema meramente semelhante ao físico, que replique os efeitos relevantes para as atividades laboratoriais em ordens de grandeza coerentes. Desta forma, a avaliação qualitativa é satisfatória, no que toca aos valores obtidos.

No que toca à eficiência da simulação, os requisitos não são cumpridos. A versão mais económica da simulação para o laser, com apenas 11 pontos calculados por pulso e com um desvio máximo relativamente à frequência central de 1% (ver figura 29; note-se que apresenta uma dispersão temporal ainda mais afastada da expectável que a obtida com a simulação na figura 27) demorou em média, para 570 iterações, 569±134 µs. Foram feitas 570 iterações, porque para o gráfico do osciloscópio do laboratório virtual tem 570 pontos, pelo que para cada *frame* é necessário simular no mínimo 570 pulsos. De acordo com a média obtida, a simulação de cada *frame* demoraria em média 0.324±0.077 segundos, ou seja, uma taxa de atualização dos gráficos máxima de 3 Hz, muito longe dos 100 Hz pretendidos ou dos 50 Hz admitidos.

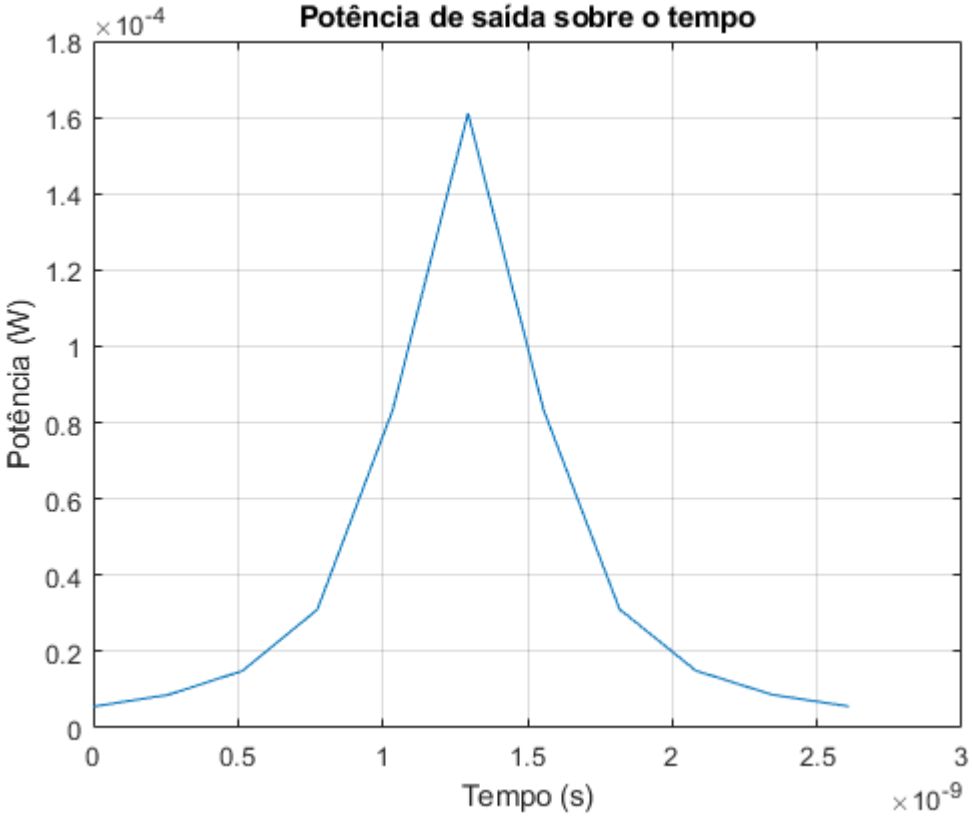

Figura 29 - LASER

Na impossibilidade de utilizar esta abordagem de simulação, investigou-se sobre abordagens alternativas, que passassem pela simulação de sistemas preferencialmente equivalentes em potência e resposta em frequência.

O artigo de M. S. Ozyazici descreve a teoria e experimentação na caracterização da equivalência elétrica de um díodo laser comercial, o Hitachi HLP-1400. O circuito equivalente resultante toma em consideração perturbações parasíticas e proporciona uma simulação eficaz da resposta a modulação do LASER, com o objetivo de otimizar os parâmetros de modulação a aplicar em sistemas de comunicação por longas distâncias. Para o projeto da simulação para o laboratório virtual, não é necessária uma análise com a profundidade e do trabalho de M. S. Ozyazici. É apenas analisado o circuito equivalente às dinâmicas intrínsecas do díodo LASER, sem as perturbações da integração do díodo num sistema de comunicação [39].

Para obter o modelo elétrico do LASER, é feita a relação (equação 8) entre a densidade de portadores no LASER e a tensão na junção. As equações de taxa de acoplamento mono-modo (equações 6 e 7) apresentadas no artigo partem da assunção – que é coerente com o LASER que se quer simular para o laboratório virtual – de que o ganho do LASER é linear a partir do momento em que a densidade eletrónica supera a população de inversão – a população de inversão é o limiar de densidade eletrónica no meio LASER para se obter um ganho positivo.

$$\frac{dN_e}{dt} = \frac{1}{qad} - A(N_e - N_{om})N_{ph} - \frac{N_e}{\tau_s} \tag{6}$$

$$\frac{dN_{ph}}{dt} = A(N_e - N_{om})N_{ph} - \frac{N_{ph}}{\tau_{ph}} - \beta\frac{N_e}{\tau_s} \tag{7}$$

$$N_e = N_i \exp\frac{qV}{2kT} \tag{8}$$

N_e é a densidade eletrónica na cavidade LASER, N_{ph} é a densidade fotónica no mesmo espaço, q é a carga elementar, d é a espessura do meio LASER, a é a área da faixa de contacto do díodo, N_{om} é a densidade eletrónica de inversão de população, A é uma constante relacionada com o processo de emissão estimulada, τ_s é o tempo de

vida para a emissão estimulada, τ_{ph} é o tempo de vida dos fotões na cavidade ótica, β é a porção de emissão espontânea que é acoplada ao modo laser e N_i é a densidade intrínseca de portadores de carga no dispositivo AlGaAs.

Tucker apresenta um método de resolução das equações (6) e (7) para sinais pequenos com introdução da equação (8) [40][41]. A impedância resultante é a mesma do circuito RLC paralelo da figura 30, que é o circuito elétrico equivalente intrínseco do LASER.

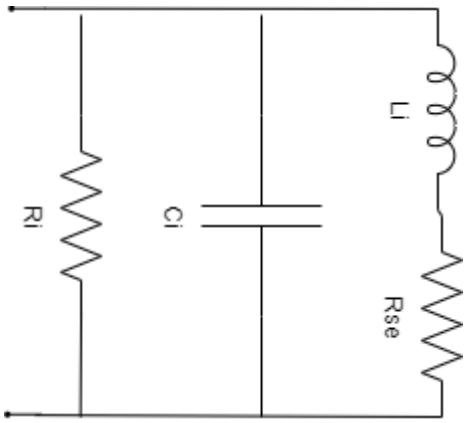

Figura 30. Circuito equivalente intrínseco do LASER

Sendo:

$$R_i = {R_d}\Big/{(n_{ph}^0 + 1)} \tag{9}$$

$$L_i = {R_d \tau_{ph}}\Big/{[(n_{ph}^0 + \beta)(n_e^0 - n_{om})]} \approx {R_d \tau_{ph}}\Big/{n_{ph}^0} \tag{10}$$

$$C_i = \tau_s / R_d \tag{11}$$

$$R_{se} = \beta R_d \frac{n_e^0}{n_{ph}^0 (n_{ph}^0 + \beta)(n_e^0 - n_{om})} \approx \beta R_d \frac{n_e^0}{(n_{ph}^0)^2} \tag{12}$$

$$R_d = \frac{2kT}{q}\frac{1}{I_d} \tag{13}$$

Onde n_{ph}^0, n_e^0 e n_{om} são os valores estáveis normalizados, respetivamente, de N_{ph}, N_e e N_{om}, R_d é a resistência diferencial do díodo, R_i e R_{se} modelam o

amortecimento da ressonância optoeletrónica, a capacidade C_i representa a capacidade criada pela difusão de portadores na camada ativa do LASER e a indutância L_i, juntamente com a capacidade C_i, modelam a ressonância e resposta em frequência do LASER [39].

O estudo de Oziazici proporciona uma forma simples de simular a resposta a modulação do LASER, mas obriga a um trabalho de caracterização do LASER.

O artigo de André et al [42] apresenta uma técnica de extração de parâmetros do LASER para as suas equações de taxa, para utilização em simulações de sistemas de comunicação ótica de alta velocidade. Notável, mas não relevante para a simulação a construir para o laboratório virtual, é que a técnica apresentada é imune aos efeitos parasitas na estimação dos parâmetros, por fazer uma razão entre duas funções de transferência para correntes de operação (*polarização*) diferentes.

Essa extração de parâmetros é feita pela modelação da resposta em frequência para diferentes correntes de operação. A modelação é feita com recurso a um algoritmo de Marquardt-Levenberg e otimizada por um método quasinewtoniano e um método de diferenças finita, aplicados a uma primeira escolha de parâmetros fisicamente razoáveis. A resposta em frequência é medida num intervalo entre os 0,04 e os 12,50 GHz, intervalo esse inalcançável pelo equipamento modelo do laboratório virtual.

4.2.1 Solução

Na impossibilidade de implementar qualquer um destes métodos de simulação, a solução encontrada foi a replicação no laboratório virtual de um conjunto de dados obtidos experimentalmente.

Para proporcionar variabilidade, é necessário simular o erro de medição. O erro pode ser modelado por uma distribuição gaussiana. Para a amostragem de distribuições normais pode usar-se um método simples de Monte Carlo. A partir do gerador de números pseudo-aleatórios uniformemente distribuídos da JVM, são gerados dois números aleatórios. É calculada a probabilidade de ocorrência do primeiro número numa distribuição normal de desvio padrão medido experimentalmente para a grandeza em questão – seja a potência ótica ou a tensão medida no osciloscópio. Se o segundo número aleatório for igual ou inferior a essa probabilidade, o resultado é aceite e é o desvio apresentado no laboratório. Senão, o processo é repetido. O valor do primeiro número aleatório pode ser limitado, por exemplo, entre ±3σ, cobrindo 99,7% da área da gaussiana.

