João Miguel Fonseca de Carvalho

Licenciado em Ciência e Engenharia Informática

Modelo de interação flexível num Editor de Modelos operado por voz e gestos

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática

Orientador: Professor Doutor Vasco Miguel Moreira Amaral,
Professor Associado,
Faculdade de Ciências e Tecnologia,
Universidade NOVA de Lisboa

Júri

Presidente:	Professor Doutor Nuno Correia
Arguente:	Professora Doutora Isabel Brito
Vogal:	Professor Doutor Vasco Amaral

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Março, 2021

Modelo de interação flexível num Editor de Modelos operado por voz e gestos

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Agradecimentos

Ao meu orientador, o Professor Doutor Vasco Miguel Moreira Amaral, o meu obrigado pela disponibilidade, saber transmitido, opinião e crítica que resultaram numa colaboração fundamental para a realização desta dissertação.

Pelo apoio incondicional, agradeço à minha família, amigos e namorada que sempre estiveram presentes e disponíveis para me transmitir uma palavra de incentivo.

Gostaria de deixar o meu agradecimento, também, à minha colega, Teresa Chambel, pela troca de ideias e companheirismo durante todas as etapas desta dissertação.

Agradeço ainda a todos aqueles que testaram a ferramenta desenvolvida. O seu contributo foi essencial para este trabalho.

A todos, o meu mais sincero obrigado.

"Tenho em mim todos os sonhos do mundo."
Álvaro de Campos, heterónimo de Fernando Pessoa, *in* "Tabacaria"

Resumo

A Engenharia Orientada a Modelos (MDE) surgiu como uma metodologia, fundada em teoria e ferramentas, para desenhar e desenvolver sistemas de *software*. A MDE recorre a linguagens de modelação, desde propósito geral, como o UML *standard*, a linguagens de domínio específico. Estas são suportadas por editores de modelação, que tipicamente oferecem um paradigma gráfico e textual, concentrado na visão, contudo ignorando outras abordagens como voz e som que poderiam ser utilizadas num contexto industrial ou mesmo até por uma questão de usabilidade.

Esta dissertação tinha a intenção de refazer o modelo de interação fixo de uma plataforma já existente, de nome Model-By-Voice, cujo propósito é construir editores de modelos que trabalham com linguagens de modelação utilizando apenas voz e som, trazendo flexibilidade na interação oferecida aos seus utilizadores quando estes fazem operações na mesma (criar elementos, ler, atualizar e apagar) e navegam nos seus modelos.

Model-By-Voice utiliza as ferramentas Sphinx-4 e Google Cloud Speech para reconhecimento de voz e a ferramenta FreeTTS para sintetizar voz. O modelo de interação antigo, embora funcional, apresentava limitações relevantes ao nível da utilização prática da plataforma, uma vez que o utilizador tinha de treinar para utilizar a ferramenta. Este era um problema ao nível da produtividade e curva de aprendizagem uma vez que a adaptação a este tipo de plataformas/IDEs exige um tempo considerável.

Desta forma, este trabalho reformulou o modelo de interação da plataforma Model-By-Voice, fazendo a plataforma adaptar-se ao utilizador e não o contrário.

Como resultado deste trabalho, um protótipo de prova de conceito foi desenvolvido e foi realizada uma experiência piloto com utilizadores reais para averiguar a sua usabilidade.

Palavras-chave: Model-By-Voice, Interação Homem-Máquina, Modelo de Interação, Engenharia de Software Orientada a Modelos, Modelling Workbench

Abstract

Model-Driven Engineering (MDE) has emerged as a methodology, grounded in theory and tooling, to design and develop software systems. MDE makes use of modelling languages, ranging from general purpose like the standard UML, to dedicated modelling languages like domain-Specific Modelling Lamguages. Those are supported by modelling editors, that typically offer graphical and textual notations, concentrated on vision, but ignoring other approaches like voice and sound that could be used in industrial settiings or even for a matter of accessibility.

This dissertation had the intent of remaking the rigid interaction model of an existing platform named Model-By-Voice, which purpose is to build model editors that deal with modelling languages just using voice and sound, to bring felixibility in the offered interaction to its users when operating (create elements, read, update, and delete) and navigating through their models.

Model-By-Voice utilizes the tools Sphinx-4 e Google Cloud Speech for voice recognition and the tool FreeTTS to synthetize voice. The previous interaction model, although functional, had relevant limitations related to the pratical use of the platform since the user had to train himself to use the tool. This was a problem in terms of produtivity and learning curve since it takes a considerable amount of time for the users to adapt to this kind of platforms/IDEs.

Therefore, this work reformulated the interaction model of the platform Model-By-Voice to make the platform itself to adapt to the user and not otherwise.

As a result of this work, a proof of concept prototype was designed and we performed an experimental pilot study with real users to learn of its usability.

Keywords: Model-By-Voice, Human-Machine Interaction, Interaction Model, Model Driven Software Engineering, Modelling Workbench

Índice

Lista de Figuras

Lista de Tabelas

Siglas

API Application Programming Interface

AToMPM A Tool for Multi-Paradigm Modeling

CRUD Create, Read, Update, Delete

DSL Domain Specific Language

EGL Epsilon Generation Language

EMF Eclipse Modeling Framework

EOL Epsilon Object Language

GMF Graphical Modelling Framework

HMM Hidden Markov model

IDE Integrated Development Environment

MDA Model Driven Architeture

MDE Model Driven Engineering

MPS Meta Programming System

OMG Object Management Group

PSM Platform Specific Model

REST Representational State Transfer

SDK Software Development Kit

SiGeR Simple Gesture Recogniser

SLE Software Language Engineer

SUS System Usability Scale

TeDub Technical Diagram Understanding for the Blind

TTS Text to Speech

UML Unified Modeling Language

WAV WAVEform audio format

XMI XML Metadata Interchange

XML Extensible Markup Language

Capítulo 1

Introdução

1.1 Motivação e enquadramento

A Engenharia de Software Orientada a Modelos (*Model Driven Engineering (MDE)*) é uma metodologia que se concentra na derivação de *software* usando os modelos como artefacto principal no processo até à geração de código. Associado a esta tecnologia está o conceito de linguagens de modelação, em especial as linguagens para domínios específicos (*Domain Specific Language* (DSLs)). Estas permitem expressar conceitos usando terminologia do domínio do problema. As DSLs têm o propósito de aumentar a produtividade, uma vez que são desenhadas para serem úteis num limitado número de tarefas e não em qualquer tipo de situação como as linguagens de propósito geral [44]. Existem diversas ferramentas destinadas à modelação de sistemas, derivadas do inglês *Modelling Workbenches*, como é o caso do Eclipse Modeling Framework (EMF) [20] / Graphical Modelling Framework (GMF) [37], MetaEdit [49], Meta Programming System (MPS) [23], A Tool for Multi-Paradigm Modeling (AToMPM) [22], entre muitas outras. Contudo, as ferramentas existentes concentram-se na representação visual de modelos, ignorando recursos como o som ou gestos. As limitações descritas deixam espaço para explorar estes recursos e integrá-los num contexto de modelação, criando novas formas mais interativas de construir modelos. Modelar através de sons e gestos poderá revelar-se uma ferramenta útil tanto num contexto educativo como profissional. Ambos os mecanismos são bastante apelativos no panorama da evolução tecnológica. Estes mecanismos de interação poderão contribuir para minorar casos extremos do problema de usabilidade, como seja o caso dos engenheiros informáticos que possuem alguma incapacidade visual, não lhes sendo possível recorrer aos meios de modelação convencionais.

Anualmente, o *website Stack Overflow* [76] leva a cabo um questionário [77], destinado a engenheiros de software, com o objetivo de inferir os mais diversos aspetos acerca

das práticas, relacionadas à Informática, mais comuns entre os intervenientes. Em 2019, 90.000 engenheiros responderam ao questionário, dos quais 1,5% possuem algum tipo de dificuldade visual ou são completamente invisuais. Até agora, a pequena percentagem descrita não justificou um investimento no desenvolvimento de ferramentas para auxiliar estes engenheiros resultando numa enorme falta de apoio aos mesmos, principalmente na área da modelação de *software*, devido à sua forte componente visual.

Apesar de já existirem algumas ferramentas, que serão descritas ao longo dos capítulos seguintes, que exploram métodos alternativos de programação e já oferecem algum tipo de suporte a engenheiros informáticos invisuais, existe ainda muito por fazer, especialmente nas áreas da Informática que estão mais dependentes de um suporte gráfico (onde tipicamente se centra a modelação).

Esta dissertação surge no contexto do projeto Model-By-Voice, que se trata de uma plataforma que permite aos seus utilizadores modelarem sistemas através do reconhecimento de voz. Na sua versão anterior a esta dissertação, a plataforma Model-By-Voice apesar de permitir aos seus utilizadores modelar diagramas através de mecanismos de reconhecimento e sintetização de voz, apresenta ainda algumas limitações para os utilizadores da mesma. Em estudos anteriores, ficou claro que a ferramenta deveria possuir um modelo de interação flexível que se adaptasse às necessidades de cada utilizador, dando alternativas de paradigmas de navegação (leitura) dos modelos. Assim, esta dissertação procurou melhorar os mecanismos de interação homem-máquina existentes atualmente na modelação de diagramas, tendo sido desenvolvida uma solução para modelar com som e gestos integrada na plataforma Model-By-Voice.

1.2 Problema

Como mencionámos antes, foi desenvolvida anteriormente a este trabalho, uma prova de conceito [47] Model-By-Voice que foi bem sucedida em demonstrar o conceito de modelação de linguagens através do reconhecimento de voz. Contudo, depois de realizadas várias provas de usabilidade com utilizadores reais, o processo de interação da plataforma revelou-se moroso, pouco prático e de difícil navegação para os mesmos, o que resulta maioritariamente de uma interação rígida e pouco customizável.

O processo de interação anterior consistia em dar instruções à plataforma através de comandos de voz, que após serem reconhecidos pelas ferramentas Sphinx-4 [75] e Google Cloud Speech [35], eram transformados em operações na plataforma, e de seguida, o resultado das operações era sintetizado para voz pela ferramenta FreeTTS [33] dando *feedback* ao utilizador. Foi identificado em avaliações experimentais de usabilidade que este processo consumia muito tempo ao utilizador, sendo pouco confortável, o que é agravado pelo facto do protocolo de interação ser o mesmo para qualquer utilizador, independentemente do nível de experiência e/ou diferentes necessidades. Foi posta, então, a hipótese de que se existisse uma forma de gerar um protocolo de interação personalizado para cada utilizador, o processo de interação seria mais confortável e acelerado, visto que existiriam

muitas acções que poderiam ser automatizadas para certo tipo de utilizadores em função, por exemplo, da sua experiência. Em particular, o *feedback* dado pela ferramenta possuía certos aspetos específicos que podiam ser personalizados com certos sons não-vocais em vez de longas descrições.

1.3 Objetivo do trabalho

Como já referido, tendo em conta os problemas encontrados no modelo de interação anterior da plataforma, verificou-se que o processo de interação consumia demasiado tempo aos utilizadores, tornando-se "maçador" e pouco útil. Concluiu-se então ser necessário a reformulação deste modelo de interação para algo mais prático, funcional e principalmente flexível. Logo, podemos formular os objetivos do trabalho da seguinte forma:

Pretende-se introduzir uma solução inovadora que permite modelar numa determinada linguagem usando apenas voz com um modelo de interação flexível e personalizável, bem como melhorar e acelerar a interação já existente na plataforma Model-By-Voice de forma a aumentar a produtividade dos seus utilizadores.

1.4 Abordagem

Pretendia-se dotar o engenheiro de linguagens com uma ferramenta para adaptar o Model-By-Voice ao utilizador final. Sendo assim, o novo modelo de interação proposto adapta-se às necessidades de cada utilizador e, desta forma, acelera a interação com a plataforma.

A experiência de interação pode continuar a ser feita por comandos de voz, como anteriormente, no entanto foi também dotada de comandos gestuais e sons codificados, como palmas e outros sons (por exemplo, sinais curtos, estalidos, assobios, etc). O novo modelo de interação gera um protocolo de interação específico para cada utilizador, permitindo obter uma ferramenta personalizável, o que facilita o esforço cognitivo do utilizador, assim como o tempo de execução de ações na plataforma.

A solução proposta procura manter um nível de abstração fácil de compreender por todos os utilizadores, incluindo pessoas invisuais num caso extremo. Adicionalmente, procura-se manter um modelo de interação que não dependa de uma noção espacial 2D, a não ser que o utilizador assim o pretenda, uma vez que é oferecida a possibilidade de escolha do paradigma de navegação entre as duas alternativas: grafo ou tabela.

Uma vez metamodelada a linguagem (usando tecnologias EMF em Eclipse), a sintaxe concreta associada à sintaxe abstrata pode ser acompanhada da definição do modelo de interação para as operações de leitura e escrita de elementos do modelo que inclua sons não-vocais, gestos desenhados 2D e uma abordagem tabelar, para além da voz.

Agora os utilizadores podem também personalizar as características do sintetizador de voz da plataforma, podendo regular a velocidade, timbre e volume da voz. Foi introduzida também a noção de "utilizador", sendo que agora os utilizadores após usarem a

ferramenta pela primeira vez, poderão guardar as suas preferências e utilizá-las posteriormente. Acreditamos que a inclusão destes novos mecanismos de interação será um ponto de partida para uma maior inclusão na área da modelação, para a criação de um ambiente de modelação em que não é prático utilizar as mãos ou ecrãs, e até mesmo para criação de linguagens para sistemas de automação de casas (*building automation*).

1.5 Contribuições

Neste trabalho, com a reformulação do modelo de interação da plataforma Model-By-Voice, introduziu-se as seguintes contribuições:

- **Modelação através de sons não-vocais** - O utilizador poderá a partir de agora recorrer a sons não-vocais, como sons e palmas, para executar comandos na ferramenta. Algum do *feedback* da ferramenta poderá ser também fornecido através destes sons.
- **Modelação através de gestos desenhados** - O novo modelo de interação permite a modelação através de gestos desenhados que executam comandos na plataforma.
- **Modelação através de um paradigma de tabela** - Em alternativa ao paradigma de grafo anteriormente usado para a modelação de diagramas na ferramenta, os utilizadores podem agora também utilizar o novo paradigma de tabela.
- **Personalização do sintetizador de voz** - Os utilizadores poderão personalizar o volume, timbre e velocidade da voz do sintetizador da forma a que se sintam o mais confortável possível com a interação.
- **Personalização da plataforma através de protocolos de diálogo adaptáveis** - A plataforma adapta-se conforme o protocolo de interação definido. Utilizadores com diferentes níveis de conhecimento e dificuldades visuais terão uma interação personalizada às suas características.
- **Introdução de novos comandos na plataforma** - O utilizador poderá a partir de agora usufruir de novos comandos como copiar, colar e cortar nós, assim como criá-los através de padrões, podendo assim criar nós que tenham as mesmas características mais rapidamente.
- **Introdução da noção de 'utilizador' na plataforma** - Os utilizadores podem agora guardar as suas preferências como a velocidade, timbre e volume da voz, e estas estarão disponíveis sempre que o utilizador fizer *log-in* com o seu nome de utilizador na aplicação.

1.6 Estrutura

A estrutura deste documento irá consistir nos seguintes capítulos:

- **Capítulo 2**: **Conceitos principais** Neste capítulo serão apresentados conceitos abstratos relacionados com a temática de Engenharia de *Software* Orientada a Modelos.
- **Capítulo 3**: **Estado da arte** Neste capítulo serão apresentadas e discutidas tecnologias e projetos relacionados com o problema que se pretendia resolver.
- **Capítulo 4**: **Trabalho relacionado** Capítulo dedicado à apresentação e discussão de projetos semelhantes ao proposto, bem como a apresentação e discussão de possíveis tecnologias úteis para solucionar o problema.
- **Capítulo 5**: **Introdução de requisitos da ferramenta** Capítulo dedicado à arquitetura detalhada da solução desenvolvida nesta dissertação e apresentação das tecnologias utilizadas na sua construção.
- **Capítulo 6**: **Cenários de aplicação** Capítulo dedicado aos diferentes casos de estudo que beneficiarão do novo modelo de interação da plataforma Model-By-Voice.
- **Capítulo 7**: **Avaliação da usabilidade** Capítulo onde é descrito o processo de avaliação do protótipo do novo modelo de interação da plataforma Model-By-Voice construído e apresentado como foi conduzida a experiência empírica com utilizadores.
- **Capítulo 8**: **Resultados do estudo de usabilidade** Capítulo onde são apresentados e discutidos os resultados obtidos na experiência empírica com utilizadores.
- **Capítulo 9**: **Conclusão** Capítulo dedicado às conclusões finais acerca do propósito da dissertação e referência a trabalho futuro.

Capítulo 2

Estado da arte

Neste capítulo irá ser descrito o estado da arte relacionado com as tecnologias e ferramentas na área que abrange o problema que a dissertação se propõe a resolver, assim como conceitos abstratos acerca da temática de Engenharia de *Software* Orientada a Modelos. Como tal, serão apresentadas e discutidas tecnologias relevantes, bem como projetos que apresentam ideias que inspiraram a solução proposta.

2.1 Conceitos principais

2.1.1 Engenharia de *Software* Orientada a Modelos

2.1.1.1 Definição

A Engenharia de *Software* Orientada a Modelos (MDE) é a área da Engenharia de *Software* que se foca em modelos de domínio. Permite aos engenheiros de *software* criar abstrações através de modelos que os ajudam a conceber uma solução no domínio específico pretendido. A MDE oferece uma abordagem que combate a inabilidade de linguagens de terceira geração de aliviar a complexidade de plataformas e expressar conceitos de domínio eficientemente [67]. O processo de *design* é simples na MDE, uma vez que os modelos são muito fáceis de reutilizar devido à semelhança existente entre si.

2.1.1.2 Modelo

Um modelo consiste numa abstração de um sistema ou ambiente que permite aos engenheiros exprimir aspetos técnicos da arquitetura dos mesmos. Os modelos possibilitam descrever sistemas complexos em vários níveis de abstração, tornando mais fácil a interpretação do seu funcionamento e aproximando o problema da implementação de *software* [7]. Um modelo é criado com o propósito de responder a um determinado problema [78].

2.1.2 Metamodelo

Um metamodelo contém os conceitos para criar modelos que sejam instâncias desse metamodelo. Desta forma, os metamodelos devem ter associados um metamodelo respetivo, que define como será feita a metamodelação [78].

2.1.2.1 *Model Driven Architeture* (MDA)

Uma das iniciativas pela qual a MDE é conhecida é a *Model Driven Architeture (MDA)* da *Object Management Group (OMG)*. A MDA é constituída por quatro modelos em diferentes níveis de abstração [69]:

- *Computation Independent Model* - Modelo com maior nível de abstração, descreve o comportamento do sistema, estrutura e tecnologia não são especificadas.
- *Platform Independent Model* - Modelo independente da tecnologia em que será desenvolvido. É transformado em um ou mais modelos específicos através de ferramentas de transformação.
- *Platform Specific Model (PSM)* - Modelo específico da plataforma. Representa as implementações disponíveis numa tecnologia de desenvolvimento.
- *Code* - Representa o sistema através de código fonte.

2.1.2.2 Linguagens de domínio específico (DSL)

Uma DSL é uma linguagem de programação direcionada a um determinado domínio específico que contém expressividade limitada. As DSLs aumentam a produtividade entre programadores e melhoram a comunicação com especialistas do domínio. Uma DSL bem escolhida pode facilitar a compreensão de um bloco de código complexo, aumentando assim a produtividade de quem trabalha nele [32]. As DSLs foram desenvolvidas para resolver um problema específico e não qualquer problema, o que pode trazer algumas desvantagens, como custos elevados de design, implementação, manutenção e a possibilidade de serem demasiado restritas, caso o problema a resolver exija uma maior expressividade [47].

2.1.3 Linguagem natural

O reconhecimento de voz deve ser feito tendo em conta a linguagem natural, ou seja, adaptar-se à linguagem utilizada regularmente no dia-a-dia. Uma abordagem que propõe resolver problemas como não assumir abreviações como palavras diferentes ou não deixar ao critério do programador decidir quais as frases que se devem computar, são levantados em abordagens como a "Composition-based on-the-fly rescoring for salient n-gram biasing"[38], e "Aprendizagem de contexto não-supervisionada"[50]. Esta abordagem introduz uma noção de contexto que é elaborada tendo em conta as frases anteriormente

ditas pelo utilizador e calcula a probabilidade de cada significado possível para as palavras que o utilizador dirá posteriormente. Desta forma, é possível fazer o reconhecimento de voz com linguagem natural, uma vez que as abreviações utilizadas no dia-a-dia são entendidas pelo programa de forma fiável. Esta abordagem possui alguns problemas, tal como exigir informação de treino para computar a linguagem, o que torna difícil a correção de erros que surjam no futuro, ou caso o treino seja feito incorretamente, a abordagem poderá cometer erros sistematicamente. Contudo, esta demonstra ser bastante bem sucedida em comparação com outras abordagens em relação ao reconhecimento de linguagem natural [39].

2.2 Ferramentas de metamodelação

Existem várias ferramentas que permitem elaborar a modelação de sistemas e linguagens (*Modelling Workbenches*).

De forma a construírem os editores de DSLs, estas ferramentas procedem ao mapeamento sintático dos conceitos da linguagem para conceitos abstratos como "nó" ou "ligação". Posteriormente dá-se a geração dos editores para as DSLs pretendidas.

A ferramenta mais notória talvez seja a EMF, uma vez que esta está integrada em ambiente Eclipse, e consequentemente ser muito utilizada, consolidada e robusta [60]. A EMF é uma ferramenta de modelação para construir ferramentas e outras aplicações baseadas num modelo estruturado. A partir de um modelo especificado em XMI, a EMF disponibiliza ferramentas e apoio em *runtime* para produzir um conjunto de classes Java para o modelo, um conjunto de classes que permitem ver e editar o modelo e um editor simples.

2.2.1 Modelos de interação das ferramentas atuais de metamodelação de linguagens

Para além da EMF existem muitas outras ferramentas de metamodelação de linguagens, como por exemplo, a MetaEdit [49], a MPS [23], AToMPM [22] ou a Sirius [72]. Contudo, todas estas possuem a mesma limitação de não suportarem uma abordagem de interação além da diagramática/textual. A ferramenta Sirius oferece ainda uma abordagem tabelar. No entanto, recursos de interação como a voz, sons não-vocais e gestos não estão a ser utilizados nas principais ferramentas deste tipo. Existem alguns trabalhos que apresentam alguns avanços na modelação através da voz, como discutido no capítulo seguinte, contudo o trabalho desenvolvido até ao momento não oferece aos utilizadores flexibilidade suficiente e a opção de escolha da forma como pretendem interagir com as ferramentas.

2.3 Leitores de ecrã e ferramentas de auxílio para invisuais

As pessoas visualmente incapacitadas, principalmente as que desempenham profissões na área da Informática, possuem diversos mecanismos e ferramentas que auxiliam a interação das mesmas com os computadores.

Existem *screen-readers*, que produzem um *output* sonoro ou em braile, disponibilizados gratuitamente, como o NV Access [59] , o JAWS [42] , o Speakup [6], o Emacspeak [64] e o eSpeak [29]. De entre os *screen-readers* enumerados, o Speakup destaca-se por estar inserido no sistema opreativo Linux, mantendo as suas funcionalidades mesmo quando sistemas de alto nível falham; O Emacspeak que foi pioneiro na leitura do ecrã em forma de discurso, não se limitando a ler somente o conteúdo [5]; O eSpeak que se trata de um sintetizador de voz com suporte de mais de 40 linguagens.

Estes facilitam muito a navegação no computador para pessoas visualmente incapacitadas, uma vez que descrevem ao utilizador tudo o que se encontra no ecrã, em detalhe suficiente para que este possa navegar sem cometer erros de interpretação. A utilização de *screen-readers* para um utilizador que não é visualmente incapacitado pode parecer pouco prática, uma vez que o computador leva alguns segundos para descrever os elementos do ecrã. Contudo, após algum tempo de prática, os utilizadores visualmente incapacitados podem acelerar este processo aumentando a velocidade de leitura do ecrã e arranjando formas de sintetizar, o mais possível, a informação com que estão a trabalhar, como por exemplo em folhas de cálculo. Os *screen-readers* conseguem ler e descrever para o utilizador folhas de cálculo em *Excel* [31]. A Microsoft disponibiliza até um manual de como criar folhas de cálculo acessíveis para pessoas visualmente incapacitadas [1], por exemplo considera-se importante na criação de uma folha de cálculo:

- Acrescentar texto alternativo a todos os elementos relevantes, *hiperlinks* e *ScreenTips*.
- Dar um nome único a todas as tabelas.
- Utilizar uma estrutura de tabela simples.
- Escrever informação relevante no cabeçalho de cada coluna.

As folhas de cálculo revelam-se muito úteis para facilitar o trabalho de pessoas visualmente incapacitadas, uma vez que agregam bastante informação num espaço de ecrã razoavelmente pequeno e muito fácil de navegar através de um *screen-reader*. Recentemente foi desenvolvido o SAS Graphics Accelerator [66] que permite aos utilizadores terem acesso à informação presente em diagramas e gráficos em folhas de cálculo, algo que não era possível anteriormente.

Outras ferramentas que auxiliam as pessoas visualmente incapacitadas, podem ser classificadas nas seguintes categorias [5] :

- Ambiente e linguagens de programação especializados.

- Representações áudio e táteis do "espaço" de programação.
- Meios áudio e táteis para melhor a acessibilidade em geral ao computador.

Quando estão a ler código, os programadores formam modelos mentais do código que pode ser tratado como informação espacial. Esta informação espacial pode ser representada e navegada não visualmente, por exemplo, com sons que representam diferentes ações em Visual Basic [5], reforçando a usabilidade das folhas de cálculo para auxiliar a organização do pensamento de informáticos visualmente incapacitados.

A maior parte dos sistemas operativos oferece auxílio para pessoas invisuais e já vêm equipados com um *screen-reader*. O Windows tem o Narrador incluído, o Linux dispõe do Orca, e o MacOS X inclui o VoiceOver. Os sistemas operativos vêm também equipados com *screen magnifiers* que permitem aumentar o tamanho do que está a ser exibido no ecrã, para pessoas que sofrem de algum tipo de dificuldade visual mas não são totalmente incapacitadas nesse aspeto. Em termos de sistemas operativos, o Linux fica um pouco atrás dos seus concorrentes por existir pouco interesse da parte de empresas em investir em aplicações deste carácter para Linux, pelos seus problemas de ecossistema [57]. Contudo, este é um sistema operativo muito comum no ramo da informática e seria muito útil que as aplicações existentes para outros sistemas operativos, que auxiliam as pessoas invisuais, nomeadamente profissionais da informática, fossem estendidas para poderem ser utilizadas em Linux. Isto é, caso esta não pudesse comprometer o bom funcionamento das mesmas nos atuais sistemas operativos em que estão desenvolvidas.

Estudos sugerem que os informáticos incapacitados visualmente e os informáticos sem problemas de visão compreendem o código de forma semelhante e são ambos capazes de produzir resumos coerentes do mesmo [5]. Verifica-se ainda que a forma de ler o código é semelhante, sendo que os informáticos visualmente incapacitados podem acelerar o processo de leitura com auxílio de *screen-readeres*, como o NV Access [59]. Esta informação é relevante para a arquitetura da solução proposta, uma vez que modelos produzidos por informáticos eventualmente invisuais devem ser lidos e interpretados por informáticos sem esta incapacidade. Se ambos processam o código da mesma maneira, pode assumir-se que a abstração do mesmo ao nível de modelos será processada de igual forma, o que facilitou a disposição visual na interface gráfica do editor de modelos.

2.4 StructJumper

Apesar dos *screen-readers* oferecerem um grande auxílio a pessoas invisuais, no contexto da programação podem tornar-se muito pouco práticos, uma vez que apenas lêem uma linha de código de cada vez, limitando-se a uma dimensão 1D, sendo impossível obter noções de navegação no código. O StructJumper [8] é um *plug-in* para o IDE Eclipse que se propõe a resolver este problema. A abordagem deste *plug-in* consiste em colocar o código em estrutura de árvore onde o utilizador pode navegar facilmente através das teclas de navegação. Quando o utilizador navega, apenas lhe é dito informações relevantes, tal

como o nome do método, e apenas de seguida lhe são lidas as restantes informações. Desta forma, a navegação torna-se muito mais rápida e eficiente. Quando o utilizador entra na aplicação existem dois modos optáveis, sendo eles o nó atual ser o nó em que o utilizador ficou antes de sair ou então o nó atual ser onde se encontra o cursor do rato.