Este método é eficiente mesmo numa larga escala de medições, mas é possível que numa simulação tão extensa de pontos como a que usamos – 57 000 pontos por segundo – sejam frequentes as iterações que acabam por demorar mais tempo,

gerando instabilidade na frequência de desenho dos gráficos. O método escolhido consiste na utilização de uma distribuição uniforme que preserve o mesmo desvio padrão, que é a distribuição uniforme no intervalo entre ±2rms.

Nas figuras 31, 32, 33, 34, 35 e 36 são apresentados os gráficos da potência ótica em função da corrente de excitação das fontes luminosas, sem modulação.

Os valores médios para o tempo de subida na modulação digital são discriminados na tabela 3:

Tabela 3. Tempos de subida em modulação digital

	LED – 1 m	**LED – 1 km**	**LASER – 1 m**	**LASER – 1 km**
Tempo de subida (ns)	9.1±0.1	12.8±0.2	8.0±0.1	10.9±0.2

O ruído de fundo medido é de 20 mVpp ≈ 7 mV rms, a amplitude do sinal medida pela fibra de 1 metro é de 1264 ± 5 mV para o LED e de 4854 ± 23 mV para o LASER. O sinal detetado após propagação pela fibra de 1 km é de 607 ± 3 mV para o LED e de 1620 ± 9 mV para o LASER.

Esta recolha de dados é disponibilizada nos ficheiros que o programa lê e armazena em variáveis para proceder à simulação.

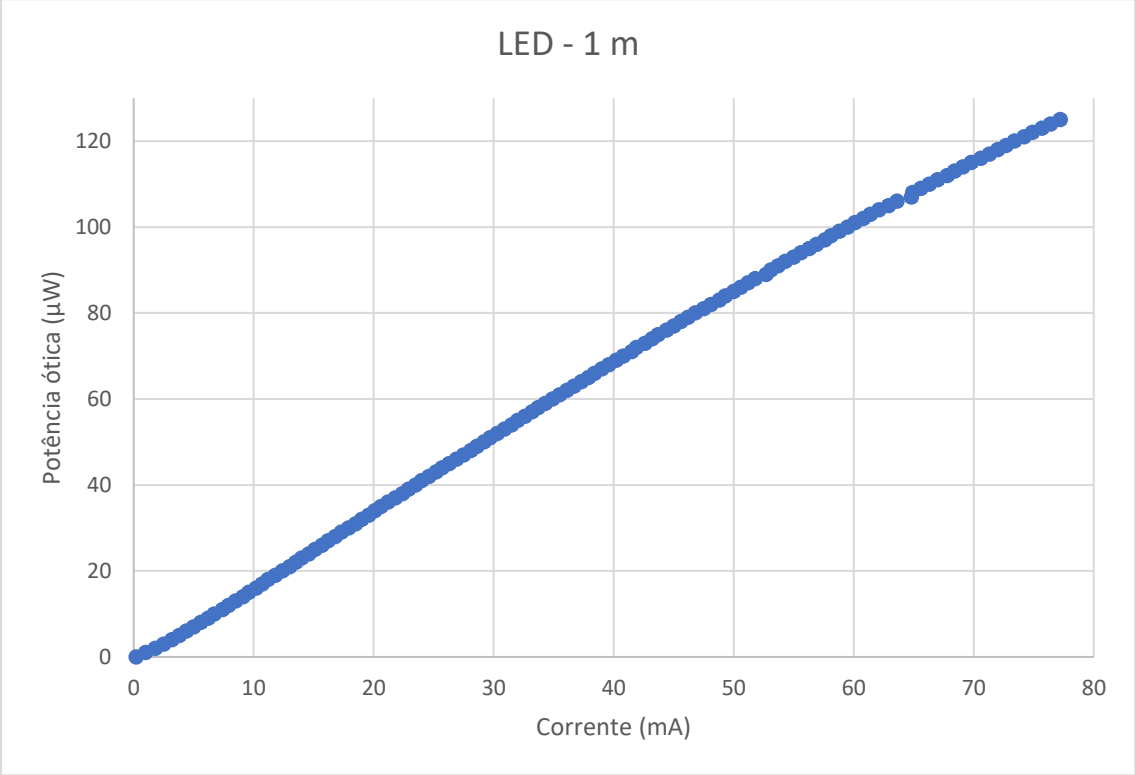

Figura 31. Potência ótica em função da corrente de excitação do Led com fibra de 1 m

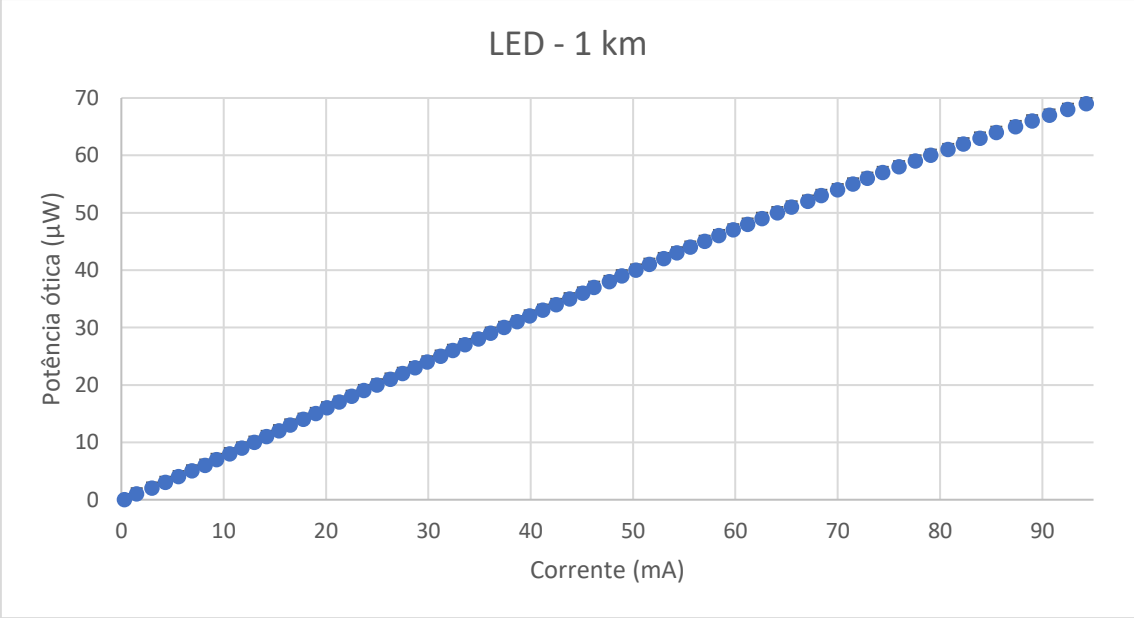

Figura 32. Potência ótica em função da corrente de excitação do LED com fibra de 1 km

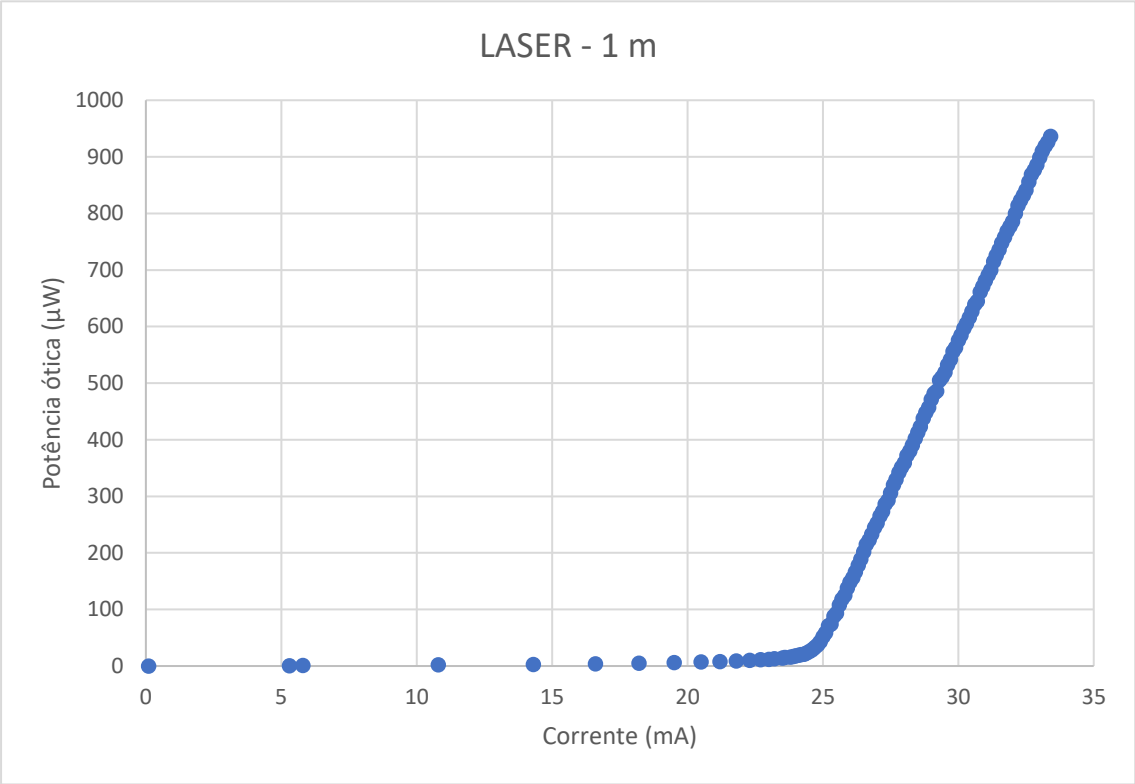

Figura 33. Potência ótica em função da corrente de excitação do LASER com fibra de 1 m