2.5 Kinect Studio

Existem ferramentas muito úteis para reconhecer gestos como é o caso do iGesture [40] e de trabalhos realizados no âmbito académico que serão discutidos no capítulo seguinte. Contudo estas ferramentas apenas funcionam com o desenho de formas na mesmas através do rato do computador ou teclado, e não através de gestos elaborados através de expressões corporais pelos utilizadores. Uma tecnologia que proporciona o reconhecimento e síntese de gestos é o Kinect [52] da Microsoft e a respetiva aplicação Kinect Studio. Kinect Studio é uma aplicação utilizada para guardar e reproduzir informações captadas pelo aparelho Kinect. Esta informação pode posteriormente ser utilizada para detetar os gestos elaborados pelo utilizador e associar os mesmos a instruções a ser seguidas por uma aplicação associada. É uma ferramenta muito útil para efetuar *debug* de aplicações Kinect [41]. Contudo, o reconhecimento de gestos através do Kinect pode produzir imagens com algum "ruído" e pouco precisas, o que pode resultar na transmissão à aplicação de informação pouco fidedigna. Foi construído um simulador que visa solucionar este problema através da simulação destas imagens com um erro de validação muito baixo, produzindo informação mais fidedigna[45].

A tecnologia Kinect também tem aplicação em videojogos como, por exemplo, o Avatar Kinect que é utilizado para mapear os movimentos de um utilizador no respetivo avatar no videojogo através do aparelho Kinect [17].

2.6 Chatbots

Uma tecnologia que tem estado na liderança da interação homem-máquinas são os *chatbots*. *Chatbots* são ferramentas que emulam a interação com um assistente humano e que operam, usualmente, segundo a inteligência artificial e aprendizagem automática. Estas permitem guiar e instruir o utilizador sobre os mais diversos aspetos de uma aplicação e também executar comandos pelo mesmo quando solicitados [61]. A interação com os *chatbots* é normalmente estabelecida através de texto e comandos de voz, sendo o segundo método o mais interessante no contexto da dissertação.

DialogFlow é o serviço de *chabots* disponibilizado pela Google. Este disponibiliza a tecnologia necessária para manter conversas de alta complexidade com os utilizadores através de inteligência artificial. O DialogFlow possui um ambiente *cloud-based* para descrever *chatbots* com voz e texto, uma interface baseada em conversação, e oferece apoio para linguagem natural em mais de vinte línguas. O que distingue o DialogFlow de outros serviços de *chatbots* como o IBM Watson Assistant [87], Amazon Lex [4] ou a Microsoft

Bot Framework [51] é a flexibilidade que oferece em se conectar com serviços externos. A ferramenta IBM Watson Assistant é semelhante ao DialogFlow no treino de aprendizagem automática que irá entender a linguagem natural dos utilizadores. Esta permite a construção de interfaces para realizar conversas e providencia um *Software Development Kit* (SDK) para construir aplicações à volta dos *chatbots*, no entanto, não se revela tão flexível como o DialogFlow neste aspeto. A Microsoft Bot Framework permite construir e fazer *deploy* de *chatbots* em sites e redes sociais. É composta por conetores de canais que ligam os *chatbots* a canais de mensagens, e um *bot builder* SDK (implementa a lógica pretendida e integra linguagem natural nos seus serviços). O Amazon Lex é um serviço para criar interfaces de realização de conversas com linguagem natural incorporada (extrai o modelo de linguagem natural de frases de treino), contudo ao contrário do DialogFlow, não tem capacidade de prestar apoio no processamento da linguagem natural. Este permite ainda criar conversas ligando *triggers* a ações.

Foi proposta um abordagem para automatizar a criação de *chatbots* modeladores que auxiliam os utilizadores, através de conversas de voz ou texto, a construir modelos de domínios específicos [61]. Pelas razões descritas acima, este projeto decidiu utilizar a ferramenta DialogFlow para a construção dos seus *chatbots*. Os *chatbots* são criados no LandBot.io ligando visualmente blocos a mensagens. Funcionalidades extra podem ser programadas utilizando uma ferramenta de programação que está incorporada na plataforma desenvolvida ou integrando serviços externos utilizando uma API REST. A plataforma desenvolvida não incorpora inteligência artificial intencionalmente, visto que era procurada uma solução simples. Este projeto tem uma contribuição relevante na temática que se propunha a explorar, uma vez que utiliza uma abordagem interativa e incremental para construir modelos e apresentou uma generalização de modelos UML para DSLs arbitrárias.

2.7 Modelos de interação em videojogos

Na indústria dos videojogos, a interação do utilizador com as personagens do jogo é uma componente fundamental para o bom desempenho do mesmo. Desta forma, esta é uma indústria que tem bastante interesse em estar na frente do desenvolvimento de ferramentas para melhorar esta experiência do utilizador.

Foi desenvolvido um sistema para gerar avatares interativos [17], no qual se desenvolveu um modelo de interação que consiste em utilizar uma ferramenta como o Dragon Speech Recognition [18] para fazer o reconhecimento e transcrição com precisão do texto ditado pelo utilizador. De seguida, este texto é passado a uma aplicação, por exemplo uma aplicação personalizada na linguagem de programação C, que irá realizar a análise do mesmo e passar esta informação ao modelo de ação do sistema que poderá ter várias formas, como por exemplo o Modelo de Markov [48]. Este modelo de ação poderá determinar uma série de gestos ou palavras que o avatar deverá dizer na transição de um

estado para o seguinte. A forma de interação do avatar com o utilizador pode ser personalizada consoante certas características do utilizador, de forma a que a interação seja a mais adequada possível. As informações relativas aos utilizadores são captadas através do microfone e da câmara do computador sendo guardadas numa base de dados. Esta é uma abordagem útil no contexto desta dissertação, uma vez que se introduziu a noção de "utilizador" no novo modelo de interação da plataforma Model-By-Voice, sendo agora possível guardar as suas preferências para que estas estejam disponíveis em utilizações futuras.

2.8 Categorização de sons

A interação com o utilizador, especialmente na ausência de ecrãs é feita através de sons apenas, estando dependente da interpretação que os utilizadores dão aos mesmos. Como tal, seria bastante útil existir um guião de "boas práticas" que categorizasse os sons mais adequados para diferentes ocasiões e quando é dado *feedback* pelo computador. O ideal seria existir uma abordagem semelhante à *"Physics Of Notation"* [54], que enuncia princípios para desenhar notações que sejam facilmente reconhecidas pela população em geral, mas em vez de abordar princípios para notações visuais, catalogar os sons que são apropriados para utilizar nas mais diversas situações. Estes sons deveriam ser imediatamente reconhecidos e associados ao seu significado, por exemplo, sons que representem erros durante a utilização de um programa.

Foi construída uma biblioteca de sons para *user inferfaces* [83] que tinha o intuito de oferecer várias opções sonoras categorizadas de forma a poderem ser adequadamente escolhidas para representar diferentes tipos de acontecimentos num sistema. Os sons foram divididos em duas categorias principais: sons de notificação e sons de interação. Os sons de notificação têm a principal função de chamar a atenção para um certo evento. A principal vantagem dos sons presentes nesta categoria é a sua total independência da interface gráfica, podendo transmitir informação ao utilizador exclusivamente através do som. Os sons de interação, por outro lado, tem a função de reagir a ações realizadas pelo utilizador e tornar a experiência do utilizador mais agradável. Tratam-se de sons que dão ao utilizador *feedback* das suas acções, se estavas correram como esperado ou se ocorreu algum erro. A escolha dos sons deve ser feita de forma a criar uma certa identidade dentro da aplicação, podendo mesmo até seguir um tema, como o *Skype* cujos sons seguem a temática da água. Desta forma o utilizador saberá o que o som representa sem necessitar de apoio de qualquer suporte gráfico.

Embora as práticas descritas tenham algum fundamento e devessem ser tidas em conta no desenho de uma plataforma que trabalhe com sons não-vocais, não existe, no estender do meu conhecimento, informação suficientemente sistematizada para tirar proveito no trabalho de reformulação do modelo de interação da plataforma Model-By-Voice.

2.9 Conclusões

Os sistemas informáticos têm desenvolvido intensamente a interação com os seus utilizadores, algo que é bastante relevante para utilizadores comuns e ainda mais para pessoas invisuais. Estas últimas dependem da interação com os sistemas para poderem elaborar o seu trabalho. As abordagens descritas acima são muito úteis, em especial para este tipo de utilizadores, e foram tidas em conta no desenvolvimento do novo modelo de interação da plataforma Model-By-Voice. Este não se foca particularmente neste tipo de utilizadores mas pretende que o mesmo seja extensível também aos mesmos, para que estes possam usufruir da ferramenta sem qualquer problema.

Pelo conhecimento, não existem soluções na área de modelação que vão na direcção que a solução proposta pretende ir. Mesmo após o trabalho desenvolvido no anterior modelo de interação da plataforma Model-By-Voice [47], não existem soluções de modelação orientadas principalmente por voz satisfatórias, como será discutido no capítulo seguinte.

Capítulo 3

Trabalho relacionado

Neste capítulo serão apresentados projetos que propõem resolver o mesmo problema enunciado no Capítulo 1, nomeadamente as partes do problema que estes resolvem e as partes para as quais não apresentam solução e que a solução proposta nesta dissertação se propõe a resolver. Também serão apresentadas tecnologias úteis e relevantes para a construção da solução proposta.

3.1 TeDub

Com a evolução da tecnologia, o surgimento de novas ferramentas que visam auxiliar as pessoas visualmente incapacitadas nas suas áreas profissionais tem tido um crescimento exponencial. Como tal, a área da Informática não é exceção, sendo o projeto TeDub [62] prova disso. Este projeto tencionava construir uma forma das pessoas visualmente incapacitadas acederem a todo o tipo de diagramas. O sistema que ambicionavam construir tinha o intuito de gerar automaticamente certas classes de gráficos, o que permitia que pessoas visualmente incapacitadas pudessem explorar os mesmos livremente. Os utilizadores poderiam então explorar e criar diagramas através de um *joystick* ou de um teclado [43]. Assim, pode-se concluir que foi adotada uma comunicação háptica para os utilizadores. Os modelos devem ser mantidos em formato XMI para que possam funcionar com modelos exportados de ferramentas como Rational Rose ou Poseidon UML. Contudo, esta ferramenta não teve o sucesso esperado entre os utilizadores alvo, uma vez que se tratava de um navegador de modelos e não de um completo editor de modelos. Depois de darem o projeto como terminado, os criadores afirmaram que o projeto poderia ter tido mais sucesso se a solução se baseasse em ferramentas de transformação para exportar os modelos para código HTML, mantendo os links e utilização em texto, que seria muito mais fácil de gerir nos *screen-readers* utilizados atualmente que já estão equipados para

ler páginas web [46].

3.2 VoiceToModel

VoiceToModel [73], também ferramenta do NOVA-LINCS, pretendia atingir o objetivo de modelar por voz. Contudo, a solução não era flexível a qualquer linguagem, sendo uma solução rígida e reduzida aos três tipos de modelos suportados, sendo estes *goal-oriented models*, *object models* e *feature models*. Comparado com o Model-By-Voice, oferece apoio para um menor número de linguagens de modelação, mas os comandos utilizados são menos genéricos, como por exemplo, existir um comando de "criar" para cada tipo de objeto.

O modelo de interação do VoiceToModel é semelhante ao modelo antigo do Model-By-Voice, utilizando as mesmas ferramentas de reconhecimento e sintetização de voz, que são as Sphinx-4, Google Cloud Speech e FreeTTS. Tal como o Model-By-Voice, os comandos possíveis de executar seguem os princípios das operações básicas Criar, Ler, Atualizar e Apagar (CRUD - Create, Read, Update, Delete). Como tal, este modelo de interação sofre dos mesmos problemas do modelo antigo do Model-By-Voice, sendo pouco prático para o utilizador uma vez que os comandos demoram algum tempo a ser validados e a dar *feedback* ao utilizador. A solução proposta nesta dissertação, apesar manter os princípios das operações CRUD, acelerou o modelo de interação e adicionou novas operações de "atalho" como "copiar", "cortar" e usar padrões para facilitar a criação de nós.

3.3 Proposta de modelação orientada por voz

A plataforma Model-By-Voice também já inspirou o desenho de plataformas semelhantes, como se pode ver nesta proposta de modelação orientada por voz [10]. Esta proposta difere do Model-By-Voice, uma vez que propõe modelação de software em todas as fases do desenvolvimento do software e não apenas no reunir de requisitos. A arquitetura proposta apresenta ainda vários objetivos para melhorar a produtividade entre modeladores com ou sem incapacidade visual. Os autores descrevem o processo da modelação orientada por voz em três passos principais:

- Processamento de discurso – Feito através de uma ferramenta, como por exemplo, a Dragon Naturally Speaking [18], que irá produzir um *output* de texto.
- Processamento de Linguagen Natural – Feito através de ferramentas como Amazon Lex [4], Google Natural Language [36] ou Watson Natural Language Understanding [87]. Para realizar este processamento de linguagem natural, os autores propõe a existência de um dicionário de comandos de modelação onde serão encontrados os conceitos de modelação. De seguida a ferramenta escolhida irá detetar palavras relevantes relativas a modelação, e por fim construir *inputs "tokenizados"*.

- Modelo do Conceito Específico – Para cada linguagem de modelação alvo, vai ser necessário realizar uma implementação independente. Alterando os *scripts* na linguagem de modelação ou aplicar transformações ao ficheiro do modelo, utilizando uma linguagem de transformação, como por exemplo, TXL.

A solução que será apresentada no capítulo seguinte também pretende que exista um protocolo de diálogo personalizado para cada linguagem de modelação alvo, na medida em que o *Software Language Engineer* (SLE) pode definir certos sons não-vocais ou gestos, para substituir comandos ou dar *feedback*, que são mais fáceis de associar para certos aspetos de modelação de uma dada linguagem. Contudo, esta personalização é feita automaticamente através de uma DSL e não diretamente nos *scripts* da linguagem de modelação, como é no caso da solução apresentada em cima. Este processo ser automático aumenta muito a produtividade do utilizador final, apresentando-lhe um modelo de diálogo adequado à linguagem de modelação que está a utilizar, sem que este se tenha de preocupar em ser ele mesmo a fazer as personalizações ao protocolo de diálogo.

3.4 Abordagem multi-paradigmática para integrar gestos e som numa ferramenta de modelação

Um artigo [3] propôs uma extensão para a atual abordagem de definição de sintaxe concreta adicionando sons e gestos, assim como a possibilidade de os combinar com as modalidades de interação tradicionais. Assim, ao aderirem à metodologia de modelação multi-paradigmática conseguiram definir uma linguagem de modelação onde é possível representar sintaxes concretas que mais tarde poderão ser exploradas para gerar o apoio necessário para implementar as modalidades de interação propostas. Trata-se de uma abordagem interessante que ambiciona verificar como o poder de expressão da linguagem que o utilizador pretende utilizar se adequa a ser estendido com novos meios de interação, como é o caso dos sons e gestos. Para o futuro deixam a intenção de estender a sintaxe concreta do EuGenia [30], de forma a que seja possível gerar apoio automaticamente para a interação entre o utilizador e a ferramenta, para a linguagem de domínio específico que se pretende utilizar. Desta forma, este trabalho constitui uma base para se construir ferramentas de modelação avançadas que dependam de formas de interação inovadoras. Esta proposta representa uma melhoria bastante significativa para programadores com algum tipo de incapacidade e também para diminuir a dificuldade de interação com modelos muito complexos.

O trabalho descrito acima revela-se útil para a solução que irá ser apresentada nesta dissertação, uma vez que na proposta acima os elementos de uma sintaxe abstrata podem ser descritos através de uma sequência de atividades, como por exemplo uma sequência de sons ou gestos, e a solução desenvolvida tem uma componente com uma DSL que gera os protocolos de interação para o utilizador conforme o seu nível de conhecimento e as

suas dificuldades. Esta possui uma abordagem de interação semelhante, que pode ser descrita em *sequence diagrams* com as ações que o utilizador pode utilizar.

3.5 Ferramenta Kevin

Num sistema de modelação por voz ou gestos, em que a componente visual não está tão vincada, é bastante fácil a modelação tornar-se confusa quando se atinge diagramas de maior complexidade e dimensão, sendo possível prever alguns problemas a nível de navegação no mesmo. Um Engenheiro de *Software* que seja visualmente incapacitado também poderá ter bastantes dificuldades a navegar num diagrama. Descrever o diagrama por palavras ou em braille irá revelar-se um processo muito moroso, pouco prático e muito desgastante para o utilizador. Este estudo [11] refere que uma boa maneira de apresentar um diagrama do tipo Data-Flow, que consiste numa representação de como a informação flui num sistema [16], a uma engenheiro de software visualmente incapacitado seria uma mapa tátil, sendo estes depois transcritos para braille ou discurso. Contudo, esta abordagem não permite ao utilizador editar e criar novos diagramas. Uma maneira de permitir ao utilizador visualmente incapacitado de ler, editar e criar novos diagramas seria , para o caso dos diagramas Data-Flow, criar um gráfico N ao quadrado que consiste numa tabela que contém toda a informação de um diagrama Data-Flow. No Gráfico N ao quadrado as transformação que ocorrem no diagrama Data-Flow estão situadas na diagonal principal da tabela (do lado de cima à esquerda para o lado de baixo à direita), uma ligação que sai de uma transformação é escrita na mesma linha que a transformação e uma conexão que entra numa transformação é escrita na mesma coluna que a transformação. Qualquer informático, até mesmo invisual, pode criar, editar e ler diagramas através do método do gráfico N ao quadrado com o auxílio de uma ferramenta chamada "Kevin" [11] que consiste num ecrã tátil que exibe uma tabela onde o utilizador pode escrever a informação necessária para criar um diagrama completamente novo ou importar diagramas já criados e editar os mesmos. Esta ferramenta é extremamente útil e adequada para indivíduos visualmente incapacidades uma vez que foi desenvolvida de forma a prestar um auxílio integrado e adequado a pessoas que sofrem desta condição. Apesar da ferramenta "Kevin" ser uma boa abordagem para realizar o modelo de interação entre o utilizador e a plataforma, pode apresentar algumas dificuldades para invisuais desde o nascimento que desconhecem qualquer noção espacial de dimensão 2D.

Desta forma, a solução proposta nesta dissertação tentou evitar uma abordagem principal semelhante, em que o utilizador tenha que ter este tipo de conhecimento, e propôs um nível de abstração adequado a qualquer tipo de utilizador, mesmo que este seja invisual, para que a ferramenta proposta tenha um impacto positivo e solucione os problemas que derivam das características de cada utilizador. Contudo, manteve-se a possibilidade do utilizador poder interagir com a plataforma através de uma estrutura em tabela semelhante à descrita, como forma alternativa às noções abstratas de nós e ligações, caso este não seja invisual ou tenha visão parcial, tendo noções básicas de uma estrutura 2D.

Esta abordagem de tabela resolverá então os problemas de navegação em diagramas mais complexos, sendo mais fácil organizar a disposição dos elementos do mesmo.

3.6 Melhorar a produtividade do utilizador em ferramentas de modelação através de *workflows* de modelação específicos

Atualmente as ferramentas de modelação revelam-se pouco eficientes, como era o caso da plataforma Model-By-Voice cujo modelo de interação anterior se reformulou. Atividades comuns em Engenharia Orientada por modelos (MDE), como criar uma DSL ou realizar uma transformação de modelos, são pouco intuitivas e a maior parte são constituídas por ações repetitivas. Foi proposta uma solução para este problema [34], onde os autores descrevem uma solução baseada em MDE onde o utilizador define um *workflow* reutilizável que pode ser parametrizado em tempo de execução. Visto que a solução pretendia automatizar as atividades o mais possível, a maior parte das tarefas são automáticas, ou seja, o utilizador não tem necessidade de as executar. As tarefas automatizadas são tarefas como carregar o formalismo para criar um metamodelo, como por exemplo, o *Ecore* no EMF (Linguagem existente no Epsilon que permite definir metamodelos), é uma tarefa que pode ser automatizada, uma vez que a localização desse formalismo é conhecida. Os autores desenharam uma DSL que é facilmente adaptável a qualquer ferramenta de modelação e que permite descrever o processo de desenhar *workflows* reutilizáveis para automatizar, mesmo que parcialmente, algumas atividades. O metamodelo desta DSL pode ser observado na Figura 3.1.

A solução que irá ser proposta no capítulo seguinte possui uma componente que consite também numa DSL que gerará o protocolo de interação personalizado para cada utilizador, tendo em conta o seu nível de conhecimento e as suas dificuldades. Assim, este

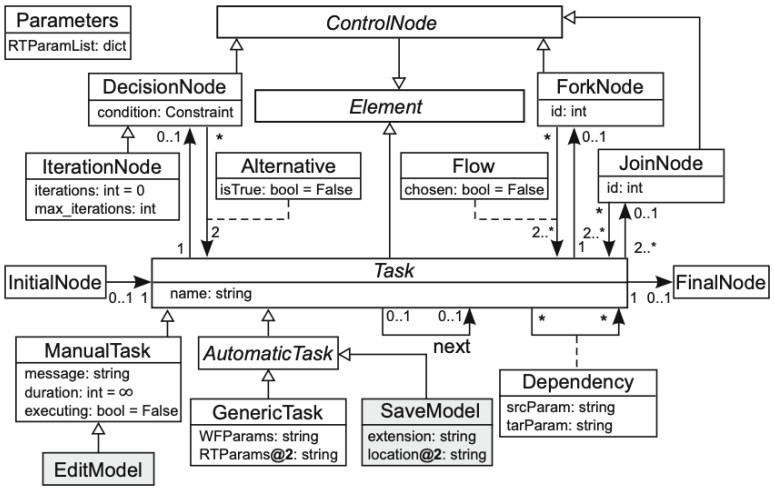


Figura 3.1: Metamodelo de *workflows* para ferramentas de modelação [Retirado do documento [34]].

trabalho revelou-se um excelente ponto de partida para construir a DSL que irá integrar a solução proposta, uma vez que a mesma faz bastante reutilização e é personalizável na construção do *workflow* dos protocolos de interação.

Na solução [34] apresentada pelos autores deste projeto, os *workflows* são descritos através de duas transformações para inicializar o *workflow* com parâmetros em tempo de execução e uma transformação de simulação para executar o mesmo.

O processo de utilização dos *workflows* é descrito pelos autores como sendo o seguinte:

- O *designer* define os *workflows* ao criar instâncias da DSL de *workflows*.
- Um utilizador, de seguida, escolhe qual o workflow que pretende utilizar.
- Posteriormente, o utilizador estabelece os parâmetros de tempo de execução que ativarão as tranformações de inicialização do *workflow*.
- De seguida, a tranformação de simulação começa e executa o *workflow* automaticamente. Quando uma tarefa manual é atingida no *workflow* é aberta uma janela que descreve qual a tarefa manual que necessita de ser executada e a simulação pára.
- Quando o utilizador completa a tarefa, a janela fecha e a simulação recomeça.

Após testarem este processo de modelação com alguns utilizadores, os autores concluíram que a automatização de tarefas revelou melhorias muito significativas relativas ao esforço cognitivo e mecânico feito pelos utilizadores para concluir as mesmas. Desta forma a modelação de *workflows* proposta pelos autores melhora a produtividade dos seus utilizadores em termos de tempo e interação com a ferramenta. Trata-se de uma abordagem que para além de automatizar tarefas comuns, também possibilita a integração de tarefas não-automatizáveis. A modelação da DSL de *workflows* foi desenhada para ser fácil de realizar diferentes atividades e fácil de estender ao automatizar mais tarefas e portátil para qualquer plataforma de modelação. A proposta apresentada pode ser implementada em qualquer plataforma de modelação, desde que a mesma possua os seguintes requisitos:

- Linguagens, botões, *canvas*, ações que o utilizador possa executar para interagir com a ferramenta têm de ser modeladas explicitamente dentro da ferramenta.
- Tem de disponibilizar um API para interagir com a plataforma e executar tarefas programaticamente.
- Tem de suportar o desenho e execução de transformações de modelos.

3.7 Tecnologias de reconhecimento e sintetização de voz

No âmbito de reformular o modelo de interação da plataforma Model-By-Voice, estendeu-se a interação entre o utilizador e a plataforma para não consistir apenas em comandos

de voz mas também em comandos gestuais e também em sons não-vocais. Antigamente, a plataforma Model-By-Voice interagia com o utilizador utilizando as ferramentas Sphinx-4 e Google Cloud Speech para reconhecimento de voz, e a ferramenta FreeTTS para a sintetização de voz. A ferramenta Sphinx-4 é uma ferramenta *open-source* desenvolvida na linguagem Java TM *Programming* que se pode conectar com as mais diversas aplicações e que incorpora padrões de *design* de sistemas existentes. Esta ferramenta dispõe também de diferentes componentes para diferentes tarefas [86]. A ferramenta Google Cloud Speech permite a conversão de voz em texto, dispõe de mais de 110 idiomas e revela-se muito útil uma vez que faz uma conversão de voz para texto muito precisa, mesmo em ambientes ruidosos. Na antiga solução do modelo de interação do Model-By-Voice a ferramenta Sphinx-4 solucionava os seguintes problemas:

- Capacidade de operar sem conexão à Internet.
- Ferramenta *open-source* que disponibilize utilização gratuita.
- Taxa de precisão credível no reconhecimento de palavras.

Esta ferramenta pode também ser treinada para descodificar sons não-vocais [15] [58] [1], o que será muito útil para melhorar a experiência de interação na plataforma Model-By-Voice através de sons codificados, como palmas.

Contudo, esta não se revelou muito precisa na conversão de voz para texto em ambientes ruidosos. Desta forma foi decidido, para o antigo modelo de interação, complementar a ferramenta Sphinx-4 com a ferramenta Google Cloud Speech, que se revelou útil para solucionar este problema [47].

Uma outra alternativa para realizar o reconhecimento de voz seria a plataforma Dragon Speech Recognition Engine [18], que se trata de uma plataforma de reconhecimento de voz que se destaca pela fiabilidade que garante ao se criar e editar ficheiros de texto e transcrição de voz para texto. Contudo, sendo uma ferramenta paga, não se revelou útil para o propósito desta dissertação.

Na plataforma Model-By-Voice a sintetização de voz era, e continua a ser feita, através da ferramenta FreeTTS. Esta ferramenta analisa o texto a converter e divide o mesmo em grupos de fonemas que posteriormente são convertidos em discurso audível. Este processo é desenvolvido através dos seguintes passos [85]:

- Normalização de texto – O Processador da ferramenta converte o texto de input em uma corrente de palavras.
- Análise linguística – O processador da ferramenta determina a informação semântica como frases.
- Análise lexical – O Processador da ferramenta determina a forma de pronunciar e identifica as sílabas.

[1] [58] retirada do *website Stack Overflow*

- Geração prosódia – O processador da ferramenta determina quando fazer pausas e a acentuação, duração e pronúncia das palavras
- Síntese de diálogo – A ferramenta gera informação áudio, geralmente concatenando unidades de diálogo

3.8 Musicg

Musicg é uma biblioteca de análise de áudio, escrita em Java, com o propósito de extrair características do áudio, sendo estas de alto ou baixo nível. Esta biblioteca permite executar operações sobre ficheiros de áudio, como ler ou cortar. Também providência ferramentas para o processamento de áudio digital e renderização para WAV *wavform* ou espectrograma [56].

Esta biblioteca dispõe da ferramenta *fingerprint* que permite comparar dois ficheiros de som em formato WAV e devolve uma percentagem de semelhança. Esta ferramenta revela-se muito útil para reconhecer sons não-vocais, uma vez que se planeava que os sons associados a comandos estivessem guardados e quando o utilizador reproduzisse um som não-vocal, este era comparado com os sons disponíveis. Caso após a análise da ferramenta *fingerprint* for devolvida uma percentagem de semelhança que se considere aceitável será executado o comando correspondente ao som que o utilizador fez.

3.9 Sonic

De forma a ir mais além na personalização do modelo de interação da plataforma Model-By-Voice, foi decidido personalizar também as características do sintetizador de voz em si, como a velocidade, timbre, taxa de amostragem e volume.

A biblioteca Sonic [74] dispõe de um simples algoritmo para acelerar ou abrandar ficheiros de áudio. Foi desenvolvida com o principal objetivo de permitir a pessoas invisuais aumentarem a sua produtividade ao acelerarem o discurso dos sintetizadores de discurso e terem uma melhor experiência de utilizador em diversas atividades do quotidiano, como ler um livro audível.

Desta forma, o utilizador, ao adaptar as características do sintetizador conforme as suas necessidades e gosto pessoal, terá uma experiência de utilizador mais confortável o que se estima que aumente a sua produtividade. A biblioteca Sonic revelou ser a escolha indicada para efetuar a personalização do sintetizador de voz uma vez que está disponível em java e previa-se uma boa integração com a plataforma Model-By-Voice.