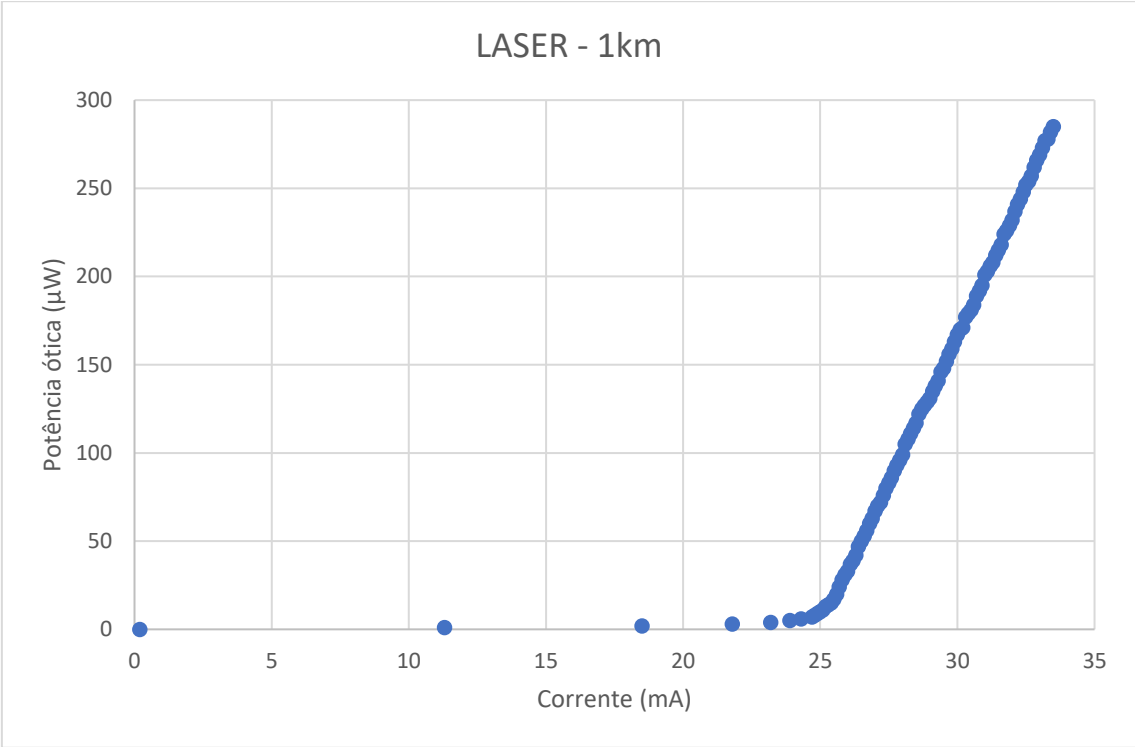

Figura 34. Potência ótica em função da corrente de excitação do LASER com fibra de 1 km

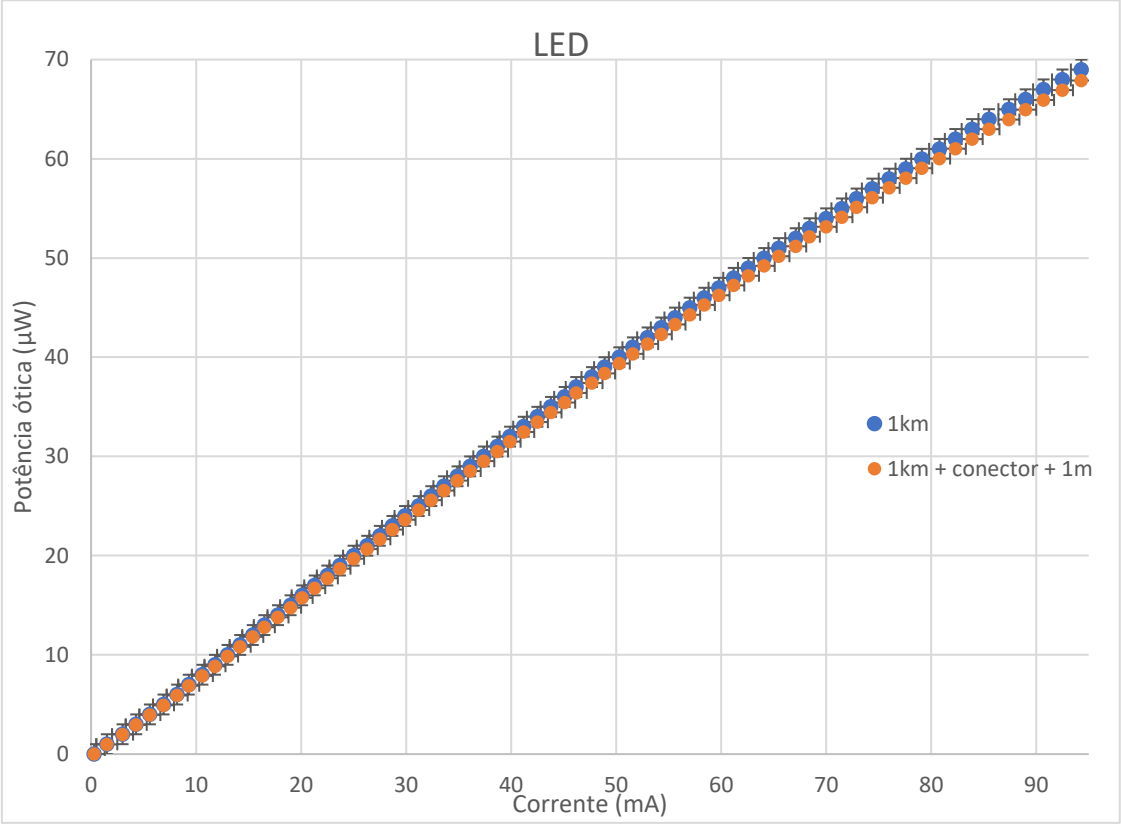

Figura 36. Atenuação pelo conector - LED

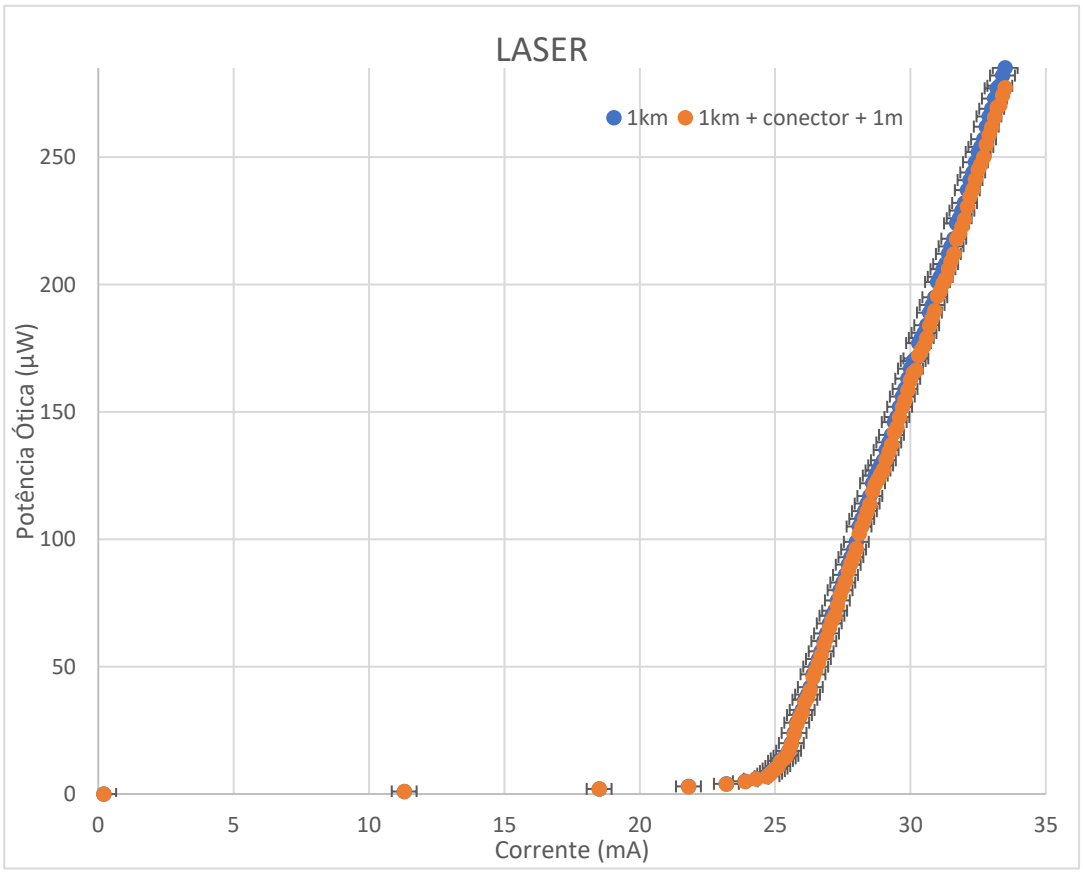

Figura 35. Atenuação pelo conector - LASER

5

Descrição do laboratório desenvolvido

A aplicação é exportada para um ficheiro JAR na versão 1.8.0.4, que é uma versão popular de desenvolvimento e geralmente compatíveis com os JRE (Java Runtime Environment) encontrados nos computadores comerciais. Para executar a aplicação, deve bastar abrir o ficheiro com a configuração pré-definida.

A Figura 37 mostra o primeiro ecrã, quando a aplicação é executada. No canto superior esquerdo (com a orientação normal para leitura) encontra-se o título do laboratório. Logo abaixo, está o painel “WAVEFORM GENERATOR”, que permite ligar e desligar a modulação do sinal de excitação, comutar entre comutação sinusoidal ou digital e manipular a frequência entre os 1 MHz e os 25 MH.

Logo abaixo desse painel, está o painel onde se fazem as conexões e, em particular, a saída elétrica do “WAVEFORM GENERATOR”.

Os painéis “LED TRANSMITTER” e “LASER TRANSMITTER” têm exatamente a mesma estrutura. Um controlador da corrente de excitação, representado por um manípulo rotativo, legendado por “DC BIAS (mA)”. O ecrã mostra o valor correspondente de corrente, com uma casa decimal.

Por baixo dos dois painéis, no painel com os conectores, vêm-se as entradas elétricas e saídas óticas dos dois instrumentos, legendados, respetivamente por “RF Input” e “Optical Output”.