3.10 iGesture

Existem várias alternativas para realizar a interação entre o utilizador e as plataformas. A plataforma Model-By-Voice já interagia com o utilizador através e reconhecimento e

síntese de comandos de voz. Com o intuito de expandir esta funcionalidade para efetuar a interação com gestos desenhados, foi investigado quais as ferramentas utilizadas de momento nas mais diversas áreas para efetuar esta funcionalidade. iGesture [40] é uma *framework* em Java para reconhecimento de gestos. Esta providencia suporte para diversos aparelhos como caneta digital, rato e até a possibilidade de testar novos aparelhos e adicionar novos algoritmos de reconhecimento de gestos. A representação de gestos na ferramenta iGesture tem algumas regras para garantir o seu bom funcionamento. Um requisito é que os gestos têm de ser agrupados por grupos, de forma a que os algoritmos de reconhecimento identifiquem facilmente de que gesto se trata. Cada classe que representa um gesto deve ser caracterizada por um nome e uma lista de descrições, de maneira a que seja possível atribuir uma forma geométrica, por exemplo, a um comando da ferramenta. Estas classes poderão ser posteriormente agrupadas e utilizadas para inicializar um algoritmo na ferramenta [71]. Os algoritmos de reconhecimento de gestos utilizados pela ferramenta iGesture são o algoritmo Rubine, para especificar os gestos através de exemplos [65], Extend Rubine, e os algoritmos Simple Gesture Recogniser (SiGeR) [79] e SiGrid. Apesar do objetivo principal da ferramenta ser o reconhecimento de processamento de gestos 2D, através do próprio rato do computador ou de aparelhos como caneta digital, o iGesture também oferece suporte para gestos 3D. O dispositivo de leitura de gestos 3D que funciona atualmente com a ferramenta é o controlador da consola *Wii*, que é conectado à mesma através de uma ligação *Bluetooth*. O algoritmo que atualmente faz o reconhecimento destes gestos 3D na plataforma é o algoritmo de Rubine que se baseia nas características dos gestos.

A arquitetura da ferramenta iGesture é constituída por um reconhecedor de gestos, ferramentas de avaliação, consola de gestão e algumas estruturas de dados como demonstra a Figura 3.2.

As estruturas de dados representam como os gestos são agrupados e representados. Uma classe de "Gesto" é uma representação abstrata de um gesto que posteriormente é agrupada em conjuntos. Esta classe não deve conter a descrição do gesto que representa, uma vez que diferentes algoritmos requerem uma diferente descrição do gesto. Assim cada classe de "Gesto" deve possuir um descritor que irá descrever, ao algoritmo a utilizar,

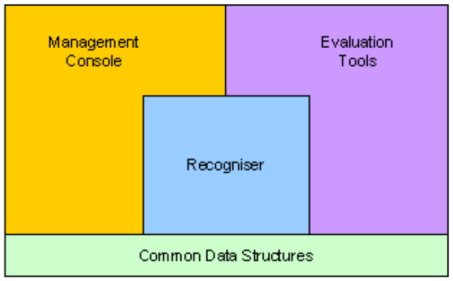


Figura 3.2: Modelo de arquitetura da ferramenta iGesture [Figura retirada de [63]].

o gesto em questão na forma necessária. O iGesture fornece descritores como o SampleDescriptor, TextDescriptor and DigitalDescriptor. O primeiro baseia a descrição no número de amostras, sendo este o descritor mais utilizado para treino. Já o TextDescriptor elabora uma descrição textual das direções que conectam os vários pontos que descrevem um gesto. O DigitalDescriptor representa um gesto através de uma imagem digital, este não é utilizado para reconhecer gestos, mas sim para visualizar os mesmos numa interface gráfica [63].

A ferramenta iGesture permite também definir e testar novos gestos. Esta possibilidade está consolidada no iGesture Workbench que contém as divisões "Gesture Set", "Test Bench", "Batch Processing" e "Test Set". O "Gesture Set" é onde o utilizador pode gerenciar os seus gestos e descritores. No "Test Bench", o utilizador pode encontrar a lista de algoritmos disponíveis e configurá-los da forma mais adequada. É nesta divisão também que o utilizador tem a possibilidade de escolher um gesto capturado por um dos dispositivos de input e correr o reconhecedor com um determinado grupo de gestos e algoritmo, permitindo ao utilizador fazer um reconhecimento manual. A divisão de "Test Set" é destinada a definir e gerir conjuntos de testes e gestos. Estes testes serão utilizados para testar os algoritmos recorrendo aos processos disponíveis na divisão "Batch Processing"[63].

O iGesture seria aplicável à solução desenvolvida, uma vez que se pode integrar em aplicações externas como seria o caso do Model-By-Voice. Uma vez que aplicações que pretendam dispor de eventos de gestos podem-se registar como "GestureEventListener" para um dado dispositivo. Caso um gesto seja executado, o dispositivo notifica todos os *listeners* registados. O *listener* posteriormente passa o gesto capturado para um reconhecedor, que depois de reconhecer o gesto notificará os gestores de eventos que se registaram com o reconhecedor. Um gestor de eventos irá implementar a interface "GestureHandler" e irá especificar qual a ação que deverá ser executada quando um certo gesto é reconhecido [63].

Existem ferramentas com objetivos semelhantes ao iGesture, tal como a *framework* Pointer ToolKit [9] que se propõe a desenvolver e testar software que recorre a vários dispositivos de input. O funcionamento da ferramenta é baseado, tal como o iGesture, na gestão de eventos do Java que gere os diferentes dispositivos de input. Tanto a Pointer Toolkit como o iGesture suportam dispositivos de input semelhantes, como é o caso do rato de computador ou caneta digital. Apesar de ambas serem baseadas em eventos, a Pointer Toolkit baseia-se nos eventos de carregar, largar, passar por cima, arrastar, etc. Existe também a ferramenta *"Mouse Gesture Recognition with Hidden Markov Model"*, que será discutida na secção seguinte, que reconhece gestos desenhados através de modelos ocultos de Markov (HMM) e acabou por ser a escolha mais viável para realizar a interpretação de gestos desenhados na plataforma Model-By-Voice

3.11 *Mouse gesture recognition with Hidden Markov Model*

Os gestos podem ser compreendidos como um problema de reconhecimento de padrões. Uma vez que um gesto consiste num movimento contínuo no tempo, um *Hidden Markov model* (HMM) é uma boa ferramenta para reconhecer os mesmos [55].

Um projeto académico realizado no âmbito do reconhecimento de gestos desenhados pelo rato do computador utilizando HMM dinâmicos [55], revelou-se muito útil e eficaz no reconhecimentos deste tipo de gestos. Uma vez também ter sido desenvolvido em Java, previa-se que uma possível integração na plataforma Model-By-Voice seria potencialmente muito boa. O projeto partiu de um outro projeto académico [80] que já expunha o potencial da utilização de HMM para reconhecer gestos, embora encontrasse alguns aspetos menos positivos. Aspetos estes como não existir a noção de protótipo, não permitindo sumarizar os gestos, o que obriga a utilizar o nome do HMM em vez do gesto em si. Também se verificou não ser possível encontrar todas as características distintivas dos modelos, o que levou a encontrar semelhanças nos mesmos e encontrar características distintivas, resultando na distinção dos gestos. Este projeto sugeria que trabalhos futuros tentassem modificar os HMM de forma a ser possível fazer mais facilmente o reconhecimento de todas as características importantes dos modelos.

O projeto [55] revela ter uma abordagem que soluciona os problemas descritos no projeto anterior, distinguindo as características dos gestos como sendo compostas por:

- Distância Normalizada pelo ponto atual relativamente aocCentro de gravidade de todos os pontos.
- Ângulo do ponto atual em relação ao centro de gravidade.
- Ângulo do ponto atual em relação ao ponto inicial.
- Ângulo do ponto atual em relação ao ponto final.
- Ângulo do ponto atual em relação ao ponto Min-X e Min-Y.
- Ângulo do ponto atual em relação ao ponto Min-X e Max-Y.
- Ângulo do ponto atual em relação ao ponto Max-X e Min-Y.
- Ângulo do ponto atual em relação ao ponto Max-X e Max-Y.

Os algoritmos utilizados, como *K-means*, *Vector Quantization*, *CodeBook Generation*, *Baum-Welch training*, *Viterbi Decoding*, foram retirados de um outro projeto do mesmo autor [81]. A implementação do projeto consiste em utilizar um *listener* para o rato do computador e um *listener* para o seu movimento. Estes *listeners* permitem recolher a informação dos pontos que estão a ser desenhados e quando estes são desenhados. A informação do tempo associada a cada *pixel* é necessária para o cálculo da velocidade e para ser possível reproduzir posteriormente o gesto desenhado. O utilizador pode realizar

as operações de Verificar/Reconhecer, Adicionar novos gestos ou nova informação de treino, Treinar o *CodeBook* e os HMM através da interface da ferramenta.

Esta implementação resultou numa ferramenta de reconhecimento de gestos com as seguintes características:

- Invariante com o tamanho do desenho do gesto entre treino e teste.
- Alta *performance*: >95%.
- Testada com mais de trinta gestos (a maior parte compostos por linhas retas e algumas linhas curvas com 'S').
- Treinada com 1040 dados de treino.
- Quantificação de ângulos enquanto compõe recursos.
- Sensível à orientação.

O autor estimou existir uma probabilidade de transição diferente de zero para obter sucesso no que a ferramenta propunha. O que verificou ser verdade após validar a mesma.

Em suma, esta é uma ferramenta muito útil para reconhecer gestos 2D e torna-se muito relevante para a reformulação do modelo de interação da plataforma Model-By-Voice, uma vez que suporta uma fácil integração em Java e a integração do rato do computador como mecanismo de *input*.

3.12 Conclusões

Todos os trabalhos descritos ao longo deste capítulo pretendem melhorar a experiência de interação com os seus utilizadores, principalmente os que possuem algum tipo de dificuldade física, nomeadamente visual, no contexto da modelação. Contudo, são muito poucos os que revelam progressos na modelação apenas através de voz, sons não-vocais, e ainda aceitando gestos como *input*. Verifica-se também que quase nenhum permite personalizar o processo de interação, que foi um foco muito grande na solução de reformulação do modelo de interação da plataforma Model-By-Voice. Também se pode verificar que nenhum dos trabalhos apresentados permite ao utilizador escolher o paradigma de navegação (tabela ou grafo) com base nas suas necessidades e conhecimentos, conceito desenvolvido na solução proposta, visando tornar o processo de interação o mais personalizável possível. Nenhum dos projetos apresentados, no contexto de modelação, permite personalizar o sintetizador de voz, permitindo ao utilizador escolher a velocidade, timbre e volume da voz. Também se pode observar que nenhum destes projetos possui a noção de "Utilizador", permitindo guardar as preferências dos utilizadores para que estas estejam disponíveis quando estes iniciarem sessão nas próximas utilizações.

Na Tabela 3.1 são resumidas e comparadas as diferentes características dos trabalhos e tecnologias descritos neste capítulo. Sendo que as características apresentadas são as que

a solução apresentada no capítulo seguinte contempla, sendo possível observar quais os projetos/tecnologias que já apresentam algumas dessas características ou as características que ainda não são observáveis em nenhum projeto/tecnologia já desenvolvido. Como se pode observar na tabela ainda não existe, de acordo com o meu conhecimento, uma solução de modelação que integre a noção de utilizador, guardando as suas preferências, e que permita escolher o paradigma de navegação. Surge assim a oportunidade de melhorar o estado da arte com a solução proposta nesta dissertação e as características que esta possui.

Tabela 3.1: Comparação entre as características dos trabalhos relacionados

	TeDub	Melhorar a Produtividade do Utilizador em Ferramentas de Modelação Através de Workflows de Modelação Específicos	Ferramenta Kevin	Tecnologias de reconhecimento e sintetização de voz (Sphinx-4, Google Cloud Speech, FreeTTS)	iGesture	Abordagem Multi-Paradigmática para Integrar Gestos e Som numa Ferramenta de Modelação	Proposta de Modelação Orientada por Voz	VoiceToModel	Musicg	Sonic	Mouse Gesture Recognition with Hidden Markov Model
Permite Modelar DSLs/Diagramas		X	X	X	X	X	X	X			
Processo de interação personalizável		X									
Permite reconhecer sons codificados				X		X			X		
Possui reconhecimento de voz				X		X	X	X			
Possui sintetização de voz	X			X		X	X	X			
Suporta linguagem natural				X			X	X			
Permite interagir através de gestos					X	X					X
Necessário o utilizador ter conhecimento de noções espaciais 2D		X	X								X
Permite acelerar a voz de ficheiros de áudio										X	
Possui noção de utilizador e guarda as suas preferências para sessões futuras											
Paradigma de interação personalizáveis											

Capítulo 4

Introdução de requisitos da ferramenta

Neste capítulo será descrito em detalhe como foi construída a ferramenta que se pretende que solucione os problemas levantados no capítulo da introdução, nomeadamente o modelo de interação anterior da plataforma Model-By-Voice ser pouco prático, fixo e pouco versátil. Serão também apresentadas as tecnologias que foram utilizadas na construção da solução proposta.

4.1 Introdução

A solução proposta pretendia resolver o problema do modelo de interação antigo da plataforma Model-By-Voice ser fixo, pouco prático e pouco flexível. Esta consistiu em gerar um editor interativo que, para além do paradigma de navegação por grafo, inclui um paradigma de navegação tabelar. Este editor é personalizado para cada utilizador, a partir do protocolo de interação definido pelo SLE, que é adaptado para diferentes níveis de conhecimento e dificuldades. Neste protocolo são definidos gestos e sons não-vocais que auxiliam tanto o utilizador nas suas dificuldades e tornam mais prática a modelação da DSL que será utilizada. É também no protocolo de interação que são definidos os comandos que o utilizador poderá utilizar na ferramenta. Com esta funcionalidade, pretende-se que uma equipa de modelação possa definir permissões para cada utilizador. É possível agora dar a um determinado utilizador permissão apenas para atualizar diagramas e não poder criar novos, ou qualquer outra combinação das propriedades que constituem a DSL de geração de protocolos de interação discutida mais adiante neste capítulo.

A solução sobre a qual esta dissertação incide pretende ser uma coleção de configurações de navegações alternativas a serem embutidas nos editores de modelos gerados a partir do metamodelo de uma determinada linguagem. A mesma é responsável pela interação com o utilizadores visando ser inclusiva, isto é, está preparada para ser utilizada por

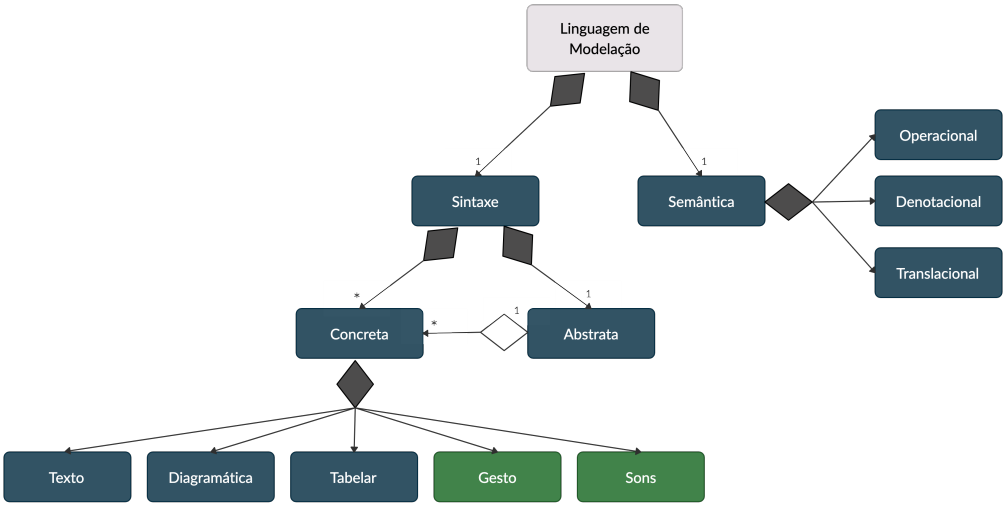


Figura 4.1: Diagrama representativo da sintaxe concreta, sintaxe abstrata e semântica de linguagens de modelação.

utilizadores com diferentes níveis de dificuldades visuais, ou mesmo totalmente invisuais. As estratégias de diálogo deste editor consistem maioritariamente em diálogo sonoro, sendo este repartido em voz e sons codificados, como sons, palmas ou ficheiros de áudio contendo sons não-vocais, e também em gestos 2D. Os utilizadores podem ainda escolher o paradigma em que pretendem modelar, tabela ou grafo, personalizar o sintetizador de voz e guardar as suas preferências num perfil próprio.

No diagrama ilustrado na Figura 4.1 estão descritas as melhorias em relação à sintaxe concreta das plataformas de modelação que de momento consistem em abordagens de modelação diagramáticas, textuais e também tabelar no caso de ferramentas como a Sirius [72]. O projeto desta dissertação pretende trazer a novidade de permitir modelação através de gestos desenhados e sons tornando estes elementos da sintaxe concreta das linguagens.

No diagrama de *features* representado na Figura 4.2 podem-se observar as funcionalidades do novo modelo de interação da plataforma Model-By-Voice.

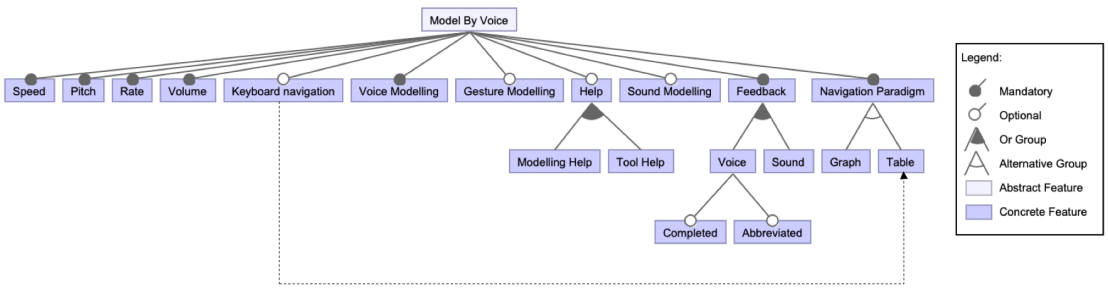


Figura 4.2: Diagrama de *features* que ilustra todas as funcionalidades do novo modelo de interação da plataforma Model-By-Voice.

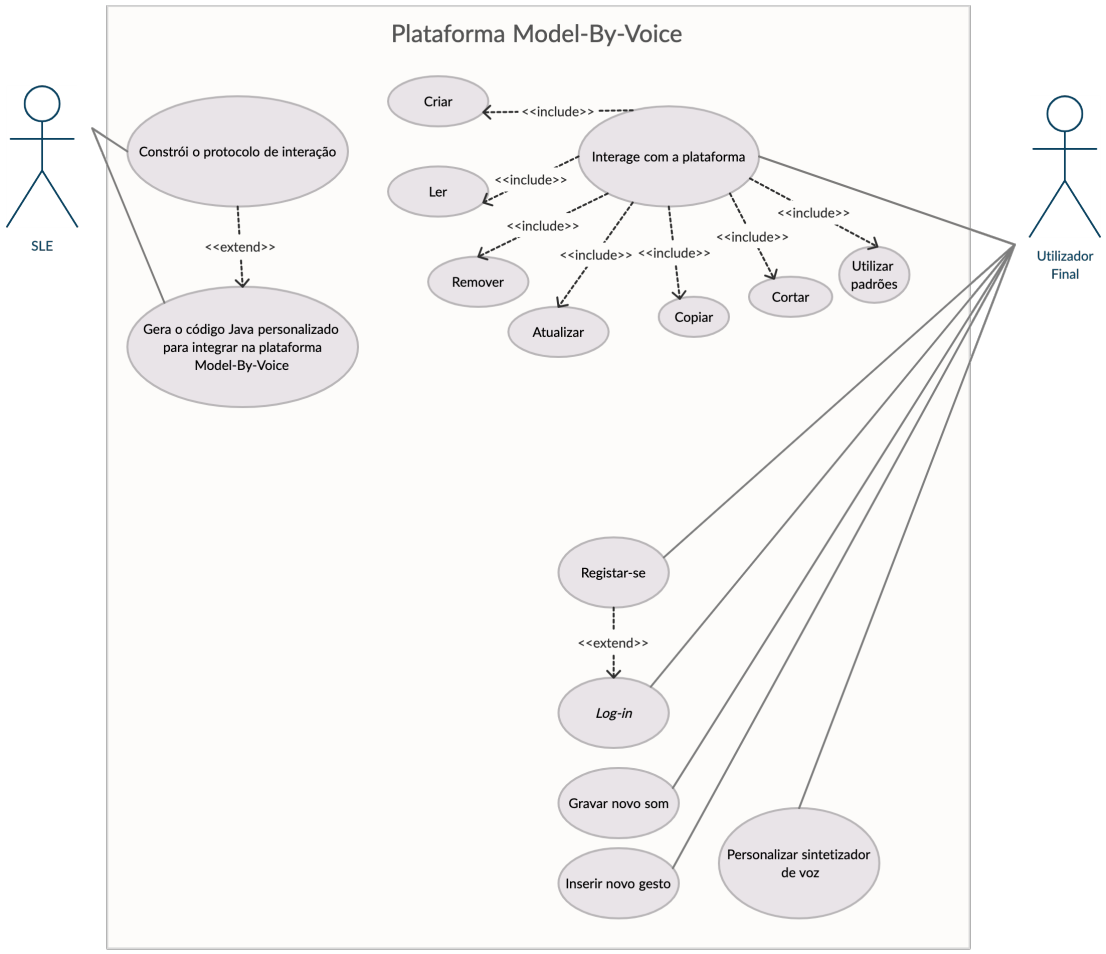


Figura 4.3: Diagrama de casos de uso que ilustra as relações do SLE e do utilizador final com as funcionalidades do novo modelo de interação da plataforma Model-By-Voice.

No diagrama de casos de uso da Figura 4.3 é possível observar as funcionalidades de edição oferecidas pelo novo modelo de interação da plataforma, e qual o papel do SLE e do utilizador final nexte contexto.

A personalização do protocolo de interação com o utilizador é feita em diferentes fases da geração do mesmo, disponibilizando ou personalizando as *features* conforme as características do utilizador. Quanto ao *feedback* fornecido pela plataforma, este será fornecido em forma de voz ou som não-vocal. Para cada mensagem de interação o SLE define no metamodelo do protocolo de interação como pretende que o *feedback* seja realizado para cada mensagem. Poderá então atribuir uma frase personalizada ou um som não-vocal a cada momento de interação, e caso tal não aconteça, então a mensagem de interação será entregue por sintetização de voz com a frase *default*.

O nível de conhecimento de modelação e de dificuldades do utilizador é também definido pelo SLE no metamodelo do protocolo de interação e irá influenciar o comportamento da plataforma em certas funcionalidades, tais como apresentação de mais ou

menos descrições ou uma maior ajuda para casos de mais dificuldades e menor conhecimento.

Os dados do metamodelo do protocolo de interação referidos acima, serão transformados em código java para ser integrado na plataforma de edição. A distribuição de funcionalidades pelos níveis de conhecimentos de modelação e dificuldades dos utilizadores pode ser descrita da seguinte forma:

- Nível de conhecimento 0: Modelação por voz, ajuda a nível de modelação, ajuda a nível da plataforma, *feedback* de voz completo.
- Nível de conhecimento 1: Modelação por voz, *feedback* de voz abreviado.
- Nível de dificuldades visuais 0: Modelação por voz, navegação através de teclas no teclado no âmbito do paradigma de tabela, *feedback* de voz abreviado.
- Nível de dificuldades visuais 1: Modelação por voz, ajuda a nível da plataforma, *feedback* de voz completo.

Quando o utilizador se regista na plataforma, de forma a que todas as suas preferências sejam guardadas para serem utilizadas em sessões posteriores, é-lhe perguntado também como este pretende que se comportem certas funcionalidades. Primeiramente é perguntado como pretende personalizar o sintetizador de voz. O utilizador poderá optar pelas classificações "baixo", "normal" e "alto" para personalizar a velocidade, timbre, taxa de amostragem e volume do sintetizador conforme as suas preferências pessoais. Esta personalização é feita através da biblioteca Sonic [74], após guardar o que é "dito" pelo sintetizador FreeTTS num ficheiro de áudio WAV e de seguida transformar esse ficheiro, conforme as classificações escolhidas pelo utilizador para as diversas características do sintetizador, reproduzindo pela plataforma o som com essas mesmas características.

Posteriormente é perguntado ao utilizador se prefere o paradigma de grafo ou tabela, se quer ativar a modelação por gestos e se pretende ativar a modelação por som.

4.2 Modelação da interação

A arquitetura da solução proposta consiste numa ferramenta desenvolvida na linguagem de programação Java que reformulou a plataforma Model-By-Voice, tornando o modelo de interação mais prático, permitindo dar uma maior resposta a eventuais problemas de usabilidade dos utilizadores. A ferramenta permite ao utilizador realizar *queries* para obter informações sobre os modelos e também realizar as operações básicas CRUD, bem como operações como copiar, colar, cortar e utilizar padrões para facilitar a criação de novos elementos.

O antigo modelo de interação da plataforma Model-By-Voice está descrito no diagrama de atividades da Figura 4.4 [47].

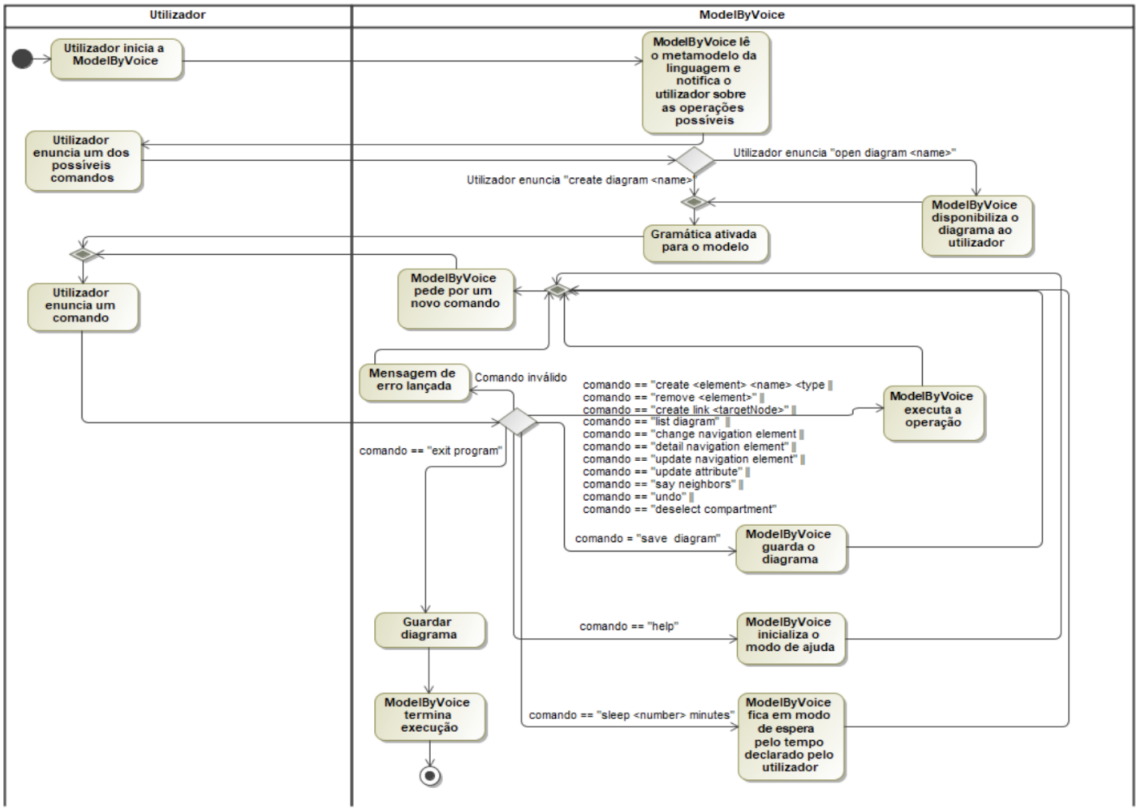


Figura 4.4: Diagrama de Atividades que representa o antigo modelo de interação da plataforma Model-By-Voice [Retirado da dissertação [47]].

O primeiro passo para a reformulação do atual modelo de interação foi utilizar uma abordagem de *sequence diagram*, em vez de *activity diagram*, para representar o modelo de interação. Esta mudança, mesmo sendo súbtil, é uma representação mais fidedigna da interação que se espera do modelo da solução proposta, uma vez que os *sequence diagram* capturam a interação entre objetos. Neste caso, entre o utilizador e a plataforma, num contexto de colaboração, e também são desenhados com o conceito temporal em mente, ou seja, as interações são representadas pela sua ordem cronológica [68]. Os *activity diagrams* são mais focados para representar aspetos dinâmicos do sistema e não são tão focados na interação [2]. A interação ser descrita em *sequence diagrams* torna-se clara quando observando o modelo de interação de alguns trabalhos relacionados como [3], em que a síntaxe abstrata da interação pode ser descrita através de uma sequência de ações.

O protocolo de diálogo da solução proposta é descrito por um *sequence diagram*. Criou-se então como parte da arquitetura da solução, uma DSL semelhante aos sequence diagrams ou a *activity diagrams* simplificados em que se gera um código semelhante ao antigo modelo de interação da plataforma Model-By-Voice, através de transformações Model-To-Code recorrendo à linguagem EGL, com alterações de forma a estar adaptado ao utilizador final. Esta DSL, após ser modelada por um SLE, irá produzir um protocolo

de diálogo com base nas características do utilizador, como o nível de conhecimento e dificuldades físicas, nomeadamente visuais. Também no metamodelo desta DSL é possível definir gestos e sons específicos adequados para certos aspetos da mesma. Assim, agora existe uma multi-geração de produtos em que para diferentes linguagens, utilizadas por diferentes utilizadores, existe um produto novo personalizado com um novo protocolo de diálogo como descrito na Figura 4.5.