O painel mais à direita tem o título de “PHOTORECEIVER” e contém um ecrã que mostra em μW a potência medida na entrada ótica em baixo. Contém também o botão que abre a janela do osciloscópio.

O painel com os conectores permite criar com o rato as ligações, escolhendo no painel mais inferior da janela o tipo de ligação que se pretende fazer. São dadas à escolha seis ligações, mas apenas estão implementadas a elétrica, representada pelo botão “RF Connectors” e as óticas de 1 metro (“Patch Cord”), 1 quilómetro (“1 km fibre”) e 1 quilómetro conectado a 1 metro (“Patch Cord + 1km”). No laboratório físico não foi possível utilizar a fibra de 2 quilómetros, porque estava danificada

Na Figura 38, está visível um exemplo de configuração do sinal de modulação, com forma sinusoidal, uma frequência de 16 MHz.

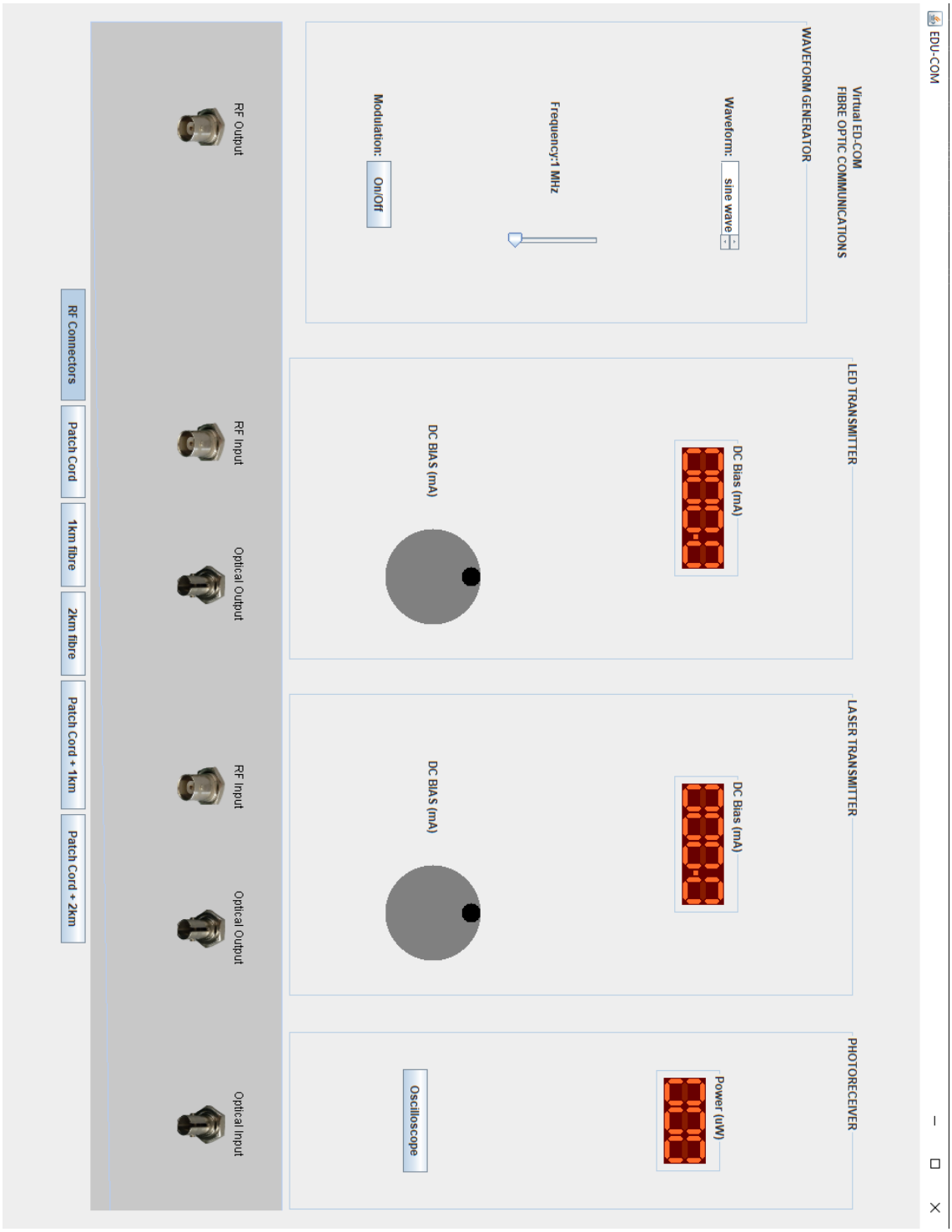

Figura 37. Vista inicial do laboratório virtual

Na figura 39, mostram-se as várias versões gráficas dos cabos conectados e desconectados. Para facilitar a utilização dos alunos, não é possível fazer conexões erradas como, por exemplo, conectar a ponta esquerda do conector RF a uma entrada elétrica.

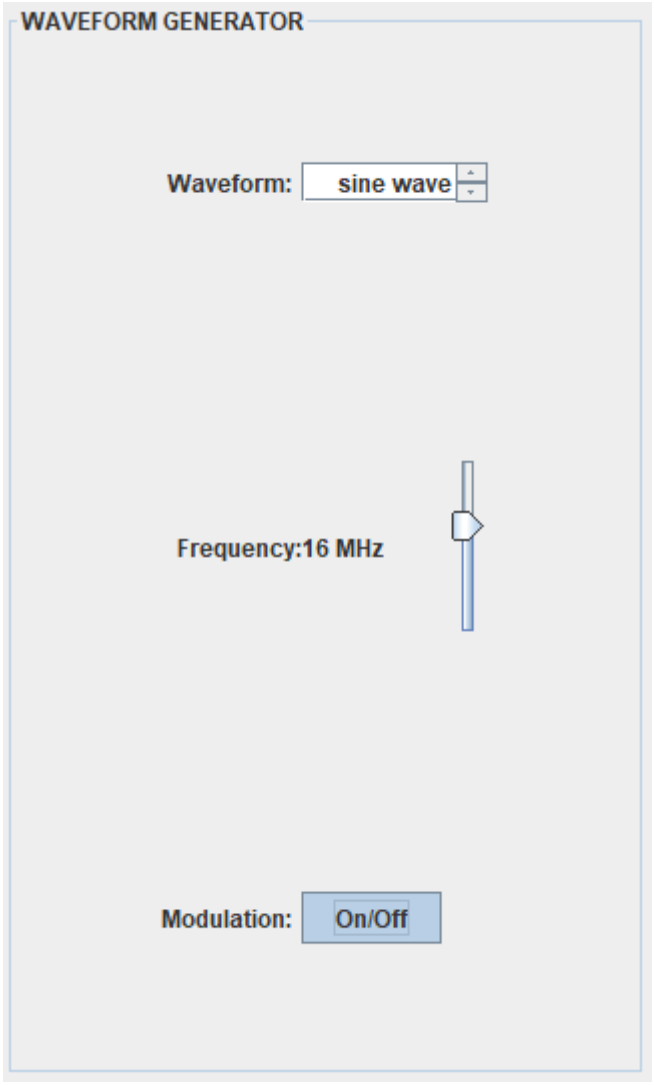

Figura 38. Painel gerador de sinais

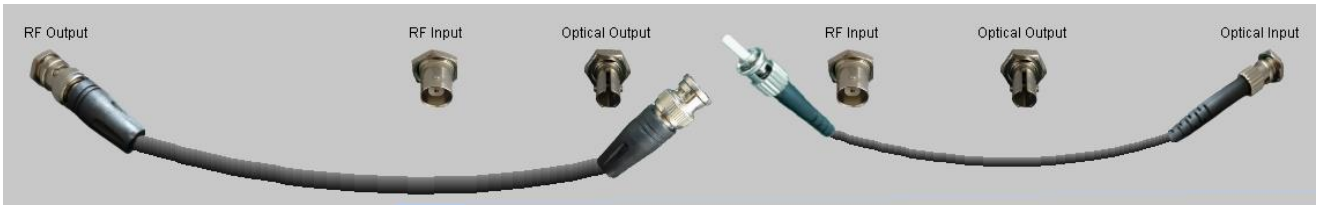

Figura 39. Painel de cablagem

Na Figura 40, pode ver-se um exemplo completo de configuração do sistema, com modulação digital, com a frequência automaticamente bloqueada em 4 MHz, o

LASER com corrente de excitação, mas desconectado, sendo que é o LED a transferir para o fotodetector a potência ótica, através da fibra de 1 metro.

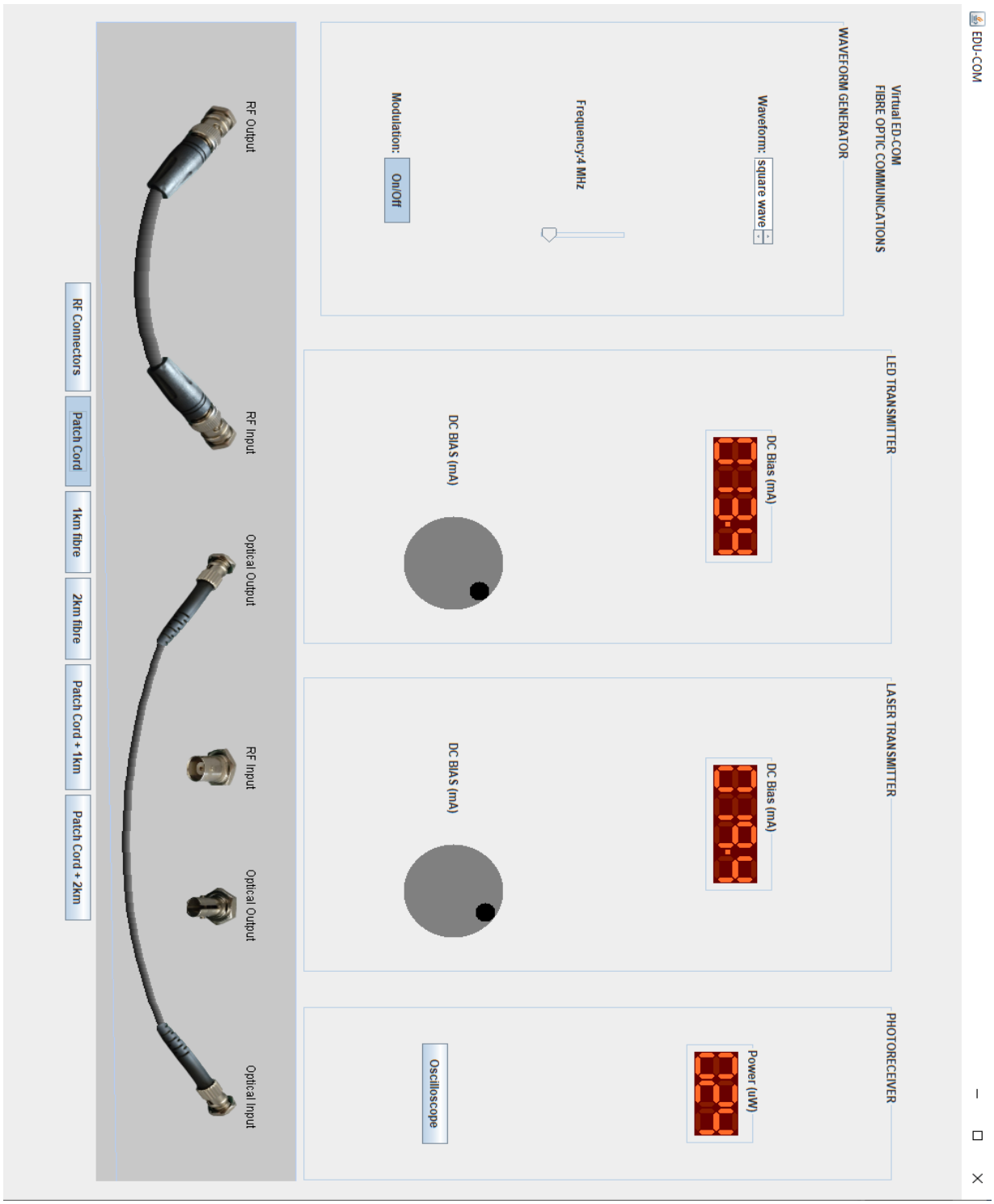

Figura 40. Exemplo de configuração LED

.

A figura 41 mostra a vista global com o osciloscópio ligado e a mostrar o sinal proveniente da configuração criada.

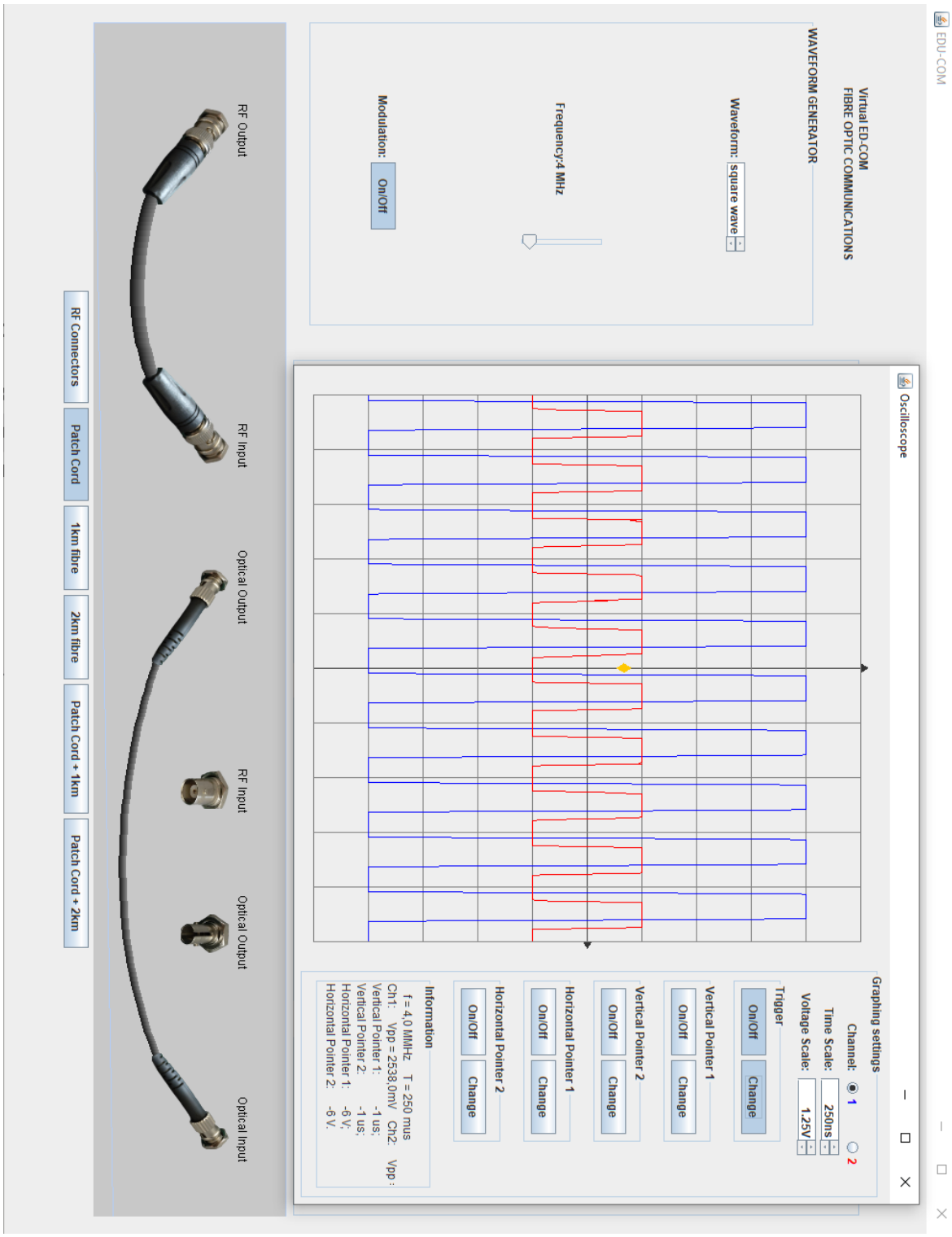

Figura 41. Exemplo de configuração para modulação digital

A figura 42, mostra o sinal no osciloscópio com o trigger configurado no ponto amarelo, no eixo vertical, uma escala temporal de 50 nanossegundos e uma escala de tensão de 1 V.

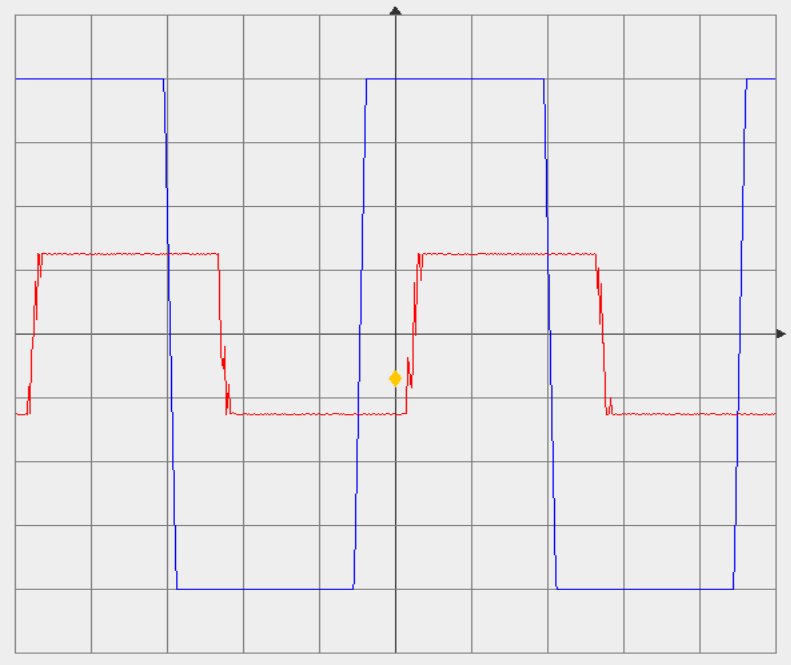

Figura 42. Gráfico do osciloscópio d modulação digital

A figura 43 mostra o gráfico do osciloscópio com modulação sinusoidal a 8 MHz e com a fibra de 1 km.

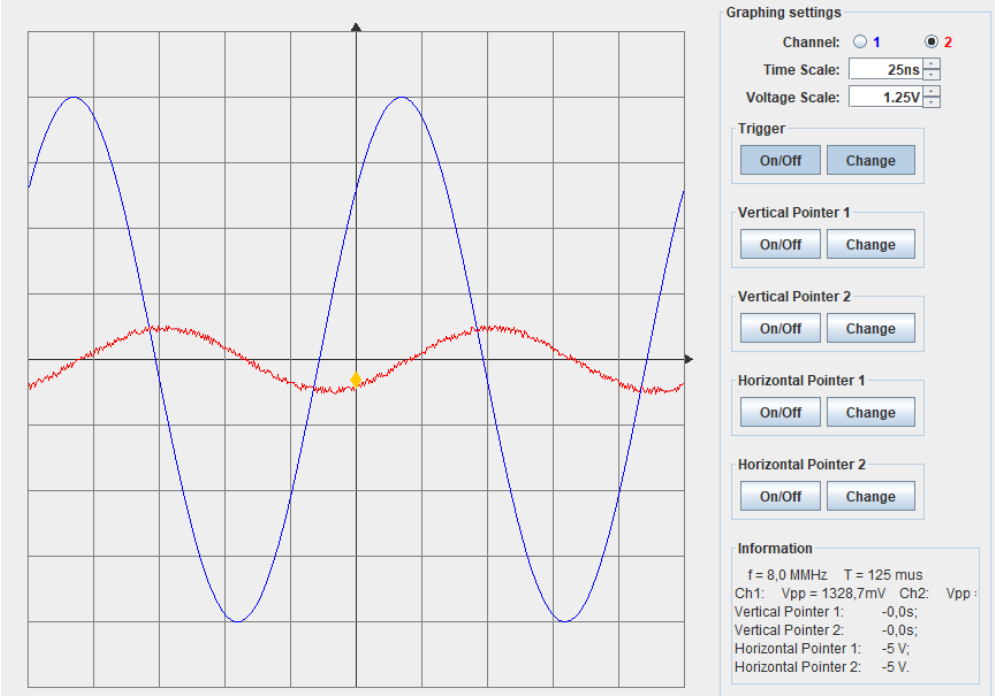

Figura 43. Exemplo de modelação sinusoidal

A figura 44 mostra o sinal com uma frequência superior, de 16 MHz. Com o *trigger* devidamente configurado, enquanto se faz uma variação relativamente lenta da frequência, é possível observar que há um ponto de cada sinal que não se mexe na horizontal, nem na vertical – a fase permanece inalterada. A diferença temporal entre esses dois pontos corresponde ao tempo de propagação na fibra ótica.