Esta abordagem representa uma melhoria muito significativa no antigo modelo de interação da plataforma Model-By-Voice, uma vez que no modelo fixo antigo, o protocolo de diálogo é sempre o mesmo independentemente dos conhecimentos e dificuldades do utilizador. A modelação é fixa não tendo em conta aspetos chave da DSL a ser modelada, o que pode levar muito facilmente a dificuldades na utilização e à execução de comandos de voz morosos para criar componentes do diagrama que poderiam ser abreviados. Seria muito útil para o utilizador, caso fosse gerado um protocolo de diálogo que desse *feedback* através de sons não-vocais personalizados e permitisse executar ações através de gestos tendo em conta notações da linguagem. Utilizando o exemplo dos *Class Diagrams*, se o utilizador estiver a criar um diagrama deste tipo, no antigo modelo de interação, quando o utilizador recorre ao comando *"Create Link"* para criar uma ligação no diagrama, este terá de enunciar o comando através de um comando de voz. No novo protocolo de interação, devido à personalização da interação através de gestos desenhados, por exemplo, é possível definir que o comando *"Create Link"*, nos *Class Diagrams*, será definido por desenhar uma linha reta. Assim, o processo de interação torna-se mais prático, menos moroso e mais intuitivo para o utilizador, uma vez que para certas ações possíveis de realizar em diferentes diagramas, podem ser escolhidos diferentes gestos e sons codificados. Estes estão intuitivamente ligados a essas mesmas ações, como por exemplo, uma linha reta para executar o comando de criar uma ligação. Com um protocolo de interação ajustado

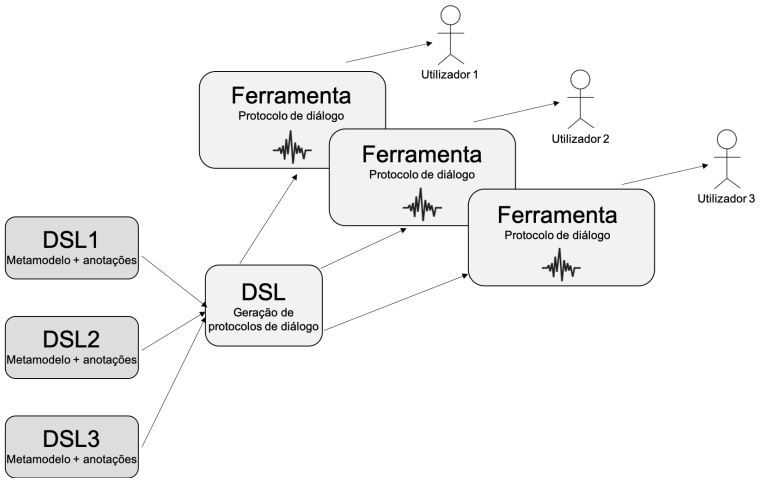


Figura 4.5: Geração de protocolos de diálogo na solução proposta para o novo modelo de interação da plataforma Model-By-Voice.

ao nível de conhecimento e dificuldades do utilizador, também é possível automatizar algumas ações e não disponibilizar um protocolo de interação destinado a utilizadores com mais ou menos dificuldades. No modelo antigo da plataforma, que é fixo, pode levar a um excesso de cuidados caso o utilizador não tenha tantas dificuldades como as previstas pelo modelo ou a uma falta de capacidade do utilizador de utilizar a plataforma, caso este tenha mais dificuldades do que as previstas. Esta geração de protocolos personalizados do novo modelo visa corresponder às dificuldades e conhecimentos daquele utilizador em específico, de forma a que este tenha a melhor experiência de interação com a plataforma possível.

A DSL construída tem uma abordagem semelhante à DSL de *workflows* apresentada e discutida no capítulo do Trabalho relacionado [34]. Esta visiona modelar o processo de interação com o utilizador no ambiente de modelação e automatizar as tarefas comuns que possam ser automatizadas e ser personalizável para tarefas que não podem ser automatizadas, em tempo de execução. A DSL de *workflows* tornou-se então um ponto de partida interessante para a construção da DSL da solução proposta uma vez que os protocolos de interação são semelhantes a *workflows*. Contudo, em vez de ser o utilizador a escolher qual o *workflow* a utilizar como acontece na DSL de *workflows*, na DSL de protocolos de interação que integra a arquitetura da solução proposta, o protocolo de interação, após ser modelado por um SLE, gera automaticamente um editor de modelação conforme os níveis de conhecimento e dificuldades do utilizador. Com a solução atual é ainda possível definir gestos e sons não-vocais que tornem a interação mais intuitiva para a DSL que o utilizador irá modelar. Desta forma, o modelo de interação gerado será o mais adequado possível para o utilizador.

Na Figura 4.6 é apresentado o metamodelo da DSL de protocolos de interação com a plataforma, onde se pode observar as atividades possíveis de realizar na plataforma e como as mesmas podem ser representadas.

4.3 Mecanismos de interação

No novo modelo de interação, após inicializar a plataforma, o utilizador poderá optar pela operação a executar, como criar um novo diagrama ou escolher atualizar outro, por exemplo. O utilizador poderá executar a partir deste ponto as operações básicas CRUD, assim como operações simples de copiar, colar, cortar e utilizar padrões para criação de elementos no diagrama, através de voz, sons não-vocais ou gestos e no paradigma de navegação escolhido (grafo ou tabela).

A arquitetura idealizada prevê que o SLE defina um protocolo de interação específico para o utilizador e onde podem ser definidos sons não-vocais e gestos para substituir comandos e o *feedback* dado na plataforma. Posteriormente, a ferramenta terá um papel semelhante ao EuGENIA [30] da família Epsilon [28], visto que irá converter o metamodelo da DSL a ser modelada no metamodelo utilizado atualmente pelo Model-By-Voice, que consiste apenas em nós e ligações. Esta abordagem de todas as DSLs de *input* serem

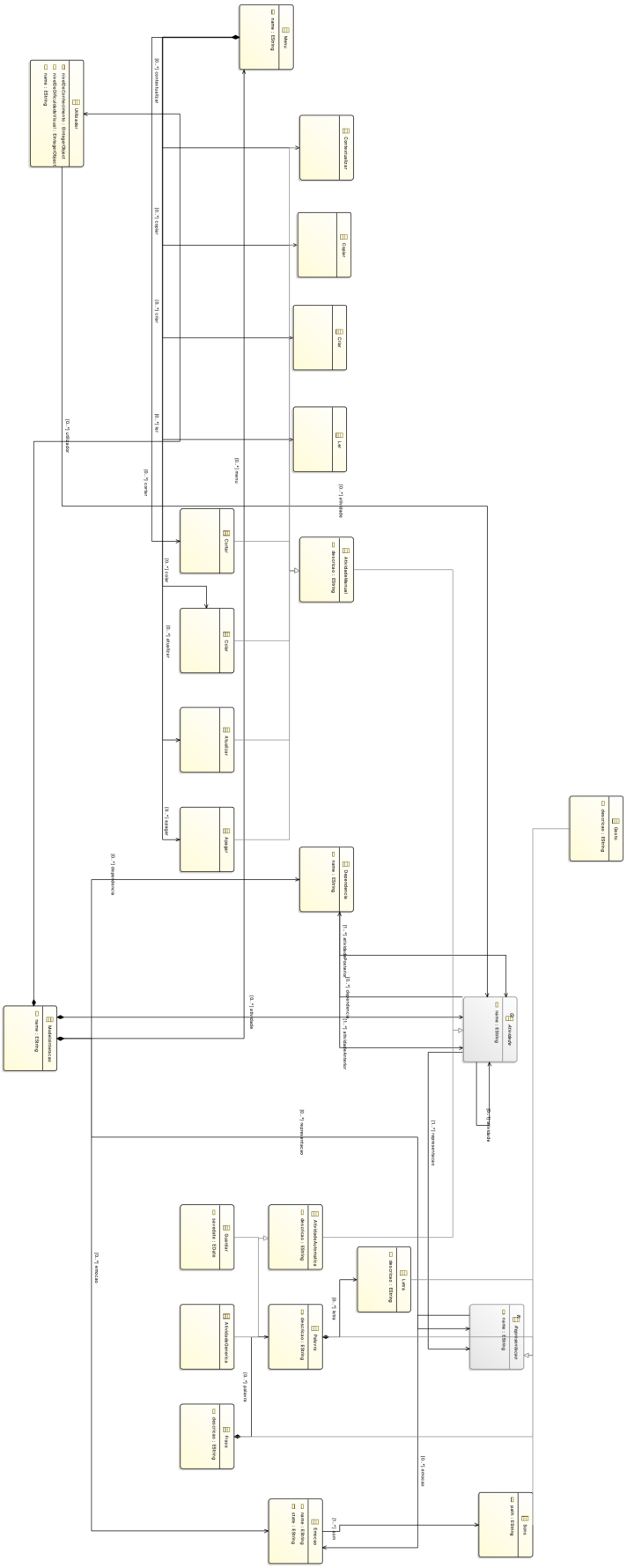


Figura 4.6: Metamodelo da DSL de protocolos de interação para plataforma Model-By-Voice.

representadas por nós e ligações é crucial para o propósito da ferramenta, uma vez que não obriga o utilizador a ter qualquer tipo de noção espacial a duas dimensões, isto é o utilizador apenas precisa de ter o conhecimento que existem pontos (nós) e arestas (ligações) que ligam esses pontos entre si formando programas na DSL. Desta forma, não é necessário que o utilizador tenha noções como as direções para navegar nos modelos construídos, o que é essencial para garantir que todos os utilizadores consigam navegar facilmente nos modelos.

4.3.1 Paradigma de tabela

A nova versão do modelo de interação também permite ao utilizador escolher interagir com a plataforma através de um mecanismo de tabelas semelhante à ferramenta "Kevin" [11]. No entanto, em vez de recorrer a um aparelho externo, utiliza apenas a modelação por voz para realizar uma simulação semelhante através de uma DSL desenvolvida em ambiente Epsilon [28]. Optou-se por dar ao utilizador a opção de escolher o paradigma de navegação, navegar em grafos (nós e ligações) ou tabelas, pois caso este não tenha limitações visuais, terá noções espaciais básicas 2D presentes. Assim, o utilizador poderá preferir a abordagem de tabelas, para realizar as operações CRUD e navegar em modelos maiores com complexidade elevada de forma mais simplificada, que lhe poderá proporcionar uma melhor experiência de interação. Na Figura 4.7 encontra-se o metamodelo que representa o paradigma de tabela na plataforma.

Os objetos do tipo tabela são constituídos por células, que podem ser nós ou ligações, que são identificadas pelas suas coordenadas na tabela. A disposição das células na tabela é feita com base nos mecanismos da ferramenta "Kevin", descrita no capítulo do Trabalho relacionado. Os nós são colocados na diagonal principal da tabela à medida que são criados e as ligações são colocadas na célula com o número de linha igual à linha do nó de saída e o número de coluna igual à coluna do nó de entrada. Para navegar nas coordenadas da tabela podem ser utilizados comandos de voz como *"go to"*, *"go up"*, *"go down"*, *"go left"*, *"go right"*, *"go next node"*, *"go previous node"* ou também as setas do teclado (cima, baixo, esquerda, direita). As teclas de barra de espaço poderão também ser utilizadas para avançar para o nó seguinte na tabela, e a tecla de *backspace* para navegar para o nó anterior. Este sistema de navegação através das teclas do teclado acelera bastante o processo de navegação e representa uma ajuda para os utilizadores se situarem melhor espacialmente na mesma, e , consequentemente, melhora a produtividade do utilizador.

Para navegar na tabela, de forma a não comprometer a estrutura base da plataforma, foi necessário garantir que o utilizador não fica situado em células que não contêm nós, o que representaria um problema por não ficar definido um *current node*, que é a base para realizar operações na plataforma. Sempre que na célula para que o utilizador navega se encontra uma ligação, o nó de saída da mesma é declarado *current node*. Assim, as ligações funcionam como uma forma muito rápida e prática de navegar ente nós, o que se espera que resulte num aumento da produtividade do utilizador. Caso não existam

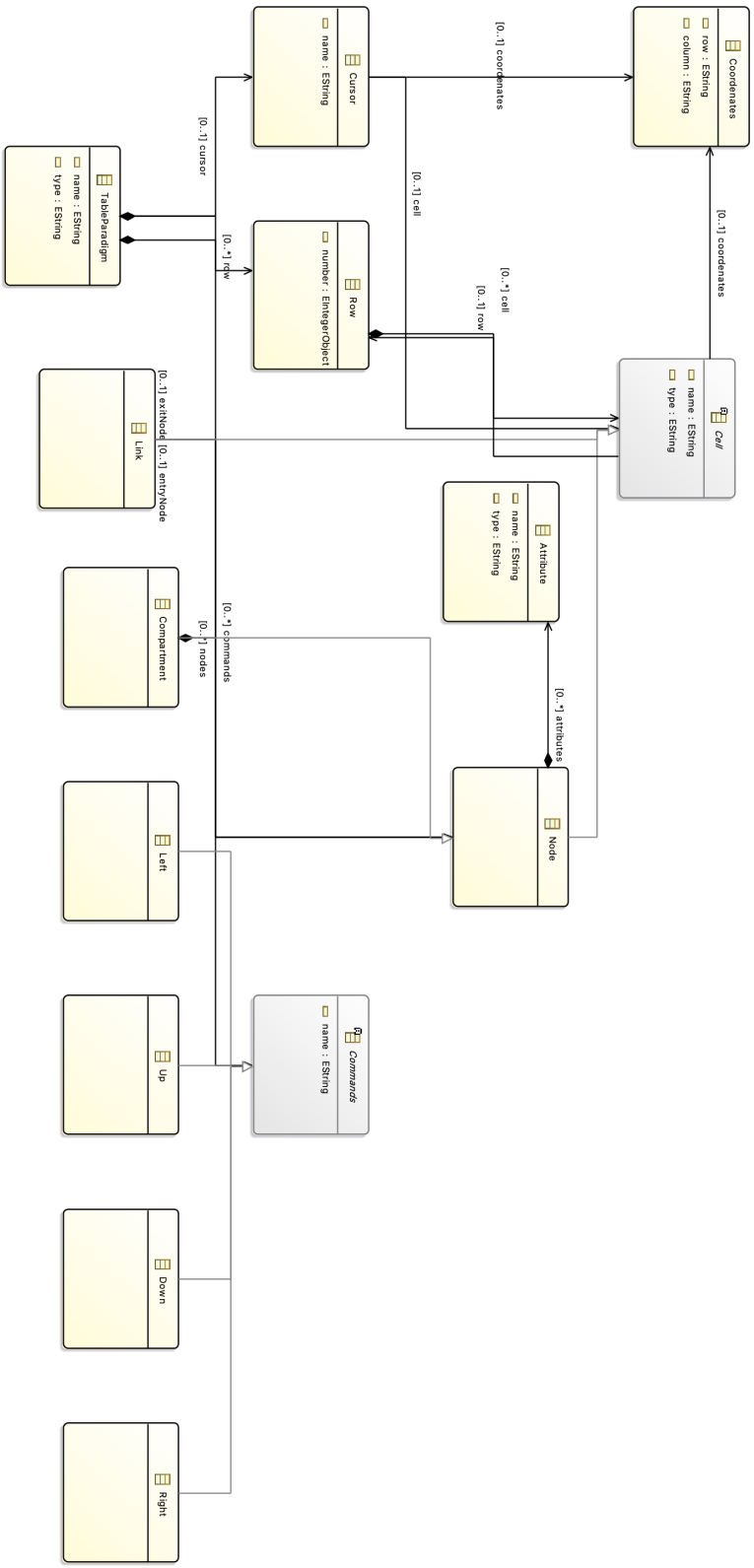


Figura 4.7: Metamodelo do paradigma de tabela na plataforma Model-By-Voice.

ligações na célula adjacente à que o utilizador se encontrava (os nós apenas existem na diagonal principal da tabela), o programa irá averiguar se existe alguma ligação na direção escolhida. Caso seja encontrada uma ligação nessa linha ou coluna da tabela, o nó de saída da mesma será definido como *current node*, caso contrário, o utilizador será informado que nada foi encontrado e então a célula onde se encontrava primeiramente continua a ser o *current node*. O sintetizador de voz vai dando sempre *feedback* das coordenadas onde o utilizador se situa quando este navega na tabela.

No paradigma de tabela é também possível ordenar os nós por ordem alfabética na diagonal principal em vez de ser pela ordem de inserção. Neste caso, todas as ligações associadas mudam também para células diferentes de forma a que as ligações entre os nós se mantenham intactas.

4.3.2 Reconhecimento e sintetização de voz

Na componente de reconhecimento e sintetização de voz esperava-se manter as ferramentas que já eram utilizadas no modelo de interação da plataforma Model-By-Voice. Nomeadamente a Sphinx-4 e Google Cloud Speech para o reconhecimento de voz e FreeTTS para a sintetização de voz. Estas previam-se ser todas mantidas, dado a não existência no mercado de tecnologias semelhantes com características superiores e disponibilizadas de forma gratuita. Contudo, a dada altura no desenvolvimento do protótipo, a comunicação entre a plataforma Model-By-Voice e a API da Google Cloud Speech não se verificava consistente. Desta forma, optou-se por manter apenas a ferramenta Sphinx-4 para realizar o reconhecimento de voz, aceitando a limitação tecnológica que esta não reconhece palavras que não estão presentes na gramática da plataformas.

4.3.3 Reconhecimento e sintetização de sons não-vocais

Outra melhoria que se efetuou no novo modelo de interação é possibilitar aos utilizadores recorrerem a sons curtos codificados, como sinais, palmas, ou mesmo assobios, em vez de longos comandos de voz, e o *feedback* da ferramenta também ser dado com este tipo de sons, em vez das longas respostas a que recorria anteriormente. Estes sons podem estar também associados a emoções. Assim, caso o SLE assim o defina no metamodelo do protocolo de interação, quando um som não-vocal é reproduzido, poderá também ser reproduzido o som correspondente à emoção associada a esse som. Por exemplo, quando se remove um nó de um diagrama, se o SLE definiu que associado à ação de remover nós estará associada a emoção de tristeza, o som que este definiu para esta emoção será reproduzido sempre que esta ação seja executada.

Estes sons não-vocais podem ser incluídos *à priori* no metamodelo do protocolo de interação pelo SLE, e também pelo próprio utilizador enquanto utiliza a ferramenta, caso este sinta que um certo som não-vocal o auxiliaria na execução de uma ação na plataforma, e que o SLE não o definiu no protocolo de interação. Para tal, o utilizador deve gravar o som através de um comando na plataforma e indicar a qual ação da plataforma pretende

associar o som. Posteriormente, quando este reproduzir o som, a plataforma já executará a ação pretendida. Considera-se muito importante que o utilizador tenha a possibilidade de adicionar novos sons não-vocais, que não foram definidos pelo SLE, pois embora existam ações que sejam previsíveis de associar a certos sons, dada alguma dificuldade visual por exemplo, poderão existir diversas ações que um dado utilizador considere relevante associar a um certo som, devido às suas características únicas.

No antigo modelo de interação, o utilizador perdia bastante tempo a ditar os comandos das operações que pretendia realizar. O tempo era posteriormente aumentado com o *feedback* igualmente longo que a plataforma lhe transmitia. Ao utilizar sons codificados para esta interação, tornamos o processo de interação utilizador-plataforma mais prático e menos moroso. Para que seja possível utilizar sons codificados, como palmas e assobios, pensou-se em utilizar as atuais ferramentas de reconhecimento de voz da plataforma Model-By-Voice, criando uma gramática com estes sons codificados na ferramenta Sphinx-4 e treinar a mesma de forma a reconhecê-los [82] [58] [1]. Uma vez que o *software* tem a habilidade de "ouvir", este poderá também reagir a este tipo de sons não-vocais [15]. Contudo, verificou-se que o treino da ferramenta Sphinx-4 para descodificar este tipo de sons, seria de uma complexidade não realizável no âmbito desta dissertação. Então optou-se por utilizar as ferramentas da biblioteca Musicg [56], uma vez que estas permitem um reconhecimento fidedigno dos sons não-vocais e permitiam uma integração simples na plataforma.

O utilizador poderá também personalizar o sintetizador de voz podendo escolher a velocidade, volume e timbre da voz do mesmo. Desta forma é possível acelerar a interação, escolhendo velocidades de voz maiores para utilizadores que já utilizam a plataforma há algum tempo e já conseguem reconhecer o que é dito pelo sintetizador a velocidades maiores. Esta personalização é feita através da biblioteca Sonic [74], como explicado anteriormente.

Quando é executada uma ação, e durante a navegação nos modelos, poderão também ser inseridos sons para assinalar ações executadas de forma correta ou errada. Caso o SLE tenha adicionado os mesmos no protocolo de interação, de forma a aplicação oferecer *feedback* ao utilizador sem ter de recorrer à sintetização de voz, o que acelerá bastante o processo de *feedback*.

4.3.4 Reconhecimento de gestos

O utilizador poderá também interagir com o editor através de gestos, desenhando os mesmos com o rato do computador. Pensou-se em utilizar a tecnologia disponibilizada pela ferramenta iGesture [40]. Planeava-se que o utilizador poderia desenhar, segundo os gestos codificados, na ferramenta iGesture, os comandos de navegação que pretendia efetuar no editor. Estes gestos deveriam ser previamente registados e treinados na ferramenta. O

[1] [58] retirada do *website Stack Overflow*

iGesture pode ser integrado em aplicações exteriores através do registo como "GestureEventListener" para um dado dispositivo de input. Assim, quando o utilizador desenha um gesto com um determinado dispositivo, todos os *listeners* registados são notificados. O *listener* passará então esta informação a um reconhecedor, que, depois de reconhecer o gesto, irá notificar os gestores de eventos que aqui estavam registados. Um gestor de eventos implementará a interface "GestureHandler" que especificará qual a ação que deverá ser executada após um certo gesto ser reconhecido [63]. Este processo é descrito de forma simplificada na Figura 4.8.

Contudo, o treino da ferramenta e integração na plataforma, não se verificaram tecnicamente realizáveis no âmbito desta dissertação, optando-se então por utilizar o projeto desenvolvido em âmbito académico [55]. Relativamente à utilização é muito semelhante à ferramenta iGesture, contudo a sua integração na plataforma Model-By-Voice revelou ser mais fácil. A utilização da própria ferramenta de reconhecimento é também mais prática, uma vez que esta já continha um elevado número de gestos treinados para ser reconhecidos através de HMM.

Desta forma, elaborando as mudanças necessárias à ferramenta [55] para que fosse integrada na plataforma Model-By-Voice e se adaptasse às necessidades dos utilizadores da mesma, foi possível integrar uma componente de reconhecimento de gestos a nível 2D. De forma a estender a ferramenta [55] ao caso extremo de interação com utilizadores invisuais ou com visibilidade reduzida, foram realizadas algumas alterações à interação com a própria ferramenta, tais como:

- Aumentar o tamanho do painel de desenho do gesto.
- Centrar o rato sempre num ponto do painel de desenho de gestos em que seja possível para o utilizador desenhar um gesto sem colocar o rato numa posição inicial.

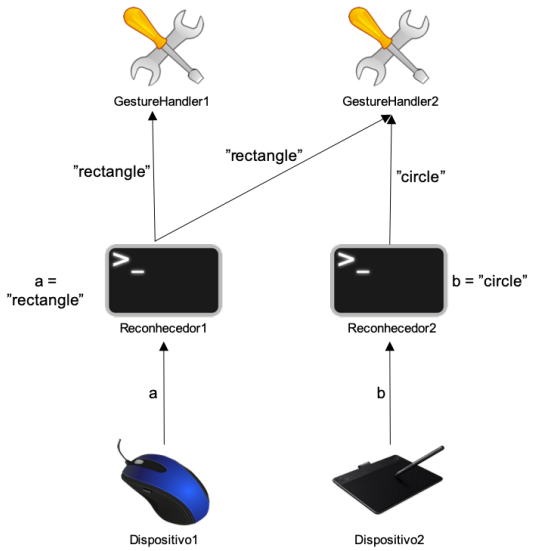


Figura 4.8: Arquitetura de reconhecimento da ferramenta iGesture.

- Automatização do processo de validação/reconhecimento do gesto quando o utilizador acaba de o desenhar.

Estas alterações são significativas quanto à usabilidade por parte de utilizadores que possuam algum nível de incapacidade visual, permitindo-lhes usufruir desta funcionalidade apenas desenhando o gesto. Os utilizadores não terão assim de se preocupar com questões de posicionamento do rato ou em carregar em botões para a plataforma reconhecer o gesto.

Na Figura 4.9 está ilustrado um exemplo de como a ferramenta [55] reconhece o gesto desenhado no painel, identificando o mesmo com os movimentos do rato que descrevem como foi desenhado, neste caso para baixo e para a esquerda.

A inclusão da possibilidade de reconhecer gestos 2D facilita muito o processo de executar as operações CRUD e o processo de navegação em diagramas na plataforma, especiamente nos que forem dotados de uma maior complexidade. Isto porque o moroso processo de descrição das ações pretendidas, que implica todo o processo de reconhecimento e sintetização de voz, mais o tempo demorado pela plataforma a dar *feedback* das operações que foram executadas ao utilizador, pode ser substituído pela possibilidade do utilizador descrever estas ações por simples gestos 2D codificados que serão associados a ações na plataforma Model-By-Voice em tempo de execução. Os gestos a ser utilizados podem também ser criados pelo utilizador, o que permite ao mesmo criar gestos 2D o mais intuítivos e práticos possíveis de acordo com a DSL em que irá trabalhar e as suas capacidades/dificuldades. Esta melhoria no atual modelo de interação da plataforma Model-By-Voice revela-se útil para pessoas invisuais, uma vez que é um processo mais rápido e prático do que o reconhecimento de voz. Para o utilizador comum é também uma melhoria, uma vez que permite navegar mais facilmente nos modelos e é uma forma rápida e interessante de modelar DSLs, até podendo ser, discutivelmente, mais prática do que modelar DSLs com um editor gráfico.

À semelhança dos sons não-vocais, os gestos desenhados poderão ser associados tanto pelo SLE no metamodelo do protocolo de interação, como pelo próprio utilizador durante

Figura 4.9: Ferramenta utilizada no reconhecimento de gestos na plataforma Model-By-Voice [55].

a utilização da plataforma, caso este considere que associar um gesto a uma ação irá auxiliar e tornar mais produtiva a sua utilização da ferramenta.

Esta funcionalidade será unidirecional uma vez que apenas o utilizador pode interagir com a plataforma através de gestos e não vice-versa, ao contrário do discurso e sons codificados em que a plataforma pode dar *feedback* ao utilizador através dos mesmos.

A usabilidade de gestos desenhados neste tipo de ferramentas revela-se útil e funcional até mesmo para pessoas visualmente incapacitadas, uma vez que estas conseguem descrever os gestos codificados de forma suficientemente precisa para que a ferramenta os reconheça [53]. Também para o utilizador comum, como já referido, trará benefícios, visto que este tipo de interação poupa bastante tempo em comparação com ter de realizar as operações de interação no editor gráfico ou por voz.

4.4 Objetivos da nova arquitetura

As melhorias propostas nesta solução tiveram como objetivo melhorar a experiência de interação do utilizador, seja ele invisual ou sem quaisquer problemas de visão, na plataforma Model-By-Voice. Prevê-se que tornem a interação com a mesma mais prática, eficaz e menos morosa, dado que o novo modelo de interação já não é fixo e possui novos mecanismos de interação, como sons codificados e gestos. A interação entre o utilizador e a plataforma Model-By-Voice, à semelhança da solução anterior, continuará a ser feita através da língua inglesa dado a sua universalidade, permitindo abranger um maior número de utilizadores. Na Figura 4.10 é apresentada uma representação visual da arquitetura da solução proposta.

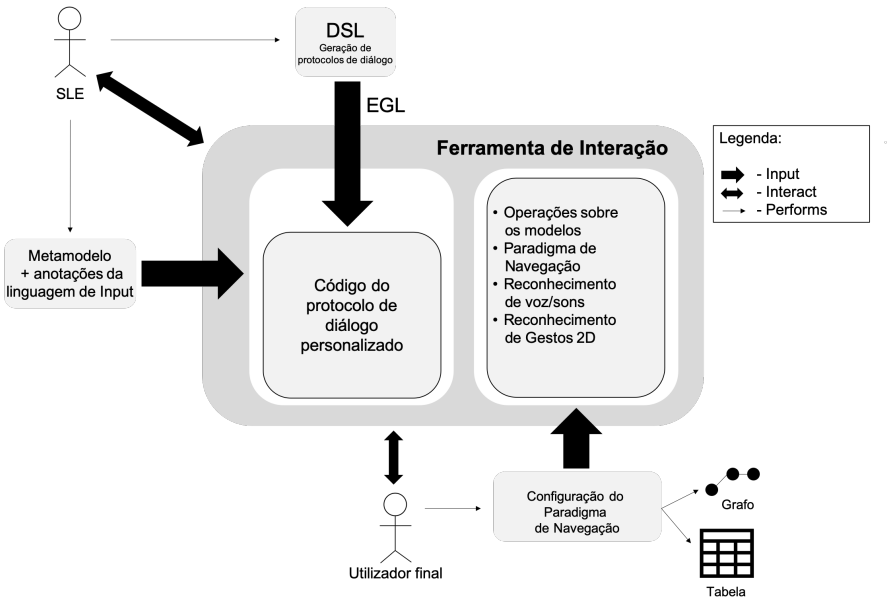


Figura 4.10: Arquitetura da solução proposta para o novo modelo de interação da plataforma Model-By-Voice.