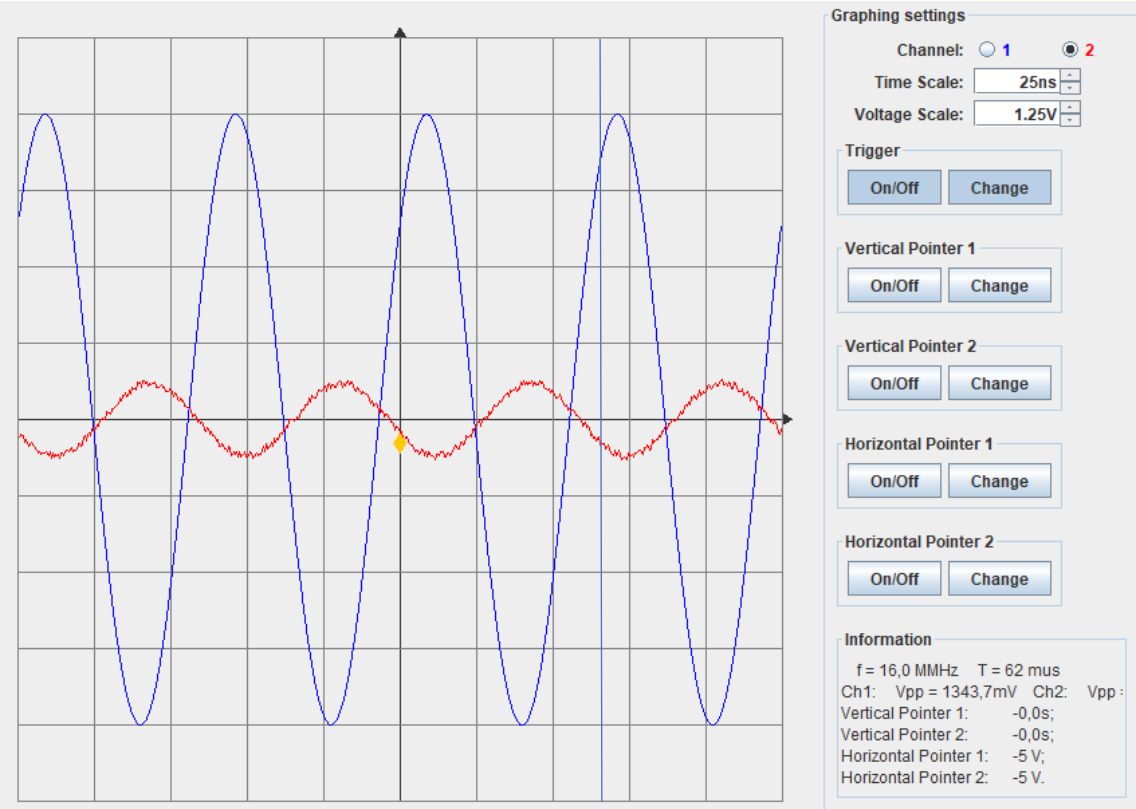

Figura 44. Exemplo de extração do atraso entre o sinal de entrada e o sinal de saída da fibra ótica

Os ponteiros podem ser ativados e depois, um de cada vez, reposicionados pelo pressionar do botão principal do rato dentro da tela dos gráficos.

6

Conclusão

O laboratório, construído à imagem do equipamento ED-COM da OptoSci, simula com sucesso um sistema simples para o ensino de comunicações por fibra ótica, constituído por um gerador de sinal de modulação, dois geradores de corrente de polarização, duas fontes de luz (um LED de 785 nm e um díodo LASER de 850 nm), fibras óticas e um fotorrecetor (fotodíodo). O sistema pode ainda ser analisado com um osciloscópio virtual, também ele simulado e desenvolvido em particular para o laboratório. À data, a única funcionalidade por implementar é o painel de informações do osciloscópio, que é o único requisito por cumprir. Pode medir-se a potência ótica de acordo com cada configuração possível do sistema, na vista de abertura da aplicação. Na vista do osciloscópio, podem medir-se tensões (como tal, os efeitos de atenuação) e intervalos de tempo, com o auxílio de ponteiros, pelo que é possível executar todas as atividades laboratoriais a que o projeto se propunha. Um dos ponteiros é o de *trigger* e há ainda dois ponteiros horizontais, para a tensão e dois ponteiros verticais, para a escala de tempo, que auxiliam nas medições, à semelhança de um osciloscópio real.

A disposição de todos os componentes interativos, replicando a do laboratório físico, bem como a interface gráfica de manipulação da cablagem, proporcionam intuitividade na operação do laboratório virtual. A compilação do código para um ficheiro .jar confere à aplicação portabilidade. A portabilidade, aliada à funcionalidade, pretende tornar o laboratório virtual numa ferramenta prática de estudo, bem como numa alternativa a considerar em situações de dificuldade em acesso ao laboratório físico, quer por indisponibilidade ou escassez de material, quer por razões de ordem sanitária, como a pandemia do COVID-19, que motivou a criação deste projeto.

Como trabalho futuro, a caracterização detalhada do equipamento laboratorial permitirá implementar simulações mais ricas, como as sugeridas por Oziazici ou André et al. Essas simulações podem, além de aumentar o realismo dos resultados obtidos, principalmente na resposta em frequência, abrir a porta a novas funcionalidades, como por exemplo o estudo da influência de diferentes parametrizações de um LASER no sistema de comunicação. O laboratório virtual é uma ferramenta de trabalho expansível,

com potencial para abranger com ainda maior profundidade o conteúdo programático dedicado aos princípios das fibras óticas e sua utilização em sistemas de comunicações.

Referências

[1] – https://www.igi-global.com/dictionary/the-design-of-virtual-laboratories-in-microwave-engineering-education/31700 acedido a 22 de agosto.

[2] – Leontyeva, I. A. (2018). Modern distance learning technologies in higher education: Introduction problems. *Eurasia Journal of Mathematics, Science and Technology Education*, *14*(10), 1–8. https://doi.org/10.29333/ejmste/92284

[3] - Andrews, J., & Higson, H. (2008). Graduate employability, "soft skills" versus "hard" business knowledge: A european study. *Higher Education in Europe*, *33*(4), 411–422. https://doi.org/10.1080/03797720802522627

[4] - Friesen, N. (2009). Open educational resources: New possibilities for change and sustainability. *International Review of Research in Open and Distance Learning*, *10*(5 SPL.ISS.). https://doi.org/10.19173/irrodl.v10i5.664

[5] - Bell, S., Douce, C., Caeiro, S., Teixeira, A., Martín-Aranda, R., & Otto, D. (2017). Sustainability and distance learning: a diverse European experience? *Open Learning*, *32*(2), 95–102. https://doi.org/10.1080/02680513.2017.1319638

[6] - Schneider, S. L., & Council, M. L. (2020). Distance learning in the era of COVID-19. *Archives of Dermatological Research*. https://doi.org/10.1007/s00403-020-02088-9

[7] - Lassoued, Z., Alhendawi, M., & Bashitialshaaer, R. (2020). An exploratory study of the obstacles for achieving quality in distance learning during the covid-19 pandemic. *Education Sciences*, *10*(9), 1–13. https://doi.org/10.3390/educsci10090232

[8] - Vasiliadou, R. (2020). Virtual laboratories during coronavirus (COVID-19) pandemic. *Biochemistry and Molecular Biology Education*, *48*(5), 482–483. https://doi.org/10.1002/bmb.21407

[9] - Stahre Wästberg, B., Eriksson, T., Karlsson, G., Sunnerstam, M., Axelsson, M., & Billger, M. (2019). Design considerations for virtual laboratories: A comparative study of two virtual laboratories for learning about gas solubility and colour appearance. *Education and Information Technologies*, *24*(3), 2059–2080. https://doi.org/10.1007/s10639-018-09857-0

[10] - Achuthan, K., Nedungadi, P., Kolil, V. K., Diwakar, S., & Raman, R. (2020). Innovation adoption and diffusion of virtual laboratories. International Journal of Online and Biomedical Engineering, 16(9), 4–25. https://doi.org/10.3991/ijoe.v16i09.11685

[11] - Altalbe, A. A. (2019). Performance Impact of Simulation-Based Virtual Laboratory on Engineering Students: A Case Study of Australia Virtual System. IEEE Access, 7, 177387–177396. https://doi.org/10.1109/ACCESS.2019.2957726

[12] - Achuthan, K., Francis, S. P., & Diwakar, S. (2017). Augmented reflective learning and knowledge retention perceived among students in classrooms involving virtual laboratories. Education and Information Technologies, 22(6), 2825–2855. https://doi.org/10.1007/s10639-017-9626-x

[13] - Hayes, D., Turczynski, C., Rice, J., & Kozhevnikov, M. (2014). Virtual-reality-based educational laboratories in fiber optic engineering. 12th Education and Training in Optics and Photonics Conference, 9289, 928921. https://doi.org/10.1117/12.2070779

[14] - Integra Web site, (December 20, 2020) www.integra.jp/en/specter/index.html.

[15] - Zemax Software, (December 20, 2020) http://www.radiantzemax.com/zemax/.

[16] - OptoDesigner, a light propagation simulator, (December 20, 2020) www.phoenixbv.com.