Tabela 4.1: Comandos já existentes na plataforma.

Comandos:	**Operação:**
create diagram	Cria um diagrama na aplicação
create node	Cria um nó no diagrama criado
create attribute	Cria um atributo no diagrama criado
create compartment	Cria um compartimento no diagrama criado
create link	Cria uma ligação entre 2 nós no diagrama criado
remove navigation element	Remove o atual elemento de navegação
remove attribute	Remove o atributo pretendido
remove link	Remove a ligação pretendida
list diagram	Lista os elementos do diagrama
update navigation element	Atualiza o elemento de navegação
detail navigation element	Detalha o elemento de navegação
speak neighbours	Enuncia os vizinhos do elemento atual de navegação
sleep	Suspende a aplicação durante o tempo pretendido
deselect compartment	O compartimento deixa de estar selecionado
save diagram	Guarda o diagrama
open diagram	Carrega um diagrama guardado
Exit program	Fecha o programa
Undo	Desfaz a última operação
activate help mode	Ativa o modo de ajuda

4.5 Comandos e operações

Para além dos comandos já existentes na plataforma, foram adicionados novos comandos com o intuito de tornar mais eficiente a atividade de modelar. Para além disso, estes comandos também permitem incorporar os novos mecanismos de interação de sons não-vocais e gestos.

Na Tabela 4.1 encontram-se descritos os comandos, e respetivas operações, que já existiam na plataforma no antigo modelo de interação. Na Tabela 4.2 podem observar-se os novos comandos adicionados com o novo modelo de interação, completando o conjunto de comandos que neste momento estão disponíveis na plataforma.

4.6 Diagrama de atividades

Na Figura 4.11 é ilustrado através de um diagrama de atividades as possíveis operações a realizar na plataforma Model-By-Voice e como passou a ser feita a interação com os dois tipos de utilizadores (SLE e utilizador final). A figura apresenta todas as opções que o utilizador final tem à sua disposição na plataforma, assim como a dinâmica existente entre SLE, utilizador final e plataforma e como é esperado que decorra uma execução normal da ferramenta.

Este processo tem início com a elaboração do protocolo de interação por parte do SLE,

Tabela 4.2: Novos comandos integrados na plataforma.

Comandos:	**Operação:**
copy	Copia o nó pretendido e cola um novo nó com as mesmas características (é possível guardar o tipo do nó copiado como padrão)
cut	Corta o nó pretendido e cola um novo nó com as mesmas características (é possível guardar o tipo do nó copiado como padrão)
use pattern	Utiliza um padrão guardado para criar um novo nó com as características do padrão
go to	O utilizador é deslocado para a célula pretendida na tabela (quando o utilizador está a utilizar o paradigma de tabela)
go up	O utilizador é deslocado para a célula acima do elemento atual na tabela (quando o utilizador está a utilizar o paradigma de tabela)
go down	O utilizador é deslocado para a célula abaixo do elemento atual na tabela (quando o utilizador está a utilizar o paradigma de tabela)
go right	O utilizador é deslocado para a célula à direita do elemento atual na tabela (quando o utilizador está a utilizar o paradigma de tabela)
go left	O utilizador é deslocado para a célula à esquerda do elemento atual na tabela (quando o utilizador está a utilizar o paradigma de tabela)
go next node	O elemento de navegação passa a ser o próximo nó na tabela (quando o utilizador está a utilizar o paradigma de tabela)
go previous node	O elemento de navegação passa a ser o nó anterior na tabela (quando o utilizador está a utilizar o paradigma de tabela)
order alphabetically	Ordena alfabeticamente os nos da tabela preservando as ligações entre os mesmos

onde este irá definir o nível de dificuldade visual e conhecimentos de modelação do utilizador que utilizará o protocolo, os comandos que este estará autorizado a utilizar, como o *feedback* será dado na ferramenta (para cada frase de feedback pode ser definida uma nova frase ou um som não-vocal) e por fim os sons não-vocais e gestos que o utilizador poderá utilizar de raiz quando abrir a plataforma pela primeira vez.

Quando o utilizador inicia a plataforma pela primeira vez escolhe um nome de utilizador e personaliza o sintetizador de voz a seu gosto (escolhendo a velocidade, timbre, volume, etc). De seguida escolhe o paradigma de navegação em que pretende modelar (grafo ou tabela). Finalmente poderá utilizar os comandos que o SLE definiu que este utilizador poderia utilizar, criando assim o fluxo que se pode observar na Figura 4.11.

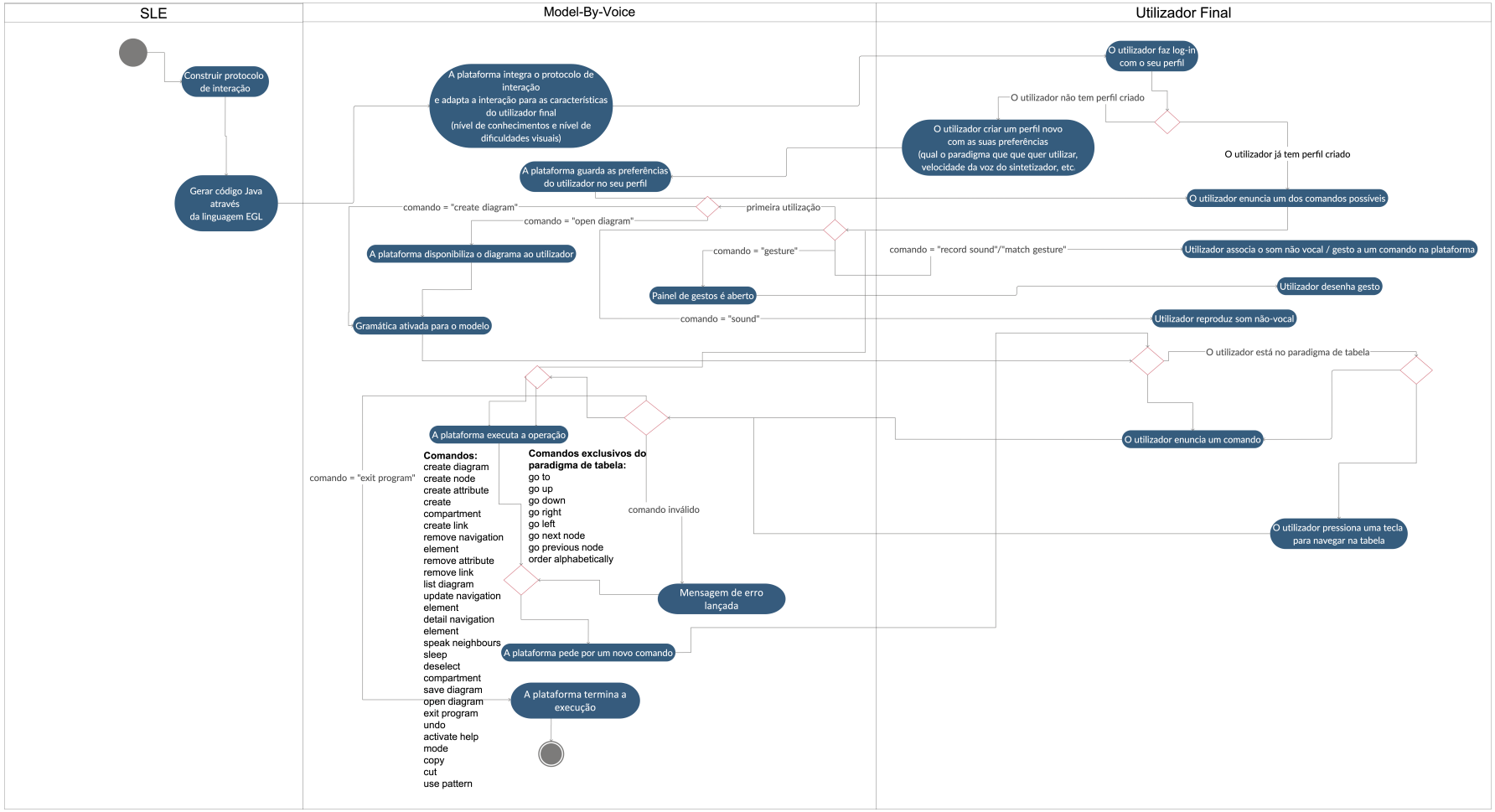


Figura 4.11: Diagrama de atividades que representa o fluxo de interação entre o SLE, utilizador e plataforma Model-By-Voice.

4.7 Protótipo

O protótipo desenvolvido no âmbito desta dissertação, cuja arquitetura foi descrita ao longo deste capítulo pode ser encontrado no seguinte repositório através do endereço eletrónico: Protótipo do novo modelo de interação da plataforma Model-By-Voice [24].

4.8 Tecnologia utilizada

4.8.1 Java

A ferramenta de interação foi desenvolvida na linguagem de programação Java, uma vez que se trata de uma linguagem de programação de propósito geral e visto que o primeiro protótipo com o antigo modelo de interação da plataforma Model-By-Voice estava desenvolvido na mesma. Assim, foi possível reaproveitar algumas componentes já desenvolvidas e focar no desenvolvimento de novas características sem necessidade de re-implementar métodos já feitos. É também a linguagem lecionada com mais ênfase no curso de Mestrado Integrado em Engenharia Informática no qual se insere a dissertação, revelando-se assim ser a escolha mais viável para o desenvolvimento da ferramenta.

4.8.2 Tecnologias de modelação do IDE Eclipse

4.8.2.1 Epsilon

De forma a ter sido possível construir a arquitetura descrita anteriormente foi necessário desenhar e implementar uma DSL que realizasse a geração de protocolos de diálogo personalizados com base no seu nível de conhecimento e as suas dificuldades físicas, nomeadamente visuais e até na DSL de *input* em que o utilizador irá trabalhar. Para desenhar e implementar esta DSL de geração de protocolos foi utilizada a ferramenta Epsilon [28] no ambiente de desenvolvimento Eclipse [19]. A Epsilon é uma família de linguagens e ferramentas para geração de código, transformação modelo-para-modelo, validação de modelos, migração, entre outros que trabalha com EMF, UML, Simulink, XML e outros tipos de modelos. É uma ferramenta que se foca no desenvolvimento de DSLs e disponibiliza um ambiente com todas as condições necessárias para desenhar, desenvolver e testar as mesmas.

À semelhança do antigo paradigma de navegação da plataforma Model-By-Voice, que funcionava à base de grafos, foi necessário construir um novo metamodelo para representar a nova opção de paradigma de navegação que será uma abordagem de tabelas. Este novo paradigma foi desenvolvido também no Epsilon e é semelhante ao antigo paradigma de navegação, que é baseado apenas em nós e ligações. No entanto, em vez de nós e ligações, este irá ser baseado em células e noções básicas de conceitos espaciais 2D como cima, baixo, direita e esquerda, permitindo realizar a navegação entre as células que constituem a tabela.

4.8.2.2 Graphical Modelling Framework (GMF)

O SLE irá desenvolver as os protocolos de interação recorrendo ao editor GMF gerado pela ferramenta EuGENIA [30]. O editor GMF é implementado sob uma GMF *Runtime* que consiste numa *framework* baseada em Java para correr editores gráficos para modelos EMF [37].

4.8.2.3 Epsilon Object Language (EOL)

A EOL [27] é uma linguagem de programação imperativa da família Epsilon para criar, fazer *queries* e modificar modelos EMF. Possui as características de uma linguagem imperativa como declarar variáveis, sequenciação, ciclos *for* e *while*, condições *if*, etc. Possuindo também também funções de pesquisa de coleções (e.g. Sequence1..5.select(x|x>3)).

A EOL é a base de linguagens como a EGL que é fundamental na arquitetura da solução proposta.

4.8.2.4 Epsilon Generation Language (EGL)

A EGL [21] é uma linguagem pertencente à família Epsilon que possibilita a geração automática de código a partir de um modelo, convertendo determinadas informações do mesmo em texto, ou seja, numa linguagem de programação alvo pretendida, que no caso da ferramenta desenvolvida será a linguagem de programação Java. O EGL é necessário para transformar o metamodelo do protocolo de interação, gerado pela DSL de protocolos de interação, em código Java, de forma a poder ser executado pela plataforma. Este passo é de elevada importância, uma vez que é o que torna possível a ferramenta ser personalizada e apresentar diferentes protocolos de interação.

Na figura 4.12 pode observar-se como as tecnologias apresentadas acima se relacionam entre si.

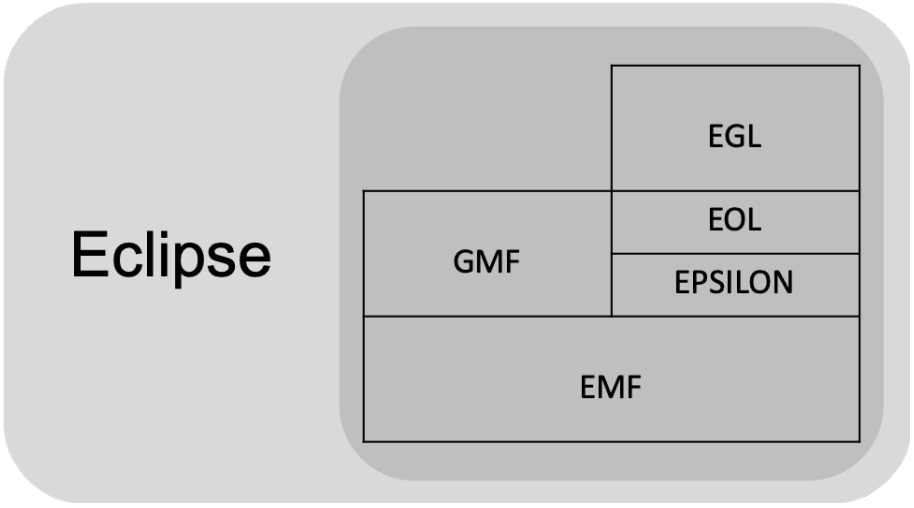


Figura 4.12: Relação entre as tecnologias de modelação do IDE Eclipse

4.8.3 Tecnologias de sintetização de voz e reconhecimento de voz, sons não-vocais e gestos

4.8.3.1 Sphinx-4

As ferramentas de reconhecimento de voz utilizadas na nova solução do modelo de interação da plataforma Model-By-Voice previam-se ser as mesmas que são utilizadas no modelo de interação antigo, sendo estas a Sphinx-4 e a Google Cloud Speech. Como referido anteriormente estas ferramentas resolviam os problemas que se consideravam importantes para o bom funcionamento da ferramenta, sendo estes:

- Capacidade de operar sem conexão à Internet.
- Ferramenta *open-source* que disponibilize utilização gratuita.
- Taxa de precisão credível no reconhecimento de palavras.
- Capacidade de filtração das palavras em ambientes ruidosos.

Contudo, devido a uma falta de consistência entre a comunicação da plataforma Model-By-Voice e a API do Google Cloud Speech-to-Text, optou-se apenas por utilizar a ferramenta Sphinx-4 que, embora apenas aceite as palavras na gramática da plataforma, revelou ter um reconhecimento de voz muito aceitável para as palavras presentes nesta gramática. A plataforma já utilizava a Sphinx-4 para comandos genéricos da aplicação que sempre estiveram definidos na gramática de regras (*create*, *remove*, *save*, *ect.*) e agora substitui também a Google Cloud Speech-to-Text, que era utilizada para reconhecer as variáveis como o nome e tipo dos objetos que o utilizador pode criar ou editar, tendo agora de estas palavras estar definidas na gramática plataforma para serem reconhecidas pela ferramenta Sphinx-4.

A ferramenta Sphinx-4 não necessita de conexão à internet para funcionar e é disponibilizada de forma gratuita. É necessário definir uma gramática de regras para definir as palavras reconhecidas, como referido anteriormente.

A decisão de manter a ferramenta Sphinx-4 para realizar o reconhecimento de voz na reformulação da solução anterior, deve-se ao facto de esta estar a funcionar corretamente e as suas características compatuarem com a nova solução que se pretendia desenvolver. De acordo com o meu conhecimento é inexistente uma ferramenta de reconhecimento de voz gratuita com características superiores à Sphinx-4.

4.8.3.2 freeTTS

A tecnologia de sintetização de voz da nova solução do modelo de interação da plataforma Model-By-Voice irá, também, manter-se a mesma que a da solução anterior, sendo esta a ferramenta freeTTS. Optou-se por manter esta ferramenta por esta estar escrita em Java e já estar integrada na plataforma Model-By-Voice e em ambiente Eclipse. De acordo com o

meu conhecimento, não existe, até ao momento, uma ferramenta de sintetização de texto para voz, disponibilizada de forma gratuita, tão versátil como a freeTTS. Outras ferramentas foram estudadas como a ferramenta MaryTTS [13], que apesar de também ser escrita em Java, revela-se menos versátil que o freeTTS, uma vez que apenas disponibiliza uma voz feminina, o que limita bastante o seu uso, dado não ser possível associar diferentes comandos a diferentes vozes [47].

4.8.3.3 Sonic

De forma a possibilitar a personalização do sintetizador de voz descrito acima, modificando características como a velocidade, timbre, taxa de amostragem e volume, foi utilizada a biblioteca Sonic [74] que dispõe de um simples algoritmo para acelerar ou abrandar ficheiros de áudio.

Foi desenvolvido um sistema que grava um ficheiro de áudio de qualquer frase sintetizada pelo sintetizador de voz freeTTS e essa frase será posteriormente modificada segundo as opções escolhidas na velocidade, timbre e volume, antes de reproduzida para o utilizador ouvir.

4.8.3.4 Musicg

A biblioteca Musicg, escrita na linguagem de programação Java, e as ferramentas de comparação de ficheiros de áudio associadas à mesma são utilizadas para fazer o reconhecimento de sons não-vocais na plataforma. Escolheu-se esta ferramenta para o reconhecimento destes sons uma vez que a biblioteca está estrita na mesma linguagem da plataforma, o que permitiu um integração muito fluída. Considera-se ainda que os resultados de reconhecimento de sons não-vocais por meio das ferramentas desta biblioteca foram bastante satisfatórios.

4.8.3.5 Mouse gesture recognition with Hidden Markov Model

O novo modelo interação irá suportar a integração de gestos desenhados, ou seja o utilizador poderá executar ações como *queries* para obter informações e operações básicas CRUD, através de gestos 2D. Desta forma o utilizador poderá comunicar à plataforma as ações que pretende executar através de gestos desenhados, em alternativa à voz.

A ferramenta escolhida para processar estes gestos foi a ferramenta descrita em [55]. Esta escolha deve-se ao facto de a mesma estar escrita em Java, prevendo-se uma integração mais fácil na plataforma Model-By-Voice, de ser *open-source* e apresentar bastante precisão no reconhecimento de gestos desenhados. O funcionamento detalhado desta ferramenta está descrito na secção do trabalho relacionado.

Na figura 4.13 é possível observar como estão relacionadas as tecnologias apresentadas acima.

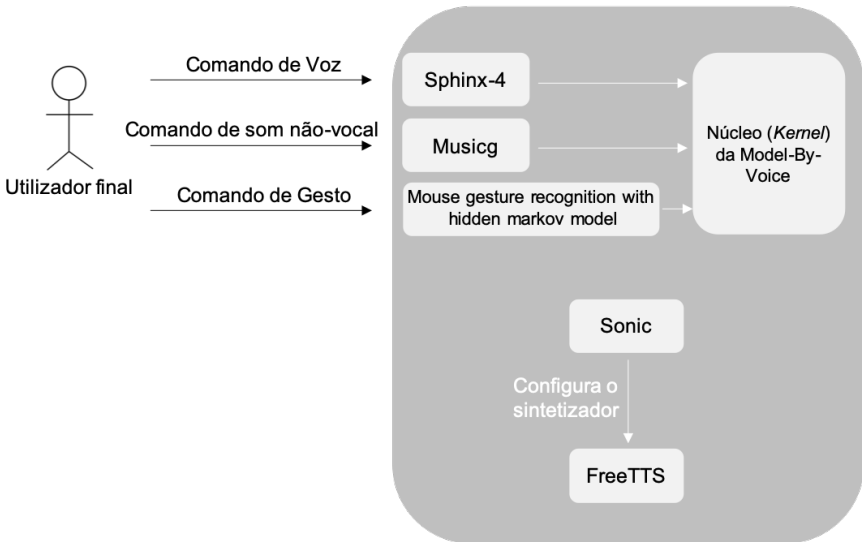


Figura 4.13: Relação entre as tecnologias de sintetização de voz (e a sua configuração) e reconhecimento de voz, sons não-vocais e gestos

Capítulo 5

Avaliação da usabilidade

Este é um capítulo dedicado à avaliação da usabilidade da solução proposta. Serão descritos possíveis cenários de aplicação relacionados com os diferentes potenciais utilizadores da plataforma Model-By-Voice e do seu novo modelo de interação. Face aos cenários propostos, serão identificadas as adversidades que podem surgir para os diferentes grupos de utilizadores ao utilizarem este tipo de plataforma de modelação e a forma como o novo modelo de interação prevê apresentar uma solução para as mesmas. Será também descrita a estratégia de validação com que se averiguou se a solução proposta era de facto viável e relevante na solução do problema que se propôs resolver, apresentando o modo como se conduziu a experiência empírica de teste da mesma. Serão enumerados os desafios propostos aos utilizadores tanto no modelo antigo de interação da plataforma como no novo.

5.1 Cenários de aplicação

5.1.1 Engenheiros de Linguagens de Software

O papel do SLE no novo modelo de interação da plataforma Model-By-Voice é desenvolver o protocolo de interação. Este protocolo é desenvolvido em ambiente Epsilon [28] através da DSL de geração de protocolos de interação desenvolvida no âmbito desta dissertação. Trata-se de uma linguagem diagramática simples onde o SLE poderá construir protocolos de interação personalizados para diversos tipos de utilizador distintos com diferentes níveis de conhecimento e diferentes dificuldades visuais.

O SLE escolhe, então, o nível de conhecimentos de modelação e o nível de dificuldades visuais do utilizador a que se destina o protocolo, os comandos que pretende que este possa utilizar, as frases e sons não-vocais que pretende que sejam utilizadas no feedback

dado pela ferramenta e os gestos desenhados e sons não-vocais que pretende que representem certos comandos na ferramenta. Caso não sejam definidas outras frases ou sons não-vocais de *feedback*, a ferramenta irá utilizar as respostas *default* existentes na mesma.

Posteriormente, é também função do SLE gerar o código java para ser integrado na ferramenta através da linguagem EGL, seguindo o processo descrito no capítulo seguinte.

5.1.2 Utilizadores finais com dificuldades visuais

Os utilizadores finais que possuam algum tipo de incapacidade visual, ou chegando mesmo a ser invisuais, são tidos como um caso de uso extremo da ferramenta, sendo o desenvolvimento do novo protocolo de interação concebido de forma a ser o mais inclusivo possível podendo-se perfeitamente estender a este tipo de utilizadores.

O novo modelo de interação apresenta uma solução para alguns problemas que utilizadores invisuais poderiam sentir ao utilizar o modelo de interação antigo da plataforma. Como descrito no capítulo de Trabalho relacionado, as pessoas invisuais que possuam noções espaciais 2D, utilizam no seu quotidiano tabelas, para sintetizarem informação e navegarem na mesma. Partindo desse princípio, este conjunto de utilizadores poderá agora usufruir do paradigma de tabela para modelar diagramas, como explicado no capítulo de Introdução de requisitos da ferramenta.

A possibilidade de personalizar o sintetizador de voz, regulando a velocidade e o timbre da voz, é algo que poderá aumentar a produtividade de utilizadores invisuais. Ao estarem habituados a *screen readers*, que podem transmitir informação a uma velocidade bastante elevada, os invisuais têm o ouvido treinado para compreender informação transmitida a velocidades mais elevadas, o que levou à inclusão desta característica de personalização do sintetizador de voz para proporcionar um ambiente em que os invisuais possam ser mais produtivos e não terem de aguardar longos períodos de tempo pelo *feedback* habitual do sintetizador antigo presente no modelo de interação antigo da plataforma.

Com o objetivo de também melhorar a produtividade, destes utilizadores em particular, adicionou-se também os 'atalhos' proporcionados pela integração de sons não-vocais, como palmas e assobios, e gestos desenhados para executar comandos, bem como no caso dos sons não-vocais, para dar *feedback* ao utilizador.

Assim, os utilizadores finais que tenham dificuldades visuais, ou que sejam mesmo invisuais, poderão usufruir de um protocolo de interação personalizado às suas dificuldades e conhecimentos de modelação, levando a um aumento da produtividade e uma melhor experiência de utilizador a modelar diagramas.

5.1.3 Utilizadores finais profissionais de Engenharia de Modelação

Os utilizadores finais que sejam profissionais na área de Engenharia de Modelação também poderão usufruir da plataforma Model-By-Voice, em especial do novo modelo de interação, uma vez que as novas funcionalidades permitem aumentar a produtividade da

atividade de modelar. O utilizador tendo um protocolo personalizado, neste caso a alguém com vasto conhecimento de modelação, irá usufruir de muitos 'atalhos' na plataforma, principalmente no *feedback* que lhe é dado, uma vez que não necessita de uma abordagem tão proteccionista da parte da ferramenta, como esta teria com um utilizador menos experiente. Este abreviamento do *feedback*, numa atividade de modelação que se estende por longos períodos, irá melhorar muito a sua experiência de utilizador e consequentemente a sua produtividade.

Este tipo de utilizador poderá beneficiar também muito da modelação através de sons não-vocais e gestos que lhe permitirão modelar de uma forma mais rápida e interativa. O paradigma de tabela também pode ser útil para profissionais da Engenharia de Modelação, uma vez que se trata de uma metodologia de trabalho muito comum em ambiente profissional e poderá ser uma forma muito útil de navegar e organizar diagramas com complexidade mais elevada.

5.1.4 Utilizadores finais estudantes de Engenharia Informática

O novo modelo de interação da plataforma Model-By-Voice foi pensado para também ter uma vertente educativa, tanto em aspetos de utilização da plataforma como em aspetos da Engenharia de Modelação em si.

Caso o protocolo de interação seja definido para utilizadores com baixos conhecimentos de modelação, a plataforma irá a cada comando fazer uma breve explicação de Engenharia de Modelação no contexto do comando que foi executado. Por exemplo, se o utilizador executar o comando *"Create Node"* ("Criar nó"), a plataforma irá, antes de prosseguir com a execução do comando, enunciar *"A node is the simplest entity in Modelling Engineering. A node is the basic element. The nodes define other entity types."* ("Um nó é a entidade mais simples na Engenharia de Modelação. Um nó é o elemento básico. Nós definem outros tipos de entidades."). O mesmo acontece para os restantes comandos, dando ao utilizador noções básicas de Engenharia de Modelação à medida que este descobre também a própria ferramenta.

Desta forma, o novo modelo de interação da plataforma Model-By-Voice, poderá também ser bastante útil para estudantes de Engenharia Informática que frequentem cadeiras relacionadas com Engenharia de Modelação, uma vez que esta plataforma oferece novos mecanismos de interação, como voz, sons não-vocais e gestos desenhados que poderão ser uma forma mais estimulante, do que a abordagem diagramática utilizada na maior parte das plataformas de modelação, para executar as tarefas a que os estudantes são propostos.

Estas novas abordagens face à modelação combinadas com a vertente pedagógica do novo modelo de interação da plataforma, poderá constituir uma ferramenta que é útil no âmbito da aprendizagem de Engenharia de Modelação.

5.2 Teste de exemplo

Com o intuito de demonstrar o novo modelo de interação da plataforma Model-By-Voice, procedeu-se à realização de um teste de execução, de forma a revelar todo o processo de construção do protocolo de interação, geração da instância da plataforma que utiliza esse protocolo para o utilizador que foi desenhado e utilização da ferramenta cobrindo os novos comandos integrados, novos mecanismos de interação e o novo paradigma de tabela.

Para se realizar este teste escolheu-se a DSL *"FileSystem"*, ilustrada na Figura 5.1, uma vez que esta é dotada de alguma complexidade e permite avaliar se o novo modelo de interação se adapta bem a linguagens complexas.