[17] - Schermer, R. T., & Cole, J. H. (2007). Improved bend loss formula verified for optical fiber by simulation and experiment. IEEE Journal of Quantum Electronics, 43(10), 899–909. https://doi.org/10.1109/JQE.2007.903364

[18] - Drake, D., Gopalakrishnan, J., Goswami, T., & Grosek, J. (2020). Simulation of optical fiber amplifier gain using equivalent short fibers. Computer Methods in Applied Mechanics and Engineering, 360. https://doi.org/10.1016/j.cma.2019.112698

[19] - Kutenin, V. S., & Kusaykin, D. V. (2019). Simulation of a multi-core optical fiber using a medium OptiBPM simulation. Scientific Development Trends and Education, 1. https://doi.org/10.18411/lj-05-2019-18

[20] - Binh, L. N. (2011). Optical fiber communications systems: Theory and practice with matlab® and simulink® models. In Optical Fiber Communications Systems: Theory and Practice with MATLAB and Simulink Models.

[21] - Clements, Paul; Felix Bachmann; Len Bass; David Garlan; James Ivers; Reed Little; Paulo Merson; Robert Nord; Judith Stafford (2010). Documenting Software Architectures: Views and Beyond, Second Edition. Boston: Addison-Wesley

[22] - https://en.wikipedia.org/wiki/Fibre-optic_Link_Around_the_Globe#cite_note-3 acessado no dia 29 de agosto de 2021.

[23] - https://www.atlassian.com/agile/project-management/user-stories acessa--do no dia 6 de setembro de 2021.

[24] - https://www.optosci.com/ed-com-fibre-optic-communications/ , acessado no dia 18 de setembro de 2021.

[25] - Ian Gorton (Março de 2011), Understanding Software Architecture, Research Gate (https://www.researchgate.net/publication/251164302).

[26] - https://www.freecodecamp.org/news/compiled-versus-interpreted-languages/ a--cessado no dia 15 de setembro de 2021.

[27] - https://www.freecodecamp.org/news/just-in-time-compilation-explained/, acessa--do no dia 15 de setembro de 2021.

[28] - Howard Austerlitz, in Data Acquisition Techniques Using PCs (Second Edition), 2003, (https://www.sciencedirect.com/topics/engineering/interpreted-language, acessado a 15 de setembro de 2021)

[29] - https://www.oracle.com/java/moved-by-java/timeline/#2020, acessado a 15 de setembro de 2021.

[30] - https://www.redhat.com/en/topics/integration/what-is-event-driven-architecture, 18 de setembro de 2021

[31] - https://medium.com/elixirlabs/event-bus-implementation-s-d2854a9fafd5 18 de setembro de 2021

[32] - https://www.udemy.com/course/java-swing-complete/, 20 de setembro de 2021

[33] - https://info.keylimeinteractive.com/the-four-pillars-of-object-oriented-programming 20 de setembro de 2021

[34] - https://docs.oracle.com/javase/tutorial/uiswing/concurrency/initial.html, março de 2021

[35] - https://docs.oracle.com/javase/tutorial/uiswing/concurrency/dispatch.html, março de 2021

[36] - https://docs.oracle.com/javase/8/docs/api/?java/io/BufferedReader.html, março de 2021

[37] - K. Thyagarajan, Ajoy Ghatak (2010), Lasers Fundamentals and Applications, 2.ª edição, Springer, p. 76

[38] - https://www.rp-photonics.com/sellmeier_formula.html, abril de 2021

[39] - M. S. Ozyazici, The Complete Electrical Equivalent Circuit of a Double Heterojuntion Laser Diode using Scattering Parameters, Journal of Optoelectronics and Advanced Materials Vol. 6, No. 4, Dezembro 2004, p. 1243 – 1253.

[40] - R. S. Tucker and D. J. Pope, "Microwave Circuit Models of Semiconductor Injection Lasers," in IEEE Transactions on Microwave Theory and Techniques, vol. 31, no. 3, pp. 289-294, Março 1983, doi: 10.1109/TMTT.1983.1131478.

[41] - R. Tucker and D. Pope, "Circuit modeling of the effect of diffusion on damping in a narrow-stripe semiconductor laser," in *IEEE Journal of Quantum Electronics*, vol. 19, no. 7, pp. 1179-1183, Julho 1983, doi: 10.1109/JQE.1983.1072005.

[42] - P. S. André, A- Nolasco Pinto, J.L. Pinto, F. da Rocha (1999). Extraction of Laser Rate Equations, *Proceedings of SPIE 3572 August 1999,* pp 141-146.

Anexo

Esqueleto da aplicação desenvolvida

Apresenta-se parte do código desenvolvido, desde a iniciação da aplicação, levada a cabo pela classe App.java, ao processamento primário da interação com o utilizador, em LedPanel.java, passando pelo ficheiro EDU_COM.java, onde a informação recebida pelo utilizador é processada novamente com o intuito de dar instruções às restantes componentes do código. São também apresentados a interface ParametersListener.java e o processador de eventos ParametersEvent.java. Os ficheiros apresentados servem de suporte à secção 4.1 do corpo do documento.

App.java:

```
import javax.swing.SwingUtilities;

public class App {

	public static void main(String[] args) {
			SwingUtilities.invokeLater(new Runnable() {
			public void run() {
				try {
					new EDU_COM();
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}
}
```

EDU_COM.java:

```
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.beans.PropertyChangeEvent;
import java.beans.PropertyChangeListener;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.Timer;

public class EDU_COM extends JFrame {

	private static final long serialVersionUID = …;
	private JLabel title;
	private GeneratorPanel generatorPanel;
	private LedPanel ledPanel;
	private LaserPanel laserPanel;
	private PhotoReceiverPanel photoReceiverPanel;
	private ConnectorsPanel connectorsPanel;
```

```
	private WiringPanel wiringPanel;
	private Oscilloscope oscilloscope;


	private String ledCurrent;
	private String laserCurrent;
	private String fibreSetup = "off";
	private boolean led;
	private boolean laser;
	private boolean patchcord;
	private boolean fibre1km;
	private boolean outputOn;
	private boolean square;

	public EDU_COM() {
		super("EDU_COM");
		title = new JLabel("<html>Virtual ED-COM<br>"
				+ "FIBRE OPTIC COMMUNICATIONS</html>");
		generatorPanel = new GeneratorPanel();
		ledPanel = new LedPanel();
		laserPanel = new LaserPanel();
		photoReceiverPanel = new PhotoReceiverPanel();
		wiringPanel = new WiringPanel();
		connectorsPanel = new ConnectorsPanel();
		oscilloscope = new Oscilloscope();
		oscilloscope.setVisible(false);

		// generatorPanel setup
		generatorPanel.setParametersListener(new ParametersListener() {
			public void parametersEventOccurred(ParametersEvent e) {

	if(e.getChangedVariable().contentEquals("waveform")) {
					square = e.isSquare();
					if(oscilloscope!=null) {

	oscilloscope.getGraphPanel().setWaveform(square);
					}
				}
				else
if(e.getChangedVariable().contentEquals("frequency")) {

	oscilloscope.getGraphPanel().setFreq(e.getFrequency());

	oscilloscope.getSettingsPanel().setF(e.getFrequency());
				}
				else
if(e.getChangedVariable().contentEquals("modulation")) {

	oscilloscope.getGraphPanel().setModulated(e.getModulation());
					//System.out.println(e.getModulation());
				}
			}
		});

		// photoReceiverPanel setup
		photoReceiverPanel.setParametersListener(new
ParametersListener() {
			public void parametersEventOccurred(ParametersEvent e) {
```

```
                        if((oscilloscope!=null && !oscilloscope.isShowing()
|| oscilloscope == null) && e.isOscilloscopeOn()) {
                                oscilloscope.setVisible(true);

      oscilloscope.setLocation(ledPanel.getParent().getLocationOnScreen().x+
ledPanel.getLocation().x+5,

      ledPanel.getParent().getLocationOnScreen().y+ledPanel.getLocation().y-
50);
                                oscilloscope.setAlwaysOnTop(true);
                                oscilloscope.setVisible(true);

      oscilloscope.getGraphPanel().setWaveform(square);

      oscilloscope.getGraphPanel().setFreq(generatorPanel.getFrequency());

      oscilloscope.getGraphPanel().setFibreSetup(fibreSetup);
                        }
                  }
            });

            // wiring panel setup
            wiringPanel.addPropertyChangeListener(new
PropertyChangeListener() {
                  public void propertyChange(PropertyChangeEvent e) {
                        String property = e.getPropertyName();
                        if(property.contentEquals("detectorScreen")) {

      if(((WiringPanel)e.getSource()).isDetectorScreen()) {

      if(((WiringPanel)e.getSource()).isLedFibreCoupled()) {
                                          photoReceiverPanel.setLed();
                                    }
                                    else {
                                          photoReceiverPanel.setLaser();
                                    }

      photoReceiverPanel.setFibre(wiringPanel.getConnectedFibre());
                                    photoReceiverPanel.setState(true);
                                    outputOn = true;
                                    if(led) {
                                    if(patchcord) {
                                          fibreSetup = "led1m";
                                    }
                                    if(fibre1km) {
                                          fibreSetup = "led1km";
                                    }
                                    }
                                    else if(laser) {
                                          if(patchcord) {
                                                fibreSetup = "laser1m";
                                    }
                                    if(fibre1km) {
                                          fibreSetup = "laser1km";
                                    }
                                    }
                                    if(oscilloscope != null) {
```

```
	oscilloscope.getGraphPanel().setFibreSetup(fibreSetup);
					}
				}
				else {
					photoReceiverPanel.setState(false);
					outputOn = false;
					if(oscilloscope != null) {

	oscilloscope.getGraphPanel().setFibreSetup("disabled");
					}
				}
				photoReceiverPanel.forceUpdate();
			}
			else if(property.contentEquals("ledScreen")) {

	if(((WiringPanel)e.getSource()).isLedScreen()) {
					ledPanel.setState(true);
					ledCurrent =
ledPanel.getScreen().getText();
					led = true;
					if(outputOn) {
					if(patchcord) {
						fibreSetup = "led1m";
					}
					if(fibre1km) {
						fibreSetup = "led1km";
					}
				}
					if(oscilloscope != null) {

	oscilloscope.getGraphPanel().setFibreSetup(fibreSetup);
					}
				}
				else {
					led = false;
				}
				ledPanel.forceUpdate();
			}
			else if(property.contentEquals("laserScreen")) {

	if(((WiringPanel)e.getSource()).isLaserScreen()) {
					laserPanel.setState(true);
					laserCurrent =
laserPanel.getScreen().getText();
					laser = true;
					if(outputOn) {
					if(patchcord) {
						fibreSetup = "laser1m";
					}
					if(fibre1km) {
						fibreSetup = "laser1km";
					}
				}
					if(oscilloscope != null) {

	oscilloscope.getGraphPanel().setFibreSetup(fibreSetup);
					}
```

```
                                }
                                else {
                                      laser = false;
                                }
                                laserPanel.forceUpdate();
                          }
                    }
              });

              Timer timer = new Timer(100, new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                     if(oscilloscope!=null) {

       oscilloscope.setLocation(ledPanel.getParent().getLocationOnScreen().x+
ledPanel.getLocation().x+5,

       ledPanel.getParent().getLocationOnScreen().y+ledPanel.getLocation().y-
50);
                     }
                     ledCurrent = ledPanel.getScreen().getText();
                     laserCurrent = laserPanel.getScreen().getText();
                     photoReceiverPanel.forceUpdate(ledCurrent, laserCurrent);
                }
            });
              timer.setRepeats(true);
            timer.start();

            connectorsPanel.setConnectorsListener(new ConnectorsListener() {
              public void ConnectorsEventOccurred(ConnectorsEvent e) {
                     if(e.getChangedVariable().contentEquals("rfConnector")) {
                           wiringPanel.setRfConnector(true);
                           wiringPanel.setFibre(false);
                     }
                     else
if(e.getChangedVariable().contentEquals("patchCord")) {
                           wiringPanel.setRfConnector(false);
                           wiringPanel.setFibre(true);
                           wiringPanel.setConnectedFibre("patchCord");
                           patchcord = true;
                           fibre1km = false;
                           if(oscilloscope != null) {
                                 if(outputOn) {
                                       if(led) {

       oscilloscope.getGraphPanel().setFibreSetup("led1m");
                                       }
                                       if(laser) {

       oscilloscope.getGraphPanel().setFibreSetup("laser1m");
                                       }
                                 }
                           }
                     }
                     else if(e.getChangedVariable().contentEquals("fibre1km"))
{
                           wiringPanel.setRfConnector(false);
                           wiringPanel.setFibre(true);
```

```
                        wiringPanel.setConnectedFibre("fibre1km");
                        fibre1km = true;
                        patchcord = false;
                        if(oscilloscope != null) {
                              if(outputOn) {
                                    if(led) {

      oscilloscope.getGraphPanel().setFibreSetup("led1km");
                                    }
                                    if(laser) {

      oscilloscope.getGraphPanel().setFibreSetup("laser1km");
                                    }
                              }
                        }
                  }
                  else
if(e.getChangedVariable().contentEquals("patchCord1km")) {
                        wiringPanel.setRfConnector(false);
                        wiringPanel.setFibre(true);
                        wiringPanel.setConnectedFibre("fibre1001m");
                  }
                  else if(e.getChangedVariable().contentEquals("fibre2km"))
{
                        wiringPanel.setRfConnector(false);
                        wiringPanel.setFibre(true);
                        wiringPanel.setConnectedFibre("fibre2km");
                  }
                  else
if(e.getChangedVariable().contentEquals("patchCord2km")) {
                        wiringPanel.setRfConnector(false);
                        wiringPanel.setFibre(true);
                        wiringPanel.setConnectedFibre("fibre2001m");
                  }
            }
          });

          layoutComponents();
            setVisible(true);

            try {
                  Thread.sleep(0);
                  repaint();
                  setTitle("EDU-COM");
                  setSize(1300,1000);
                  setResizable(true);
                  setVisible(true);
                  setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            } catch (InterruptedException e1) {
                  // TODO Auto-generated catch block
                  e1.printStackTrace();
            }
      }

      public void layoutComponents() {

            setLayout(new GridBagLayout());
```

```
            GridBagConstraints gc = new GridBagConstraints();

            gc.fill = GridBagConstraints.NONE;


            ////// FIRST PANEL //////////
            //label
            gc.weightx = 0.025;
            gc.weighty = 0;
            gc.gridx = 0;
            gc.gridy = 0;
            //gc.anchor = GridBagConstraints.LINE_END;
            add(title, gc);
            //waveform generator
            gc.gridy = 1;
            add(generatorPanel, gc);
            ////// LED PANEL /////////
            gc.gridy = 0;
            gc.gridheight = 2;
            gc.gridx++;
            gc.weighty = 0;
            add(ledPanel,gc);
            ////// LED PANEL /////////
            gc.gridy = 0;
            gc.gridheight = 2;
            gc.gridx++;
            gc.weighty = 0;
            add(laserPanel,gc);
            ////// PHOTORECEIVER PANEL /
            gc.gridy = 0;
            gc.gridheight = 2;
            gc.gridx++;
            gc.weighty = 0;
            add(photoReceiverPanel,gc);
            ////// WIRING PANEL ////////
            gc.gridy = 2;
            gc.gridheight = 1;
            gc.gridx = 0;
            gc.gridwidth = 4;
            add(wiringPanel,gc);
            ////// CONNECTORS PANEL ////////
            gc.gridy = 3;
            gc.gridheight = 1;
            gc.gridx = 0;
            gc.gridwidth = 4;
            add(connectorsPanel,gc);

      }
}
```