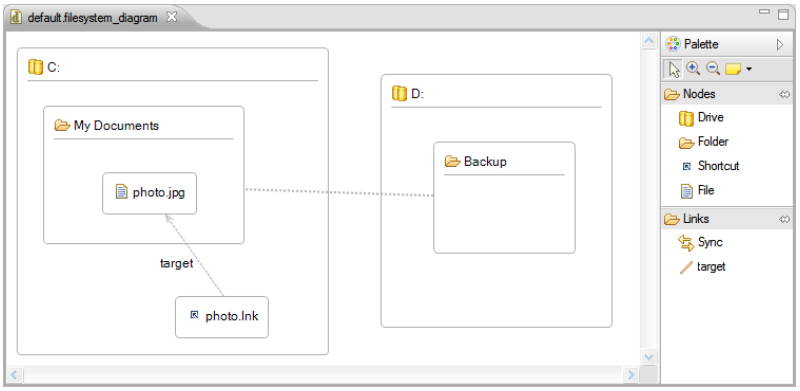


Figura 5.1: Exemplo de modelo na DSL *File System*.

Primeiramente o SLE teve de seguir um pequeno conjunto de passos para construir um protocolo de interação.

No projeto de modelação de protocolos de interação ("protocolgen19") teve de se dirigir à pasta "*model*" e posteriormente à pasta "*examples*". Nesta, o SLE carregou no botão direito do rato e posteriormente em "*New*". De seguida o SLE carregou em "*Other...*"(Figura 5.2).

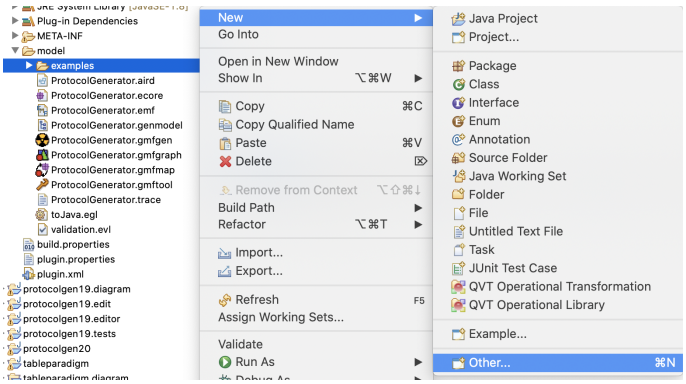


Figura 5.2: Primeiro passo para a criação do diagrama de protocolo de interação.

Posteriormente, O SLE escolheu "Protocolgen19 Diagram" e clicou em "*Next*" para

poder começar a modelar o mesmo (Figura 5.3).

Agora o SLE pode começar a modelar protocolos de interação arrastando os elementos do mesmo presentes na coluna da direita para a área de modelação do diagrama (Figura 5.4).

Neste teste de exemplo foi utilizado o protocolo de interação ilustrado na Figura 5.5.

Na secção 1 da Figura 5.5 observa-se o objeto utilizador. Nas propriedades deste objeto (Figura 5.6) o SLE define o nível de conhecimentos de modelação e o nível de dificuldades visuais do utilizador a que se destina o protocolo de interação.

Na secção 2 da Figura 5.5 estão ilustrados os comandos que o SLE escolheu para o utilizador utilizar na plataforma, o utilizador apenas terá permissão para utilizar os comandos que o SLE definir.

Na secção 3 da Figura 5.5 observa-se um objeto do tipo gesto onde é definido o movimento gestual que se irá traduzir num comando na plataforma. Nas propriedades, define-se o comando no campo "Nome" e a descrição do gesto no campo "Descrição" (Figura 5.7).

Na secção 4 da Figura 5.5 estão presentes dois objetos do tipo "Som" que são destinados a substituir comandos na plataforma. O comando a ser substituído será descrito nas propriedades do objeto no campo "Nome" e o *path* para o som que se pretende que o substitua é definido no campo *"Path"* (Figura 5.8).

Por fim, na secção 5 da Figura 5.5 é possível observar um objeto do tipo "Som" destinado a substituir todas as frases de *feedback* que são enunciadas quando o nome de algum objeto é *"null"*. O tipo de frase de *feedback* que se pretende substituir é indicado nas propriedades do objeto no campo "Nome" e o *path* para o som que se pretende que

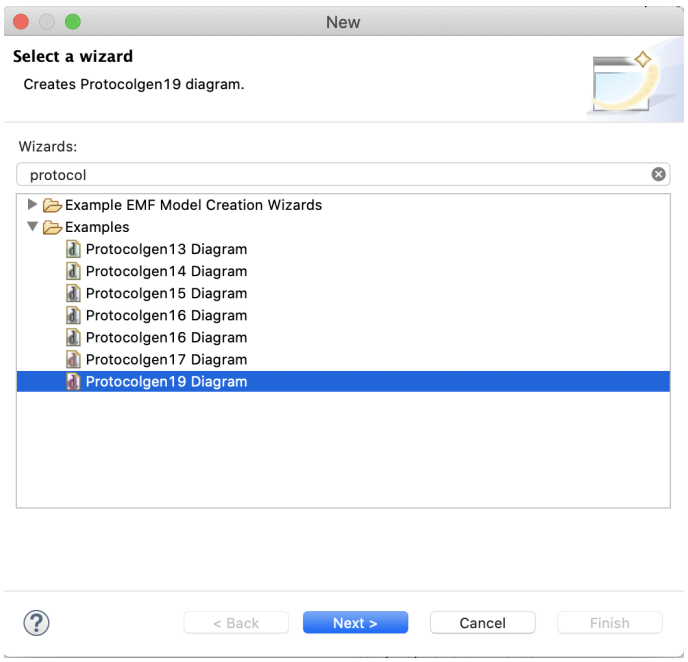


Figura 5.3: Segundo passo para a criação do diagrama de protocolo de interação.

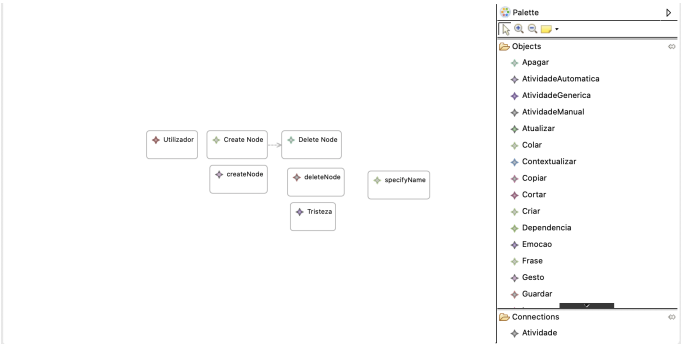


Figura 5.4: Área de modelação do diagrama e elementos possíveis a integrar no diagrama.

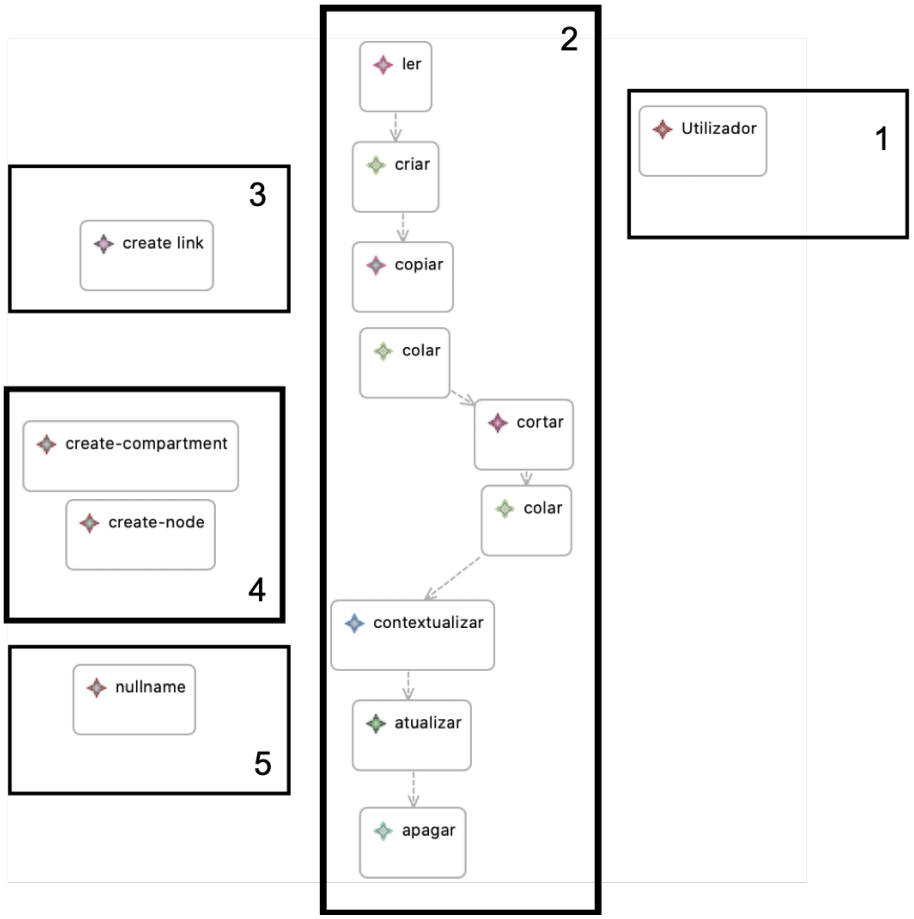


Figura 5.5: Diagrama exemplo do protocolo de interação utilizado neste teste de exemplo.

substitua a frase de *feedback* é definido no campo *"Path"* (Figura 5.9).

Posteriormente a realizar o protocolo de interação, o SLE teve de proceder da seguinte forma para que este seja transformado em código java e integrado na plataforma.

O primeiro passo foi na pasta "protocolgen19", dirigir-se ao ficheiro "toJava.egl". Neste ficheiro encontra-se o código EGL que irá transformar o protocolo de interação com o utilizador consoante o modelo de interação desenhado pelo SLE. O SLE, então, carregou no botão direito to rato por cima deste ficheiro, de seguida em "*Run as*" e por fim em "*Run*

Property	Value
Atividade	
Name	Utilizador
Nivel De Conhecimento	0
Nivel De Dificuldade Visual	1

Figura 5.6: Propriedades do objeto utilizador.

Property	Value
Descricao	downandleft
Emocao	
Name	create link

Figura 5.7: Propriedades do objeto Gesto.

Property	Value
Emocao	
Name	create-compartment
Path	/Users/joaomiguel/Desktop/SOUNDSMODELBYVOICE/WAVSOUNDS/create-compartment.wav

Figura 5.8: Propriedades do objeto Som quando se destina a substituir um comando na plataforma.

configurations...", como descrito na Figura 5.10.

De seguida, o utilizador encontrou o painel ilustrado na Figura 5.11.

Aqui, na secção *Template*, o utilizador selecionou para a secção *Source* "/protocolgen19/model/toJava.egl", para que o ficheiro "toJava.egl" seja executado. Na secção *"Text generated should be printed to:"*, foi selecionado "*The following file:*" e escolhido o ficheiro "/ModelByVoice1/src/Main/ModelByVoice1". Assim quando o programa tiver terminado, o código *main* da plataforma Model-By-Voice será alterado para respeitar o novo protocolo de interação com o utilizador, de acordo com o metamodelo desenvolvido pelo SLE.

De seguida, e após carregar no botão "*Apply*", o utilizador dirigiu-se à secção *Models*, ainda na página de *Run Configurations*. (Figura 5.12)

Nesta secção, o SLE carregou no botão "*Add...*" e posteriomente em "*EMF Model*" e por fim em "OK"(Figura 5.13).

Posteriormente o SLE encontrou o painel onde lhe foi pedido para indicar qual o modelo EMF que pretende utilizar (Figura 5.14).

O SLE deu o nome pretendido para identificar o modelo. No caso de exemplo ilustrado em 5.14 o nome escolhido foi "protocol01". De seguida, na secção "*Files/URIs*", o SLE escolheu para inserir em "*Model file*" o diagrama que representa o protocolo de interação que este desenvolveu. No exemplo apresentado na Figura 5.15 este seria o diagrama "`default.protocolgen19_diagram`".

Por fim o SLE selecionou, na secção "*Metamodels*", o metamodelo "*Protocol Generator.ecore - protocolgen19/model*" que se trata do metamodelo dos protocolos de interação

Property	Value
Emocao	
Name	nullname
Path	/Users/joaomiguel/Desktop/SOUNDEXPERIENCE/nullName.wav

Figura 5.9: Propriedades do objeto Som quando se destina a substituir uma frase de *feedback* na plataforma.

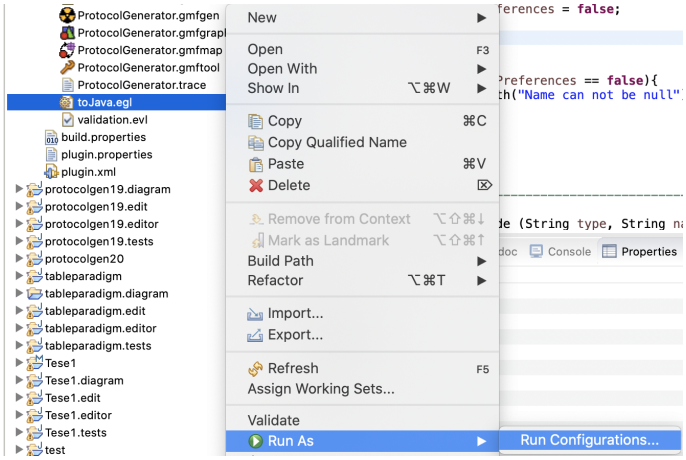


Figura 5.10: Primeiro passo para a conversão do protocolo de interação para código java, através da linguagem EGL, para que este possa ser integrado na plataforma Model-By-Voice.

(Figura 5.16).

O SLE carregou então no botão "OK", posteriormente em "*Apply*" e por fim no botão "*Run*". Quando o programa terminar de ser executado, a classe *main* da plataforma Model-By-Voice terá sido reformulada para respeitar o protocolo de interação definido pelo SLE no diagrama que este selecionou na secção de *Models*. Agora o utilizador final alvo do diagrama poderá utilizar a plataforma com este protocolo de interação e ter uma experiência mais adequada aos seus conhecimentos de modelação e/ou necessidades.

O utilizador final posteriormente modelou a DSL *"File Systems"* com o exemplo ilustrado na Figura 5.1. Este teste de exemplo demonstra a primeira interação do utilizador com a plataforma, assim antes de se dar início à modelação, este terá de criar o seu perfil na plataforma onde define as suas preferências.

Na Figura 5.17 está ilustrado o *feedback* textual que foi apresentado por meio da

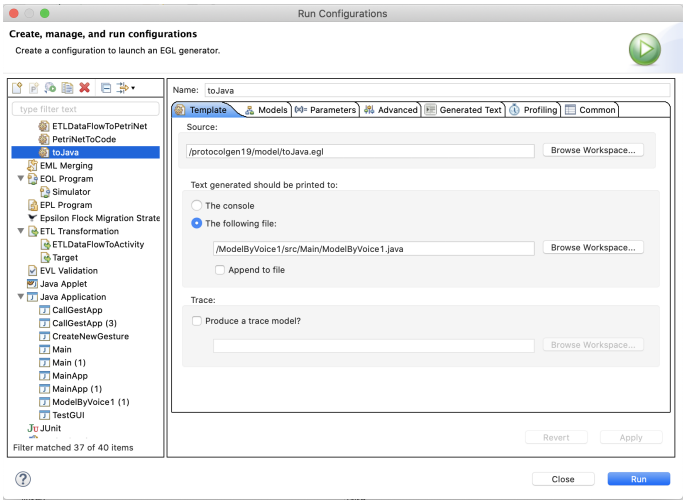


Figura 5.11: Painel de "*Run Configurations*"na secção *Template* para o ficheiro "toJava.egl".

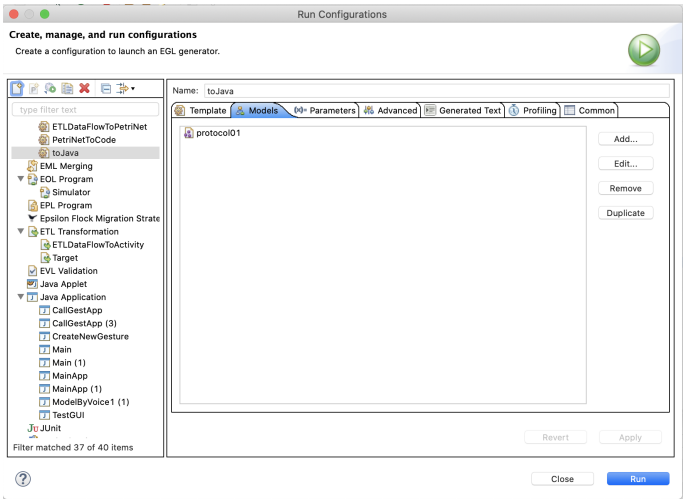


Figura 5.12: Painel de "*Run Configurations*"na secção *Models* para o ficheiro "toJava.egl".

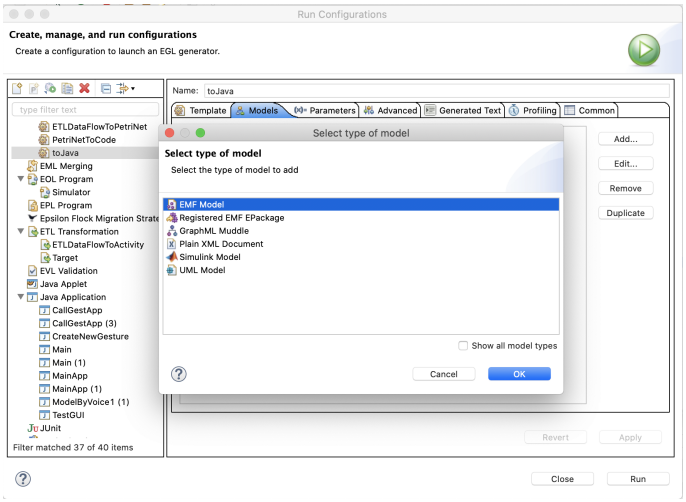


Figura 5.13: Adicionar modelo EMF aos modelos de "*Run configurations*"do ficheiro "toJava.egl".

consola/linha de comandos da plataforma *Eclipse* durante a execução deste teste exemplo.

A execução deste teste de exemplo do novo modelo de interação está arquivado na plataforma Zenodo em suporte audiovisual e pode ser visualizado no seguinte endereço eletrónico: Vídeo de execução do teste de exemplo descrito [26].

5.3 Processo de validação e preparação da experiência

O processo de validação da solução consistiu em desenvolver um protótipo da arquitetura proposta e, uma vez implementado, proceder a avaliações empíricas através da utilização da mesma por parte de utilizadores reais.

Previamente a testar o novo modelo de interação da plataforma Model-By-Voice com utilizadores reais foi feita uma sessão de treino, de forma a detetar alguns erros mais

triviais e ao mesmo tempo detetar algum problema de interação que pudesse ser logo corrigida na ferramenta que foi apresentada aos utilizadores.

De forma a testar todas as vertentes do novo modelo de interação, esperava-se reunir utilizadores representativos da população em geral, incluindo utilizadores com dificuldades visuais ou mesmo invisuais. Contudo, devido à impossibilidade de realizar testes com pessoas que estivessem de alguma forma condicionadas, apenas foi possível realizar a experiência de teste com utilizadores "comuns". Contudo, mesmo dentro deste conjunto de utilizadores esperavam-se encontrar diferentes perfis. Assim, foi concebido um inquérito para ser realizado previamente à realização da experiência, de forma a obter o perfil dos utilizadores que iriam realizar a mesma. Assim foram inferidos aspetos que podiam interferir na análise dos resultados, como os conhecimentos que dado utilizador tem na área de modelação ou se está habituado a utilizar ferramentas deste tipo. O inquérito realizado pode ser consultado no Anexo I.

Foram escolhidos 10 utilizadores para participar na sessão experimental. Todos os participantes eram atualmente estudantes de Engenharia Informática ou já Engenheiros Informáticos formados a exercer na área da Informática. Os participantes tinham idades compreendidas entre 21 e 40 anos e 90% estavam familiarizados com atividades de modelação.

A análise da experiência empírica avaliou se os diferentes utilizadores conseguem executar ações de forma mais prática, quanto tempo demoram para executar as mesmas

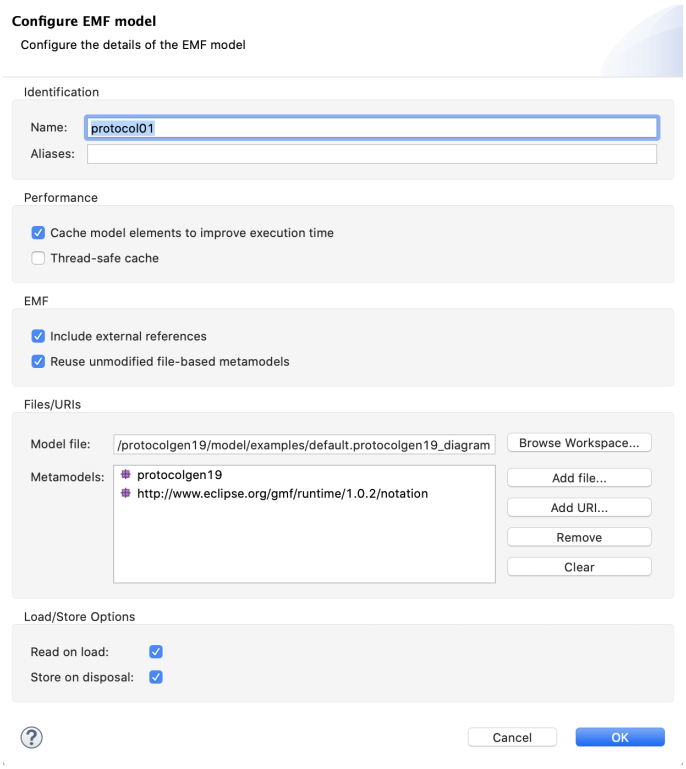


Figura 5.14: Painel para indicar o modelo EMF a ser utilizado.

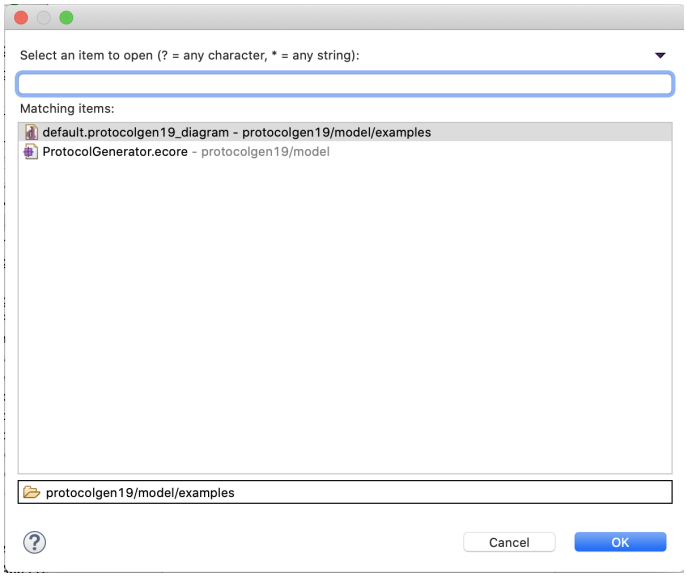


Figura 5.15: Painel para escolher o diagrama representativo do protocolo de interação a utilizar.

e qual o grau de satisfação com a experiência de interação. Desta forma foi possível retirar das experiências importantes conclusões acerca da *performance* do novo modelo de interação, e como este se compara em termos de usabilidade ao modelo de interação anterior da plataforma Model-By-Voice.

Durante a experiência foram medidas a correção, eficiência e evolução da nova solução do modelo de interação da plataforma Model-By-Voice. A correção foi averiguada considerando a forma em que o utilizador conseguia utilizar o novo modelo de interação para

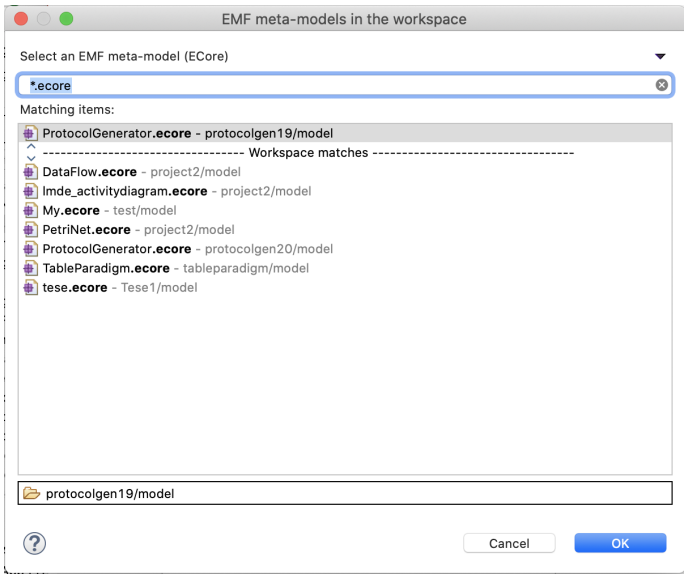


Figura 5.16: Painel para escolher o metamodelo dos protocolos de interação "*Protocol Generator.ecore - protocolgen19/model*".

Figura 5.17: *Screenchots* da execução do teste de exemplo de modelação com a DSL *"File System"* utilizando o novo modelo de interação.

o que é pretendido. Foram também contabilizados os erros cometidos para a medição desta componente, e se o utilizador cometeu mais erros com o novo modelo de interação ou com o antigo. A eficiência da nova solução foi averiguada medindo o tempo que o utilizador demora a executar as tarefas propostas, comparando com o tempo que demora a executar as mesmas tarefas na solução antiga. A evolução foi averiguada avaliando se o utilizador conseguiu evoluir um protocolo de interação estando este já criado e com alguns componentes.

Antes de se dar início à experiência empírica, foi fornecida aos participantes uma breve sessão de explicação das funcionalidades da plataforma e dos comandos que poderiam utilizar durante a experiência. Foi-lhes também fornecido material de apoio que continha alguns esclarecimentos e onde puderam consultar estes mesmos comandos e também um guião que enunciava detalhadamente as tarefas propostas na plataforma.

5.4 Descrição das atividades da experiência

Foram apresentadas diversas tarefas na nova solução e na antiga com perspetivas de utilizadores distintos: utilizadores finais e SLEs. Todos os participantes realizaram a experiência nas duas perspetivas.

Como SLEs, os utilizadores criaram um pequeno protocolo de interação de raíz e completaram/evoluiram um protocolo de interação que se encontrava incompleto.

Como utilizadores finais, os participantes foram convidados a executar um conjunto de testes selecionados para por à prova as funcionalidades do modelo de interação. Os utilizadores seguiram um guião que detalha todos os passos para a conclusão de cada teste. Estes testes contemplaram as novas funcionalidades propostas como modelar com som, com gestos, com o paradigma de tabela e utilizar novos comandos de "atalho" como os de copiar e usar padrões para criar novos nós.

A fim de testar a usabilidade e *performance* das novas funcionalidades de modelação com som e gestos, e dos novos comandos de "atalho", foi pedido aos utilizadores que modelassem uma pequena DSL do tipo do controlador de semáforos *"Traffic Lights"*, que simula o funcionamento de um sinal de trânsito, transitando entre os estados "verde", "amarelo" e "vermelho". Este exemplo da DSL *"Traffic Lights"* foi utilizado também na dissertação [47] para avaliar a *performance* da plataforma Model-By-Voice após esta ser desenvolvida. A experiência foi repetida também no modelo de interação antigo da plataforma utilizando apenas os comandos antigos e a interação não personalizada por voz. Desta forma, foi possível questionar o utilizador acerca do método de modelação em que se sentiu mais confortável e também averiguar o tempo que o mesmo levou a concluir a modelação nos diferentes cenários.

De forma a averiguar a usabilidade e *performance* do paradigma de tabela, foi pedido aos utilizadores que modelassem, a partir de uma DSL de *"file systems"*, um pequeno sistema de ficheiros utilizando o exemplo ilustrado na Figura 5.18. Este diagrama de sistema de ficheiros foi modelado pelos utilizadores no paradigma de tabela de forma a

permitir averiguar se os participantes tem a percepção de que este paradigma é útil e de fácil compreensão.

Visto que o paradigma de tabela foi adicionado à plataforma Model-By-Voice para tornar a gerência e percepção de modelos mais complexos mais trivial, tornando a navegação e pesquisa nos mesmos mais fácil, escolheu-se este exemplo, uma vez que apesar de pequeno, terá complexidade suficiente, sendo um exemplo realista e representativo das funcionalidades da tabela. Trata-se de um exemplo que contém compartimentos para além de nós normais, tornando-se assim mais complexo, sendo ideal para se averiguar se o paradigma de tabela é de facto mais útil do que o paradigma de grafo em casos de maior complexidade.

No fim da experiência os utilizadores foram convidados a responder a algumas questões que irão permitir inferir se o utilizador consegue interpretar algumas situações na plataforma e se consegue identificar erros na mesma. Foram também convidados a preencher um questionário de satisfação onde puderam avaliar o desempenho dos modelos de interação de ambas as plataformas e comparar os mesmos entre si. Estas questões e o questionário realizado após a experiência poderão ser consultados no Anexo I.

Na Figura 5.19 é apresentado um diagrama de atividades que representa como foram conduzidas as sessões experimentais de teste do novo modelo de interação.

O guião que foi utilizado para conduzir a experiência empírica de teste do novo modelo de interação da plataforma Model-By-Voice, assim como o material de apoio disponibilizado para a mesma, encontram-se disponíveis no Anexo II.

Os resultados da experiência encontram-se apresentados e discutidos no capítulo seguinte, onde são debatidas as respostas aos questionários, assim como os tempos de duração das tarefas e as conclusões a partir daí inferidas.

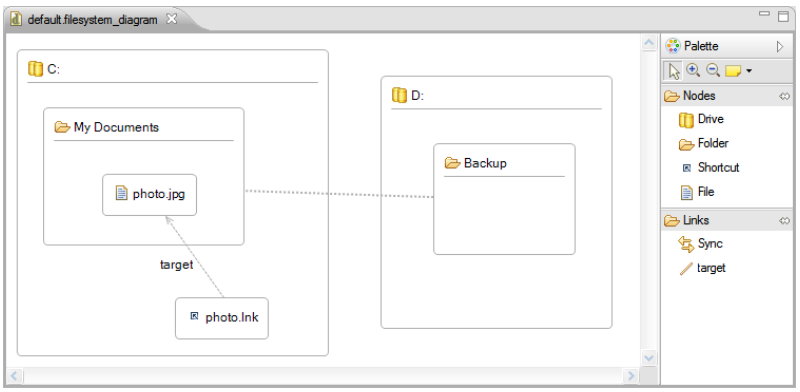


Figura 5.18: Sistema de ficheiros proposto a ser modelado pelos utilizadores.

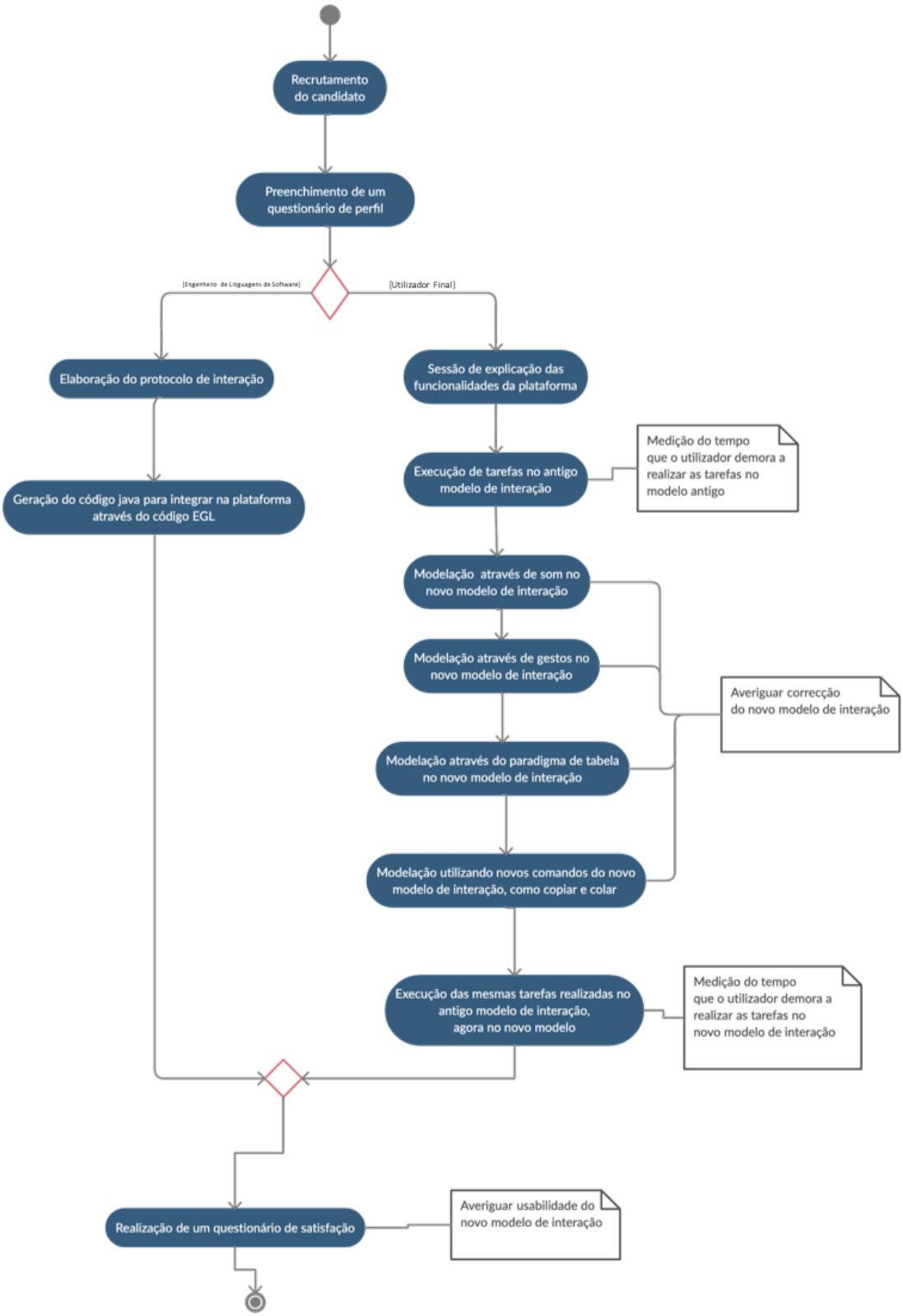


Figura 5.19: Planeamento das sessões de avaliação do novo modelo de interação.

Capítulo 6

Resultados do estudo de usabilidade

Este é um capítulo dedicado à apresentação e discussão dos resultados obtidos nas experiências de avaliação da solução proposta apresentadas no capítulo anterior.

6.1 Objetivos

O objetivo que se pretendia alcançar no final da experiência era perceber se o novo modelo de interação era fácil de utilizar por diferentes tipos de utilizadores:

- Utilizadores sem experiência.
- Utilizadores experientes.
- Engenheiros de Linguagem de *Software*.

Pretende-se que o novo modelo de interação reflita um uso mais personalizado da plataforma, permitindo atingir resultados mais rapidamente e com menos dificuldade.

Os principais objetivos descritos foram divididos em objetivos mais precisos que foram inferidos através da análise dos questionários de satisfação da experiência. Estes objetivos podem ser descritos nas seguintes questões inspiradas em [84]:

- A interação é fácil de compreender?
- O modelo de interação leva a cometer erros?
- As tarefas propostas foram fáceis de completar?
- O protocolo de interação é fácil de evoluir?
- A personalização da ferramenta torna a utilização da ferramenta mais rápida/fácil?

- Os utilizadores ficaram agradados com as melhorias implementadas no novo modelo de interação?

As variáveis independentes foram:

- A interpretação da interação com a plataforma, em que o utilizador tem de compreender o *feedback* dado pela ferramenta e compreender que comandos pode executar posteriormente.
- Os erros produzidos, como não identificar que dado diagrama não foi criado como era pretendido.
- A completude, na medida em que é dado, ao utilizador, um protocolo de interação incompleto e este tem de o completar.
- A evolução, em que é dado ao utilizador um protocolo de interação e lhe é proposto atribuir um novo sentido a um som ou um gesto, por exemplo.

As variáveis que se encontravam dependentes das enunciadas acima eram as respostas dos utilizadores, que irão ter em conta a sua interpretação da ferramenta, os erros que cometeram, se conseguiram completar as tarefas propostas e se acharam a ferramenta fácil de evoluir.

Uma pré-condição para realizar uma experiência empírica é ter presente uma hipótese clara. A hipótese irá ditar como será conduzida a experiência, decidindo quais as variáveis a inserir no estudo e como medi-las [70].

Para cada questão foram definidas as hipóteses "nula" (H0) ou "alternativa" (H1). As duas hipóteses destinam-se a comparar o novo modelo de interação da plataforma Model-By-Voice com o antigo quanto à rapidez, correção e à usabilidade. Foram definidas H0 e H1 relativamente a estas características, tal como descrito na Tabela 6.1.

Para avaliar a rapidez com que os utilizadores terminam com sucesso as tarefas propostas na plataforma foram comparados os tempos em que estes terminam as mesmas utilizando o modelo de interação antigo e o novo.

A correção do modelo de interação foi avaliada através dos erros cometidos pelos utilizadores durante a execução das tarefas propostas.

A usabilidade do novo modelo de interação foi avaliada através de um questionário de usabilidade (SUS), apresentado na Figura 6.1, que providencia um valor de 0-100.

6.2 Recolha de dados - 1ª Fase

Antes de se ter dado início à experiência empírica descrita no capítulo anterior, procedeu-se ao primeiro questionário que tinha o objetivo de inferir os diferentes tipos de perfis dos participantes, os seus conhecimentos na área da Engenharia de Modelação e a sua

Tabela 6.1: Definição das hipóteses H0 e H1 relativamente a rapidez, correcção e usabilidade.

	H0	**H1**
Rapidez	Usar o novo modelo de interação em relação ao antigo não influencia a rapidez de executar uma tarefa na plataforma.	Usar o novo modelo de interação em relação ao antigo influencia a rapidez de executar uma tarefa na plataforma.
Correcção	Usar o novo modelo de interação em relação ao antigo não influencia o cometer de erros ao executar uma tarefa na plataforma.	Usar o novo modelo de interação em relação ao antigo influencia o cometer de erros ao executar uma tarefa na plataforma.
Usabilidade	Usar o novo modelo de interação, com a possibilidade de o personalizar a meu gosto, em relação ao antigo não influencia a minha perceção de usabilidade da plataforma.	Usar o novo modelo de interação, com a possibilidade de o personalizar a meu gosto, em relação ao antigo influencia a minha perceção de usabilidade da plataforma.

percepção de quais seriam boas maneiras para realizar a interação com uma plataforma de modelação.

Os 10 participantes realizaram a experiência tanto na perspetiva de SLE e de utilizador final. Estes tinham idades compreendidas entre 21 e 40 anos, não apresentavam qualquer limitação ao nível visual, eram estudantes de engenharia informática ou profissionais na área e possuíam um nível de inglês de médio ou avançado.

A Figura 6.2 demonstra que apenas 10% dos participantes não estavam familiarizados com Engenharia de Modelação de Software, sendo estes 10% correspondentes a um dos Engenheiros Informáticos Profissionais, que apenas esteve em contacto com atividades de modelação durante o seu curso, não exercendo atividades deste tipo desde então. A maioria dos participantes quando pratica atividades de modelação recorre à plataforma StarUML [25].

60% dos participantes sente que não consegue personalizar a interação e utilizar atalhos em plataformas de modelação como o StarUML, como é ilustrado na Figura 6.3. Podendo-se assim assumir que existe necessidade de mercado de dotar uma plataforma de modelação com estas características.

Os participantes apontaram alguns aspetos negativos nas atuais plataformas de modelação como por exemplo a falta de *feedback*, uma curva de aprendizagem lenta, a dificuldade de editar modelos e ter presente as mudanças efetuadas. Com a introdução de um modelo de interação personalizável, a curva de aprendizagem deveria tornar-se mais rápida, uma vez que o utilizador pode personalizar a maior parte dos aspetos. Como o

		Strongly disagree 1	2	3	4	Strongly agree 5
1	I think that I would like to use the new interaction model of Model-By-voice frequently.	☐	☐	☐	☐	☐
2	I found the new interaction model to be unnecessarily complex.	☐	☐	☐	☐	☐
3	I found the new interaction model to be easy to use.	☐	☐	☐	☐	☐
4	I think that I could not use the new interaction model without the support of a technical person.	☐	☐	☐	☐	☐
5	I found the new functions in the new interaction model were well integrated.	☐	☐	☐	☐	☐
6	I thought there was a lot of inconsistency in new interaction model.	☐	☐	☐	☐	☐
7	I would imagine that most people would learn to use the new interaction model very quickly.	☐	☐	☐	☐	☐
8	I found the new interaction model very cumbersome to use.	☐	☐	☐	☐	☐
9	I felt very confident using the new interaction model.	☐	☐	☐	☐	☐
10	I could not use the new interaction model without having any knowledge of Modeling Engineering.	☐	☐	☐	☐	☐

Figura 6.1: Questionário de usabilidade (SUS) apresentado aos utilizadores no fim da experiência de teste do novo modelo de interação da plataforma Model-By-Voice I.

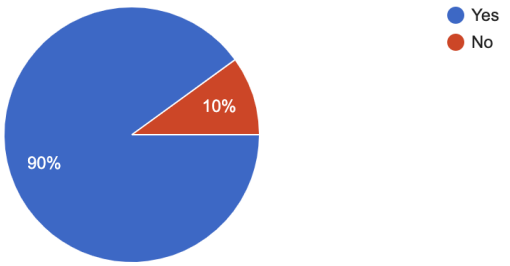


Figura 6.2: Resultados da familiarização com engenharia de modelação.

novo modelo de interação adapta o *feedback* aos níveis de conhecimentos de modelação e dificuldades visuais do utilizador, este nunca deverá sentir que não obtém *feedback* suficiente ou que está a obter demasiado *feedback* para os conhecimentos que possui.

Antes de realizarem a experiência, como se pode ver representado na Figura 6.4, 90% dos participantes considerou que a voz era uma forma útil de navegar em modelos, 50% destacou os gestos para realizar esta atividade, 40% os sons não-vocais e houve um participante que sugeriu utilizar *screen-readers* para realizar a navegação nos modelos. Devido à plataforma Model-By-Voice não ser baseada numa estrutura visual, os *screen-readers* para efetuar a navegação em modelos, de momento não apresentam muita usabilidade. Contudo, poderia ser uma boa opção para auxiliar utilizadores invisuais a guiarem-se melhor no IDE Eclipse e desempenharem até mesmo atividades que competem ao SLE como gerar o seu próprio código do protocolo de interação.

Quando questionados se um paradigma de tabela podia ser uma forma útil para

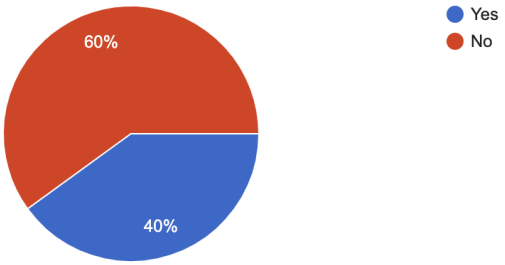


Figura 6.3: Resultados da possibilidade de personalizar a interação e utilizar atalhos nas atuais plataformas de modelação.

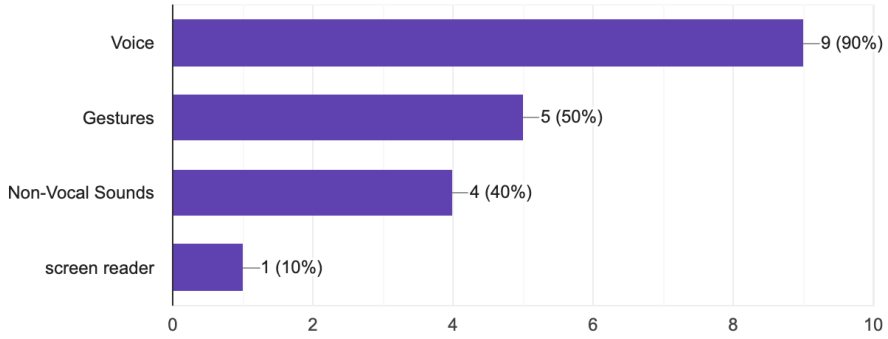


Figura 6.4: Resultados dos métodos úteis para navegar em modelos.

navegar em modelos, mesmo para pessoas invisuais, 50% dos participantes respondeu que sim, 40% respondeu que sim apenas para os invisuais que já possuíram visão e 10% respondeu que sim apenas para pessoas não cegas, como se pode observar ilustrado na Figura 6.5. Desta forma pode-se inferir que todos os participantes consideram uma abordagem de tabela, um paradigma útil para navegar em diagramas, pelo menos para pessoas não cegas.

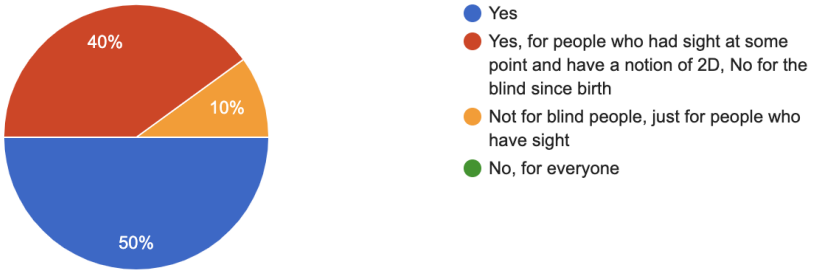


Figura 6.5: Resultados dos métodos úteis para navegar em modelos.

6.3 Recolha de dados - 2ª Fase

Após realizarem a experiência enunciada no capítulo anterior, os participantes foram convidados a responder a um questionário onde responderam a algumas questões sobre a experiência que acabaram de efetuar.

Quando apresentados com dois protocolos de interação distintos e questionados qual dos dois não estava correto segundo as regras da plataforma, apenas 10% respondeu de forma errada, o que demonstra que foi perceptível e que aprenderam os conceitos que usaram na experiência.

Todos os participantes consideraram que o novo modelo de interação era mais útil que o modelo antigo para os utilizadores, uma vez que este permite uma interação personalizada, possui um paradigma de tabela, tem novos mecanismos de interação como gestos desenhados e sons não-vocais e permite o uso de "atalhos" como copiar, colar e usar padrões. Tendo todos os participantes respondido que preferiram modelar no novo modelo de interação, em relação ao antigo, na experiência da DSL *"Traffic Lights"* que realizaram.

Também todos os participantes consideraram fácil adicionar novos sons não-vocais/gestos à plataforma e que o novo modelo de interação oferece aos utilizadores finais/SLE bastante expressividade.

O mecanismo de interação favorito dos participantes para modelar, tendo em conta a experiência com a DSL *"Traffic Lights"*, revelou-se ser a voz, tendo 80% dos utilizadores escolhido esta e 20% escolhido os gestos desenhados, como está representado na Figura 6.6. Nenhum dos participantes escolheu os sons não-vocais como mecanismo favorito de modelação, o que pode estar relacionado com o facto de ser algo menos óbvio da sua utilidade quando comparado com os gestos e voz, que são conceitos perfeitamente conhecidos e usados amplamente no quotidiano de qualquer pessoa.

A maior parte dos utilizadores preferiu modelar no paradigma de tabela em relação ao paradigma de grafo tendo 60% dos participantes escolhido este como paradigma de modelação preferido como ilustrado na Figura 6.7.

Para averiguar se o utilizador compreendeu bem o funcionamento do paradigma de

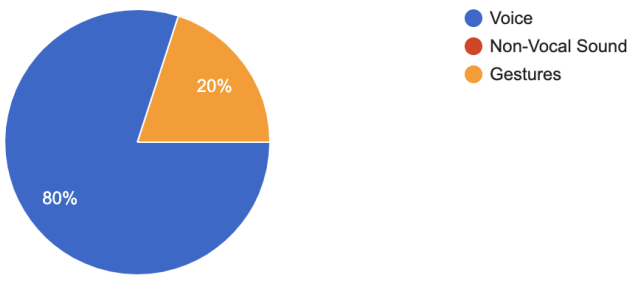


Figura 6.6: Resultados para os mecanismos favoritos de modelação.

Tabela 6.2: Tabela representativa do exemplo apresentado da DSL *File Systems*.

C:					
	My Documents				
		photo.jpg	target		
			photo.lnk		
				D:	
	sync				Backup

tabela, foram também propostas 3 atividades relacionadas com o exercício da DSL *"File Systems"* que modelaram na experiência de teste. Foi-lhes apresentada a tabela ilustrada na Tabela 6.2 que é a representação tabelar do exemplo que estes foram propostos a modelar.

As atividades propostas foram as seguintes:

- 1 - Liste todas as ligações existentes na tabela e o seu sentido.
- 2 - É possível saber quantos nós estão "sincronizados" com um dado nó? Descreva como.
- 3 - Como pode saber qual é o "alvo" do objeto "Photo.lnk"? Enumere os comandos que usaria no paradigma de tabela.

Todos os utilizadores conseguiram executar corretamente a atividade número 1, podendo-se assim inferir que é claro como funciona o sentido das ligações no paradigma de tabela.

Em relação à atividade 2, quatro dos participantes recorreriam ao comando *"List Diagram"* de forma a ver todas a ligações existentes e contariam as que estariam sincronizadas com um dado nó, o que está correto. Os restantes 60% dos participantes embora não tivessem indicado o comando *"List Diagram"* relataram por palavras um modo de proceder semelhante.

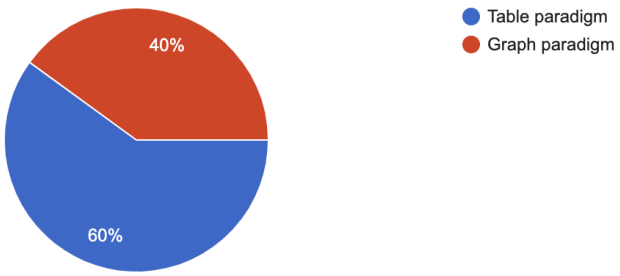


Figura 6.7: Resultados para qual o melhor paradigma de navegação.

Na pergunta 3, 90% dos participantes descreveram que ao estarem no nó de entrada "Photo.lnk" se recorrecem ao comando *"go up"* este iria levá-los à célula onde se encontra a ligação do tipo *"target"*, e que esta ao ser uma ligação, iria encaminhá-los para o nó de saída da mesma, indo assim parar ao "alvo" do objeto "Photo.lnk". Este tratava-se do procedimento correto e 90% dos participantes conseguiu descrevê-lo, podendo-se assim concluir que o paradigma de tabela tem um modo de funcionamento intuitivo e de rápida aprendizagem.

Por fim, os participantes responderam ao questionário de usabilidade representado na Figura 6.1. Os questionários de usabilidade (SUS) providenciam um valor de pontuação entre 0 e 100 com média em 68. Considera-se que sistemas com valores superiores a 68 têm uma boa usabilidade [12]. O questionário da experiência de teste obteve uma pontuação de valor igual a 80,75 na mesma escala.

Para cada tarefa proposta na experiência de teste foram medidos os tempos que os participantes demoraram a completá-las. A média, mínimo e máximo dos tempos por tarefa podem ser observados na Tabela 6.3.

6.4 Discussão dos dados

Após a análise dos dados recolhidos no decorrer da experiência, é possível inferir informação conclusiva acerca do novo modelo de interação.

Os participantes conseguiram concluir as tarefas propostas, tanto na perspetiva de SLE como de utilizador final. Na perspetiva de SLE todas as tarefas tiveram o resultado esperado. Na perspetiva de utilizador final, alguns diagramas das diversas tarefas apresentaram elementos com nomes diferentes do que estava definido. Este fenómeno deveu-se à ferramenta de reconhecimento de voz Sphinx-4 por vezes não conseguir perceber com

Tabela 6.3: *Timestamps* das tarefas executadas durante a experiência de teste

Tarefa	**Média**	**Mínimo**	**Máximo**
Criar protocolo de interação	3 min. 48 seg.	3 min. 10 seg.	4 min. 49 seg.
Completar protocolo de interação	2 min. 08 seg.	1 min. 20 seg.	4 min. 05 seg.
Modelar exemplo da DSL *"Traffic Lights"* no modelo de interação antigo	2 min. 48 seg.	1 min. 50 seg	5 min. 04 seg.
Modelar exemplo da DSL *"Traffic Lights"* no modelo de interação novo	2 min. 43 seg.	1 min. 48 seg.	4 min. 10 seg.
Modelar exemplo da DSL *"File Systems"* no novo modelo de interação no paradigma de tabela	2 min. 36 seg.	2 min. 03 seg.	3 min. 30 seg.

clareza as palavras proferidas. Contudo, apesar desta limitação tecnológica, os participantes conseguiram concluir as tarefas, tanto no novo modelo de interação como no antigo, com sucesso. Desta forma pode-se então concluir que o novo modelo de interação obteve resultados positivos quanto à correção, possuindo a vantagem de ter a possibilidade de executar comandos por gestos e sons não-vocais face ao modelo antigo, uma vez que estes mecanismos de interação contornam as limitações tecnológicas da ferramenta de reconhecimento de voz Sphinx-4, fazendo com que a plataforma compreenda sempre comandos enunciados através destes mecanismos alternativos, o que se poderá traduzir numa maior correção por parte do novo modelo de interação em relação ao antigo.

Em termos de rapidez a executar tarefas, como se pode observar na Tabela 6.3, os participantes demoraram em média menos 5 segundos a terminar a mesma tarefa no modelo de interação novo face ao modelo antigo. Embora se trate de uma diferença pequena, esta é escalável para modelos maiores que exijam que o utilizador enuncie mais comandos, uma vez que ao associar estes a gestos e sons não-vocais, a plataforma irá compreender mais rapidamente os mesmos, e quantos mais comandos forem enunciados destas formas, mais tempo será poupado. Como já foi enunciado, utilizar estes mecanismos do novo protocolo de interação tem também a vantagem de contornar as limitações tecnológicas atuais da plataforma em relação ao reconhecimento de voz, uma vez que durante a experiência empírica, estes mecanismos apresentaram uma consistência muito superior à do reconhecimento de voz. A rapidez da execução de tarefas poderá também ser aumentada caso o utilizador aumente a velocidade da voz do sintetizador, que é uma das possibilidades no novo modelo de interação, o que naturalmente acelerá todo o processo de modelação e poderá ser uma ferramenta muito útil uma vez que o utilizador se habitue a utilizar a plataforma.

É possível concluir que os participantes da experiência compreenderam bem o funcionamento do novo modelo de interação, desde a criação do protocolo de interação até à utilização do novo modelo como utilizadores finais. 90% dos participantes conseguiu identificar qual o protocolo de interação errado quando lhes foi apresentado dois protocolos distintos. Assim é possível concluir que os passos necessários para realizar determinadas tarefas na plataforma são intuitivos e de rápida aprendizagem. Todos os utilizadores acharam fácil a forma de adicionar/editar novos gestos/sons não-vocais ao protocolo de interação, tendo todos conseguido realizar esta tarefa rapidamente, completando assim o protocolo de interação que lhes foi apresentado. Assim, concluímos que os protocolos de interação do novo modelo são fáceis de completar e de evoluir, uma vez que os utilizadores conseguem dar rapidamente um novo contexto a um gesto/som não-vocal apenas alterando as propriedades dos mesmos. Como os participantes, de uma forma geral, responderam corretamente às perguntas relativas à compreensão do exercício de modelação no paradigma de tabela, enunciadas no capítulo anterior, podemos concluir que este novo paradigma permite uma leitura e interpretação acessíveis, estando inerente a si uma rápida aprendizagem do seu funcionamento.

A usabilidade do novo modelo de interação foi suportada pela pontuação obtida no

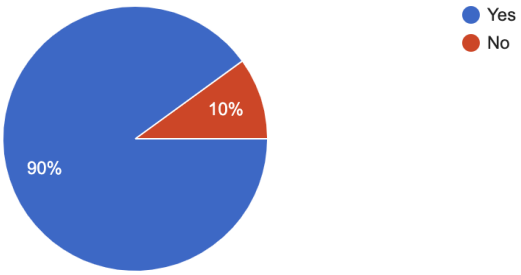


Figura 6.8: Respostas dos utilizadores em relação à existência de um *chatbot* nas plataformas de modelação

questionário SUS, representado na Figura 6.1, uma vez que este obteve uma pontuação de 80,75 e que se considera que sistemas com valores superiores a 68 têm uma boa usabilidade [12]. Uma vez que todos os participantes da experiência responderam que prefiram modelar com o novo modelo de interação do que com o antigo, conclui-se assim que o novo modelo de interação influencia a percepção de usabilidade dos utilizadores em relação ao modelo antigo.

Tendo em conta esta análise dos dados recolhidos é possível deduzir que o novo modelo de interação, devido às suas características de personalização, é uma melhor opção para exercer a atividade de modelação face ao modelo antigo.

6.4.1 Sugestões de implementação

Os participantes da experiência empírica tiveram também oportunidade de sugerir algumas implementações que poderiam melhorar a experiência do utilizador e tornar a interação mais rápida.

Foi sugerido que a voz do sintetizador não fosse tão artificial, um utilizador sugeriu que uma voz mais "humana" iria melhorar bastante o *feedback* dado pela plataforma na medida em que por vezes com a voz atual é difícil perceber o que é dito.

Foi também referido que a compreensão da voz do utilizador por parte da ferramenta por vezes não era a melhor e sugeriu-se encontrar uma ferramenta que realizasse um melhor reconhecimento de voz.

Outra sugestão consistia na existência de um *chatbot* que substituísse o atual modo de ajuda da plataforma. Um *chatbot* assistente que pudesse ser contactado por via de texto ou voz e respondesse às questões do utilizador, em vez de um modo estático de ajuda. Quando questionados, no questionário de perfil, se achavam que a possibilidade de existir um *chatbot* numa plataforma de modelação para navegar nos modelos e pedir ajuda seria útil para o utilizador, 90% respondeu que sim, como se pode observar na figura 6.8

6.5 Ameaças à validade

Existem algumas ameaças que colocam em risco a avaliação feita através da experiência empírica ao novo modelo de interação da plataforma Model-By-Voice.