LedPanel.java:

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.beans.PropertyChangeEvent;
```

```
import java.beans.PropertyChangeListener;


import javax.swing.BorderFactory;
import javax.swing.JLabel;
import javax.swing.JPanel;

import javax.swing.border.Border;


public class LedPanel extends JPanel {

	private static final long serialVersionUID = …;

	private JScreen screen;
	private JPanel screenPanel = new JPanel();
	private JLabel dcBiasLabel;
	private JKnob dcBiasKnob;
	private ParametersListener parametersListener;
	boolean state = true;

	public LedPanel() {

		Dimension dim = getPreferredSize();
		dim.width = 330;
		dim.height = 610;
		setPreferredSize(dim);
		setMinimumSize(dim);

		screen = new JScreen("0.0 mA", "ledPanel");
		screenPanel.add(screen,BorderLayout.CENTER);
		Border screenInnerBorder = BorderFactory.createTitledBorder("DC
Bias (mA)");
		Border screenOuterBorder =
BorderFactory.createEmptyBorder(5,5,5,5);

	screenPanel.setBorder(BorderFactory.createCompoundBorder(screenOuterBo
rder,screenInnerBorder));

		dcBiasLabel = new JLabel("DC BIAS (mA)");

		dcBiasKnob = new JKnob(0);

		dcBiasKnob.addPropertyChangeListener(new
PropertyChangeListener() {
			public void propertyChange(PropertyChangeEvent e) {
				double theta = ((JKnob)e.getSource()).getAngle();
				double bias = theta;
				if(state) {
					screen.setText(String.valueOf(theta));
				}
				ParametersEvent ev = new ParametersEvent(this,
bias, "dcBias");
				if(parametersListener != null) {

	parametersListener.parametersEventOccurred(ev);
				}
			}
```

```
            });

            // set borders
            Border innerBorder = BorderFactory.createTitledBorder("LED
TRANSMITTER");
            Border outerBorder = BorderFactory.createEmptyBorder(5,5,5,5);
            setBorder(BorderFactory.createCompoundBorder(outerBorder,
innerBorder));

            layoutComponents();

      }

      public void layoutComponents() {

            setLayout(new GridBagLayout());

            GridBagConstraints gc = new GridBagConstraints();

            gc.gridwidth = 2;
            gc.gridx = 0;
            gc.gridy = 0;
            gc.weightx = 0.025;
            gc.weighty = 0.025;
            add(screenPanel, gc);

            gc.anchor = GridBagConstraints.LINE_END;
            gc.gridwidth = 1;
            gc.gridy++;
            gc.weightx = 1;
            add(dcBiasLabel, gc);
            gc.anchor = GridBagConstraints.CENTER;
            gc.gridx++;
            gc.weightx = 1;
            add(dcBiasKnob, gc);

      }

      public void forceUpdate() {
            if(!state) {
                  screen.setText("0");
            }
            else {
                  screen.setText(String.valueOf(dcBiasKnob.getAngle()));
            }
      }

      public void setState(boolean state) {
            this.state = state;
      }

      public JScreen getScreen() {
            return screen;
      }

}
```

ParametersListener.java:

```
import java.util.EventListener;

public interface ParametersListener extends EventListener {
	public void parametersEventOccurred(ParametersEvent e);
}
```

ParametersEvent.java:

```
import java.util.EventObject;

public class ParametersEvent extends EventObject{

	private static final long serialVersionUID = -6448968969397565947L;
	private String changedVariable;
	private int frequency;
	private boolean square; //waveform
	private boolean modulation;
	private double dcBias;
	private boolean oscilloscopeOn;


	public ParametersEvent(Object source) {
		super(source);
	}

	public ParametersEvent(Object source, int value, String variable) {
		super(source);
		this.changedVariable = variable;
		if(variable.contentEquals("frequency")) {
			this.frequency = value;
		}
	}

	public ParametersEvent(Object source, boolean value, String variable)
{
		super(source);
		if(variable.contentEquals("modulation")) {
			this.modulation = value;
		}
		else if(variable.contentEquals("oscilloscope")) {
			this.oscilloscopeOn = value;
		}
		else if(variable.contentEquals("waveform")) {
			this.square = value;
		}
		this.changedVariable = variable;
	}

	public boolean isSquare() {
		return square;
	}

	public Boolean getModulation() {
		return modulation;
	}

	public void setModulation(Boolean modulation) {
```

```
		this.modulation = modulation;
	}

	public double getDcBias() {
		return dcBias;
	}

	public void setDcBias(double dcBias) {
		this.dcBias = dcBias;
	}

	public void setFrequency(int frequency) {
		this.frequency = frequency;
	}

	public ParametersEvent(Object source, double value, String variable) {
		super(source);
		if(variable.contentEquals("dcBias")) {
			this.dcBias = value;
		}
	}

	public int getFrequency() {
		return frequency;
	}

	public String getChangedVariable() {
		return this.changedVariable;
	}

	public boolean isOscilloscopeOn() {
		return oscilloscopeOn;
	}
}
```