No que diz respeito aos testes realizados com utilizadores existiram algumas limitações. Abaixo encontra-se a lista de ameaças aplicáveis à validação da experiência segundo *Cook* e *Campbell* [14].

6.5.1 Valididade de conclusão

O número de participantes que efetuaram a experiência de teste foi reduzido. Devido à situação de pandemia que decorria durante a elaboração desta dissertação, não foi possível realizar um maior número de testes.

Não foram testadas todas as novas funcionalidades adicionadas ao novo modelo de interação da plataforma, uma vez que tal resultaria em sessões de teste muito longas. No entanto, foram escolhidas funcionalidades consideradas fundamentais para inferir a usabilidade essencial do sistema atual, deixando-se as seguintes para um segundo estudo mais alargado.

6.5.2 Validade interna

Os participantes podem ter sido afetados pelo decorrer do tempo durante a experiência. Podem ter sofrido efeitos negativos como cansaço, dada a duração da experiência, ou positivos como aprendizagem. Esta ameaça foi minimizada, uma vez que se avisou os participantes da duração estimada que a experiência duraria. Assim é possível concluir que estes estavam mentalmente preparados para a duração da mesma. Os testes foram também realizados, na sua maioria, ao início do dia, o que por sua vez permite inferir que os participantes estariam mentalmente mais ativos e menos sujeitos a manifestar cansaço.

6.5.3 Validade de construção

Ao responderem às perguntas do inquérito realizado após a experiência, existe a possibilidade dos participantes terem tentado inferir os resultados esperados às questões, sem terem a certeza das respostas para as mesmas. Este risco foi menorizado pelo facto da experiência ter sido realizada, em grande parte, com participantes desconhecidos, não estando assim tão propensos a não terem um julgamento justo.

6.5.4 Validade externa

Não foram realizados testes com invisuais. Uma vez não se ter encontrado nenhum invisual com conhecimentos de Engenharia Informática que se disponibilizasse para realizar a experiência de teste, não foi possível avaliar a prestação da ferramenta para este grupo restrito de utilizadores. Contudo os participantes, durante a experiência, apenas estavam

a ver o guião da mesma e o painel de gestos (quando este foi requisitado). Como os participantes interagiam com a plataforma apenas através da voz, sons não-vocais e gestos desenhados, não estando a visualizar o *feedback* gráfico/textual da mesma, esta ameaça foi menorizada.

Colmatar os aspetos anteriormente mencionados é algo que poderá ser reforçado numa futura repetição deste estudo empírico com um número maior de participantes.

6.6 Conclusões

Após realizarmos as experiências de teste com alguns utilizadores foram encontrados resultados positivos quanto à usabilidade. Os utilizadores consideraram que a inclusão de sons não-vocais, gestos e paradigma de tabela na modelação fazia sentido e permitia um melhor desempenho da plataforma em certos aspetos. A possibilidade de gerar um protocolo de interação personalizado para cada utilizador revelou aumentar a produtividade dos utilizadores e a sua compreensão da plataforma na medida em que esta se adapta aos seus conhecimentos de modelação/necessidades.

De forma a se validar ainda mais o novo modelo de interação, teriam de ser realizados testes numa escala maior, com uma população de teste com maior disparidade de conhecimentos e dificuldades, nomeadamente visuais. O trabalho desenvolvido, o qual foi testado num número reduzido de participantes sem incapacidades visuais funciona, assim, como um estudo piloto para o novo modelo de interação da plataforma Model-By-Voice, que apresenta indicadores positivos e serve de base para um estudo alargado.

Capítulo 7

Conclusão

De forma a concluir esta dissertação, este capítulo resume as contribuições para a plataforma Model-By-Voice, através do novo modelo de interação, as limitações e desafios encontrados ao longo do processo e o trabalho que pode ser realizado futuramente de forma a melhorar alguns aspetos do modelo de interação desta plataforma.

7.1 Resumo

Esta dissertação teve como finalidade a reformulação do modelo de interação da plataforma Model-By-Voice, que se verificou ser pouco prático e moroso devido ao mesmo ser fixo. O novo modelo de interação é dotado agora da possibilidade de personalização das necessidades de cada utilizador, sendo este adequado às necessidades visuais e conhecimentos de modelação do mesmo. A plataforma é agora dotada da noção de "utilizador", guardando as preferências de cada utilizador para que estas estejam disponíveis quando estes fizerem *log-in* com o seu *username*. A ferramenta integra agora, para além de mecanismos de reconhecimento e sintetização de voz, mecanismos de reconhecimento e reprodução de sons não-vocais, como palmas ou assobios, reconhecimento de gestos desenhados, bem como a possibilidade de modelar diagramas através de um paradigma de tabela. Foram incluídos novos comandos com intuito de acelerar a modelação dos diagramas na ferramenta, como os comandos de copiar, colar, cortar e utilizar padrões guardados previamente que podem ser reutilizados para criar diagramas com as mesmas características do padrão escolhido. Por fim, o utilizador pode ainda optar por personalizar o sintetizador a seu gosto, podendo escolher o timbre, volume e velocidade da voz do mesmo. Assim, podemos afirmar que o objetivo de tornar o modelo de interação da plataforma Model-By-Voice mais flexível e personalizável foi atingido. O modelo de interação da plataforma Model-By-Voice encontra-se, agora, melhorado e permite uma interação

mais eficiente.

Relativamente ao conteúdo e estrutura do documento, o Capítulo 1 dá a conhecer a temática da dissertação, fazendo uma breve introdução ao problema de que esta trata e dos objetivos gerais da mesma. No Capítulo 2, são introduzidos alguns conceitos acerca de Engenharia de Modelação e no Capítulo 3 é apresentado o estado da arte onde são discutidas algumas ferramentas de interação. Por sua vez, no Capítulo 4, são enunciados trabalhos e ferramentas que se enquadram no âmbito da modelação por voz, sons não-vocais e gestos, bem como trabalhos relevantes do ponto de vista de adaptação de diversas plataformas aos utilizadores. Este capítulo representa então um estudo do trabalho que tem vindo a ser desenvolvido na personalização de plataformas e de mecanismos de interação relevantes no contexto do problema desta dissertação. No Capítulo 5 é explicada detalhadamente a arquitetura do novo modelo de interação da plataforma Model-By-Voice, desenvolvido no âmbito desta dissertação, assim como as tecnologias utilizadas para desenvolver o mesmo e uma breve discussão das decisões tecnológicas tomadas. O Capítulo 6 foi destinado a enunciar cenários de aplicação, onde são descritos os diferentes tipos de utilizador que se prevê utilizarem a plataforma e como o novo modelo de interação se aplica e ajusta a cada um deles. No Capítulo 7, é apresentada a experiência de avaliação da usabilidade da solução, onde se submeteu um conjunto de utilizadores a certos testes, tanto no novo modelo de interação como no antigo, desenvolvidos de forma a averiguar a prestação da solução proposta. Por fim, no Capítulo 8 foram discutidos os resultados obtidos na experiência empírica descrita no capítulo anterior.

7.2 Limitações, desafios e trabalho futuro

Relativamente a questões tecnológicas, a principal limitação encontrada foi a dificuldade de conectar a ferramenta à API do Google Cloud Speech-To-Text. Assim só foi possível utilizar a Sphinx-4, o que obriga os utilizadores a utilizarem apenas as palavras pertencentes à gramática da plataforma para as utilizar como variáveis nas DSLs que pretendem modelar.

Embora tenham existido problemas de integração nas tecnologias inicialmente pensadas para reconhecimento e sintetização de sons não-vocais e reconhecimento de gestos, foram facilmente encontradas alternativas que garantiam também bons resultados e uma boa integração na ferramenta.

Durante a experiência empírica, os utilizadores encontraram limitações como o reconhecimento de voz não ser totalmente fiável e a voz do sintetizador por vezes não ser compreensível.

Considerando o *feedback* dado pelos utilizadores após concluírem a experiência empírica de utilização do novo modelo de interação da plataforma Model-By-Voice, conclui-se que existem aspetos da interação com a mesma que ainda poderão ser melhorados de futuro. Foi sugerido que se encontrasse uma forma de tornar a voz mais "humana" de forma a ser mais compreensível. Tal poderá ser alcançado integrando a tecnologia da

Google Cloud Text-To-Speech que permite escolher diversos tipo de vozes de diversas nacionalidades que são emulações muito próximas da voz humana.

Foi também sugerido que se construísse um *chatbot* dentro da plataforma, com o qual se pudesse interagir através de texto ou voz, para se colocar questões sobre o funcionamento da plataforma em vez do modo de ajuda estático atual.

Outra melhoria que se poderá elaborar de futuro seria a DSL de protocolos de interação ter em conta mais otimizações relacionadas com a DSL a modelar, que não só gestos e sons não-vocais mais adequadas à mesma, mas também a geração de atalhos que levem a um menor esforço cognitivo da parte de utilizadores. Por exemplo, no caso dos *"Data-Flow Diagram"* existe apenas um tipo de ligação que é o *"Data Flow"*, então uma melhoria que se poderia elaborar seria para este tipo de diagrama, por exemplo, quando se cria uma ligação, não ser perguntado qual o tipo de ligação que se pretende criar, uma vez que neste tipo de diagrama apenas existe um tipo de ligação.

Bibliografia

[1] *Accessible SpreadSheets*. https://support.office.com/en-us/article/make-your-excel-documents-accessible-to-people-with-disabilities-6cc05fc5-1314-48b5-8eb3-683e49b3e593?ui=en-US&rs=en-US&ad=US. Accessed: 2020-01-05.

[2] *Activity Diagrams*. https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/. Accessed: 2020-02-17.

[3] V. Amaral, A. Cicchetti e R. Deshayes. "A Multiparadigm Approach to Integrate Gestures and Sound in the Modeling Framework". Em: vol. 1112. Set. de 2013.

[4] *Amazon Lex*. https://aws.amazon.com/pt/lex/. Accessed: 2020-01-08.

[5] A. Armaly, P. Rodeghero e C. McMillan. "A Comparison of Program Comprehension Strategies by Blind and Sighted Programmers". Em: *IEEE Transactions on Software Engineering* 44.8 (2018), pp. 712–724. ISSN: 2326-3881. DOI: 10.1109/TSE.2017.2729548.

[6] A. Armaly. "Making linux accessible for the visually impaired with speakup". Em: *Linux Journal* 2005.140 (2005), p. 6.

[7] J. M. Atlee, R. France, G. Georg, A. Moreira, B. Rumpe e S. Zschaler. "Modeling in Software Engineering". Em: *29th International Conference on Software Engineering (ICSE'07 Companion)*. 2007, pp. 113–114. DOI: 10.1109/ICSECOMPANION.2007.53.

[8] C. M. Baker, L. R. Milne e R. E. Ladner. "StructJumper: A Tool to Help Blind Programmers Navigate and Understand the Structure of Code". Em: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: Association for Computing Machinery, 2015, 3043–3052. ISBN: 9781450331456. DOI: 10.1145/2702123.2702589.

[9] M. Bastéa-Forte, R. Yeh e S. R. Klemmer. "Pointer: Multiple Collocated Display Inputs Suggests New Models for Program Design and Debugging". Em: *Stanford University HCI, USA* (2007).

[10] D. Black, E. J. Rapos e M. Stephan. "Voice-Driven Modeling: Software Modeling Using Automated Speech Recognition". Em: *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE. 2019, pp. 252–258.

[11] P. Blenkhorn e D. Evans. "Using speech and touch to enable blind people to access schematic diagrams". Em: *Journal of Network and Computer Applications* 21.1 (1998), pp. 17 –29. ISSN: 1084-8045. DOI: 10.1006/jnca.1998.0060.

[12] J. Brooke. *SUS: a "quick and dirty'usability*. Vol. 194. London: Taylor e Francis, 1996, pp. 189–194.

[13] M. Charfuelan. "MARY TTS HMMbased voices for the Blizzard Challenge 2012". Em: *Blizzard Challenge Workshop*. Vol. 2012. 2012.

[14] T. D. Cook, D. T. Campbell e A. Day. *Quasi-experimentation: Design & analysis issues for field settings*. Vol. 351. Houghton Mifflin Boston, 1979.

[15] G. Cuhac. *Design and Implementation of a Wireless Home Automation Control System with Speech Recognition*. 2013.

[16] *Data-Flow Diagrams*. https://www.visual-paradigm.com/guide/data-flow-diagram/what-is-data-flow-diagram/. Accessed: 2020-01-05.

[17] P. R. Davis. *Systems and Methods for Generating an Interactive Avatar Model*. US Patent App. 13/852,126. 2013.

[18] *Dragon Speech Recognition Engine*. https://www.nuance.com/dragon.html. Accessed: 2019-12-28.

[19] *Eclipse*. https://www.eclipse.org. Accessed: 2020-01-15.

[20] *Eclipse Modeling Framework*. https://www.eclipse.org/modeling/emf/. Accessed: 2020-02-06.

[21] *EGL*. https://www.eclipse.org/epsilon/doc/egl/. Accessed: 2020-02-01.

[22] AToMPM. https://atompm.github.io. Accessed: 2020-09-29.

[23] Meta Programming System. https://www.jetbrains.com/mps/. Accessed: 2020-09-29.

[24] Protótipo do novo modelo de interação da plataforma Model-By-Voice. https://github.com/jcavalho/newinteractionmodelmodelbyvoice. Accessed: 2020-11-30.

[25] StarUML. https://staruml.io. Accessed: 2020-09-29.

[26] Vídeo de execução do teste de exemplo do novo modelo de interação da plataforma Model-By-Voice. https://rb.gy/urhohw. Accessed: 2020-11-30.

[27] *EOL*. https://www.eclipse.org/epsilon/doc/eol/. Accessed: 2020-02-01.

[28] *Epsilon*. https://www.eclipse.org/epsilon/. Accessed: 2020-01-13.

[29] *eSpeak*. http://espeak.sourceforge.net. Accessed: 2020-01-05.

[30] *EuGenia*. https://www.eclipse.org/epsilon/doc/eugenia/. Accessed: 2020-01-13.

[31] *Excel Spreadsheets For The Blind*. https://www.perkinselearning.org/technology/digital-transitions/all-about-excel-spreadsheets. Accessed: 2020-01-02.

[32] M. Fowler. *Domain-specific languages*. Pearson Education, 2010.

[33] *FreeTTS*. https://freetts.sourceforge.io. Accessed: 2020-02-17.

[34] M. Gamboa e E. Syriani. "Improving user productivity in modeling tools by explicitly modeling workflows". Em: *Software & Systems Modeling* 18.4 (2019), pp. 2441–2463. DOI: 10.1007/s10270-018-0678-1.

[35] *Google Cloud Speech*. cloud.google.com/speech-to-text/. Accessed: 2020-02-17.

[36] *Google Natural Language*. https://cloud.google.com/natural-language/. Accessed: 2020-01-08.

[37] *Graphical Modeling Framework Documentation*. https://wiki.eclipse.org/Graphical_Modeling_Framework/Documentation. Accessed: 2020-09-29.

[38] K. Hall, E. Cho, C. Allauzen, F. Beaufays, N. Coccaro, K. Nakajima, M. Riley, B. Roark, D. Rybach e L. Zhang. "Composition-based on-the-fly rescoring for salient n-gram biasing". Em: *INTERSPEECH* (2015).

[39] B. Haynor e P. Aleksic. "Dialog-aware language models for speech recognition". Em: *Technical Disclosure Commons* (2019).

[40] *iGesture*. http://www.igesture.org. Accessed: 2020-01-02.

[41] A. Jana. *Kinect for windows SDK programming guide*. Packt Publishing Ltd, 2012.

[42] *JAWS*. https://www.freedomscientific.com/products/software/jaws/. Accessed: 2020-01-02.

[43] A. King, P. Blenkhorn, D. Crombie, S. Dijkstra, G. Evans e J. Wood. "Presenting UML software engineering diagrams to blind people". Em: *International Conference on Computers for Handicapped Persons*. Springer. 2004, pp. 522–529.

[44] I. Kurtev, J. Bézivin, F. Jouault e P. Valduriez. "Model-Based DSL Frameworks". Em: *Companion to the 21st ACM SIGPLAN Symposium on Object-Oriented Programming Systems, Languages, and Applications*. OOPSLA '06. Portland, Oregon, USA: Association for Computing Machinery, 2006, 602–616. ISBN: 159593491X. DOI: 10.1145/1176617.1176632.

[45] M. J. Landau, B. Y. Choo e P. A. Beling. "Simulating Kinect infrared and depth images". Em: *IEEE transactions on cybernetics* 46.12 (2016), pp. 3018–3031. DOI: 10.1109-TCYB.2015.2494877.

[46] J. Lopes, J. Cambeiro e V. Amaral. "ModelByVoice-towards a general purpose model editor for blind people." Em: *MODELS Workshops*. 2018, pp. 762–769.

[47] J. Lopes. "Plataforma de modelação para pessoas portadoras de deficiência visual". Tese de mestrado. Faculdade de Ciências e Tecnologias, Universidade Nova de Lisboa, 2018.

[48] *Markov Model*. https://en.wikipedia.org/wiki/Markov_model. Accessed: 2020-01-02.

[49] *MetaEdit*. https://www.metacase.com/mep/. Accessed: 2020-09-29.

[50] A. H. Michaely, M. Ghodsi, Z. Wu, J. Scheiner e P. Aleksic. "Unsupervised context learning for speech recognition". Em: *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2016, pp. 447–453.

[51] *Microsoft Bot Framework*. https://dev.botframework.com. Accessed: 2020-01-08.

[52] *Microsoft Kinect*. https://developer.microsoft.com/pt-pt/windows/kinect/. Accessed: 2020-02-17.

[53] M. Modzelewski e E. B. Kaiser. "Hand gesture recognition interface for visually impaired and blind people". Em: *2012 IEEE/ACIS 11th International Conference on Computer and Information Science*. IEEE. 2012, pp. 201–206.

[54] D. Moody. "The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering". Em: *IEEE Press* 35.6 (2009), pp. 756–779. DOI: 10.1109/TSE.2009.67.

[55] *Mouse Gesture Recognition with Hidden Markov Model*. http://blog.gtiwari333.com/2011/07/mouse-gesture-recognition-with-hidden.html. Accessed: 2020-09-28.

[56] *Musicg*. https://github.com/loisaidasam/musicg. Accessed: 2020-09-28.

[57] R. Nacheva. "Current Perspectives on Linux Accessibility Tools for Visually Impaired Users". Em: *Economics and computer science* 2 (2019), pp. 6–11.

[58] *Non-Speech Recognition Sphinx-4*. https://stackoverflow.com/questions/4098279/non-speech-noise-or-sound-recognition-software. Accessed: 2020-02-17.

[59] *NV Access*. https://www.nvaccess.org. Accessed: 2020-01-02.

[60] V. Pelechano, M. Albert, J. Muñoz e C. Cetina. "Building Tools for Model Driven Development. Comparing Microsoft DSL Tools and Eclipse Modeling Plug-ins." Em: *DSDM*. 2006.

[61] S. Pérez-Soler, M. González-Jiménez, E. Guerra e J. de Lara. "Towards Conversational Syntax for Domain-Specific Languages using Chatbots". Em: *Journal of Object Technology* 18.2 (2019).

[62] H. Petrie, C. Schlieder, P. Blenkhorn, G. Evans, A. King, A.-M. O'Neill, G. T. Ioannidis, B. Gallagher, D. Crombie, R. Mager et al. "Tedub: A system for presenting and exploring technical drawings for blind people". Em: *International Conference on Computers for Handicapped Persons*. Springer. 2002, pp. 537–539. DOI: 10.1007/3-540-45491-8_102.

[63] B. Puype. "Extending the igesture framework with multimodal gesture interaction functionality". Tese de mestrado. Vrije Universiteit Brussel, 2010.

[64] T. Raman. "Emacspeak–A speech interface". Em: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (1996).

[65] D. Rubine. *Specifying gestures by example*. Vol. 25. 4. ACM, 1991.

[66] *SAS Graphic Accelerator*. https://support.sas.com/software/products/graphics-accelerator/samples/index.html. Accessed: 2020-01-02.

[67] D. C. Schmidt. "Model-driven engineering". Em: *Computer-IEEE Computer Society* 39.2 (2006), p. 25.

[68] *Sequence Diagrams*. https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/. Accessed: 2020-02-17.

[69] H. R. Sharifi, M. Mohsenzadeh e S. M. Hashemi. "CIM to PIM Transformation: An Analytical Survey". Em: *IJ of Computer Technology & Applications* 3 (2012), pp. 791–796.

[70] F. Shull e J. Singer. *Book Guide To Advanced Empirical Software*. Springer, 2008.

[71] B. Signer, M. C. Norrie e U. Kurmann. "iGesture: A Java framework for the development and deployment of stoke-based online Gesture recognition algorithms". Em: *Technical report/Swiss Federal Institute of Technology Zurich, Department of Computer Science* 561 (2011).

[72] *Sirius*. https://www.eclipse.org/sirius/overview.html. Accessed: 2020-09-29.

[73] F. Soares, J. a. Araújo e F. Wanderley. "VoiceToModel: An Approach to Generate Requirements Models from Speech Recognition Mechanisms". Em: *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. SAC '15. Salamanca, Spain: Association for Computing Machinery, 2015, 1350–1357. ISBN: 9781450331968. DOI: 10.1145/2695664.2695724.

[74] *Sonic*. https://github.com/waywardgeek/sonic. Accessed: 2020-09-28.

[75] *Sphinx-4*. https://cmusphinx.github.io/wiki/tutorialsphinx4/. Accessed: 2020-02-17.

[76] *Stack Overflow*. https://pt.stackoverflow.com. Accessed: 2020-02-06.

[77] *Stack Overflow Survey 2019*. https://insights.stackoverflow.com/survey/2019. Accessed: 2020-02-06.

[78] T. Stahl, M. Voelter e K. Czarnecki. *Model-driven software development: technology, engineering, management*. John Wiley & Sons, Inc., 2006.

[79] S. Swigart. "Easily write custom gesture recognizers for your Tablet PC applications". Em: *Tablet PC Technical Articles* (2005).

[80] D. O. Tanguay. "Hidden Markov models for gesture recognition". Tese de doutoramento. Massachusetts Institute of Technology, 1995.

[81] *Text-prompted Remote Speaker Authentication*. http://blog.gtiwari333.com/2010/12/text-prompted-remote-speaker.html. Accessed: 2020-09-28.

[82] *Training an acoustic model for CMUSphinx*. https://cmusphinx.github.io/wiki/tutorialam/. Accessed: 2020-02-17.

[83] *UI Sounds: From Zero To Hero*. https://icons8.com/articles/ui-sounds/. Accessed: 2020-09-29.

[84] J. C. Vidal, P. Carreira, V. Amaral, J. Aguiam e J. Sousa. "Towards high-level fuzzy control specifications for building automation systems". Em: *Software and Systems Modeling* (2019), pp. 625–646. DOI: 10.1007/s10270-019-00755-8.

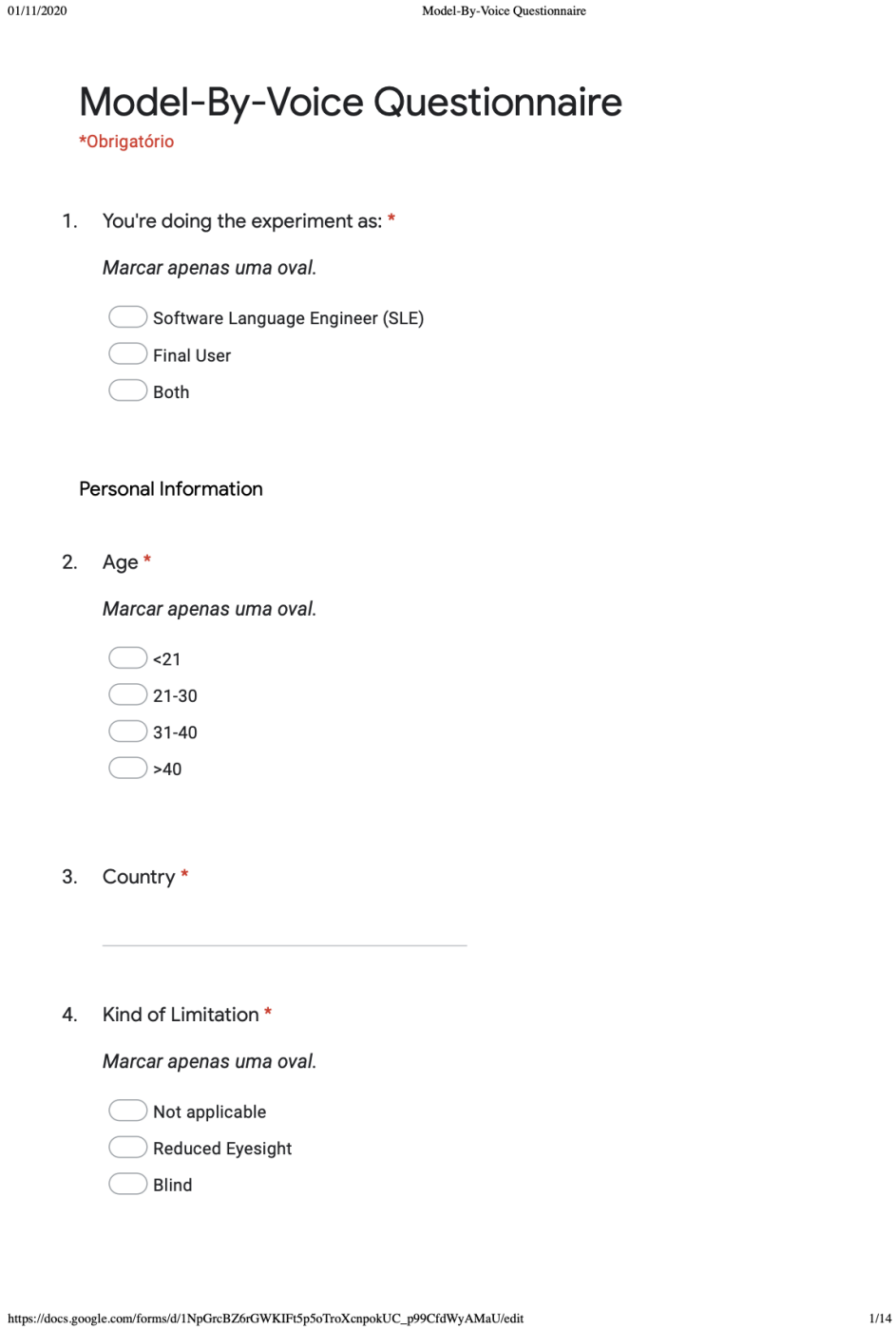
[85] W. Walker, P. Lamere e P. Kwok. "FreeTTS: a performance case study". Em: *Computer Science* (2002).

[86] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf e J. Woelfel. "Sphinx-4: A flexible open source framework for speech recognition". Em: *Computer Science* (2004).

[87] *Watson Natural Language Understanding*. https://www.ibm.com/pt-en/marketplace/natural-language-understanding. Accessed: 2020-01-08.

ANEXO I

QUESTIONÁRIO REALIZADO AOS UTILIZADORES APÓS A EXPERIÊNCIA EMPÍRICA



01/11/2020 Model-By-Voice Questionnaire

Model-By-Voice Questionnaire

*Obrigatório

1. You're doing the experiment as: *

Marcar apenas uma oval.

- ◯ Software Language Engineer (SLE)
- ◯ Final User
- ◯ Both

Personal Information

2. Age *

Marcar apenas uma oval.

- ◯ <21
- ◯ 21-30
- ◯ 31-40
- ◯ >40

3. Country *

4. Kind of Limitation *

Marcar apenas uma oval.

- ◯ Not applicable
- ◯ Reduced Eyesight
- ◯ Blind

https://docs.google.com/forms/d/1NpGrcBZ6rGWKIFt5p5oTroXcnpokUC_p99CfdWyAMaU/edit 1/14

Figura I.1: Página 1 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

5. Origin of the Limitation *

Marcar apenas uma oval.

- ◯ Not applicable
- ◯ Birth
- ◯ After Birth

6. Develops for *

Marcar apenas uma oval.

- ◯ Does not develop
- ◯ Hobby
- ◯ Work

7. Knowledge on computer science *

Marcar apenas uma oval.

- ◯ None
- ◯ Basic
- ◯ Medium
- ◯ Advanced

8. Level Of English *

Marcar apenas uma oval.

- ◯ None
- ◯ Basic
- ◯ Medium
- ◯ Advanced

Modelling

Figura I.2: Página 2 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

9. Are you familiar with software modeling engineering? *

Marcar apenas uma oval.

- Yes
- No

10. Have you built a Domain-specific language? *

Marcar apenas uma oval.

- Yes
- No

11. Are you familiar with language meta-models? *

Marcar apenas uma oval.

- Yes
- No

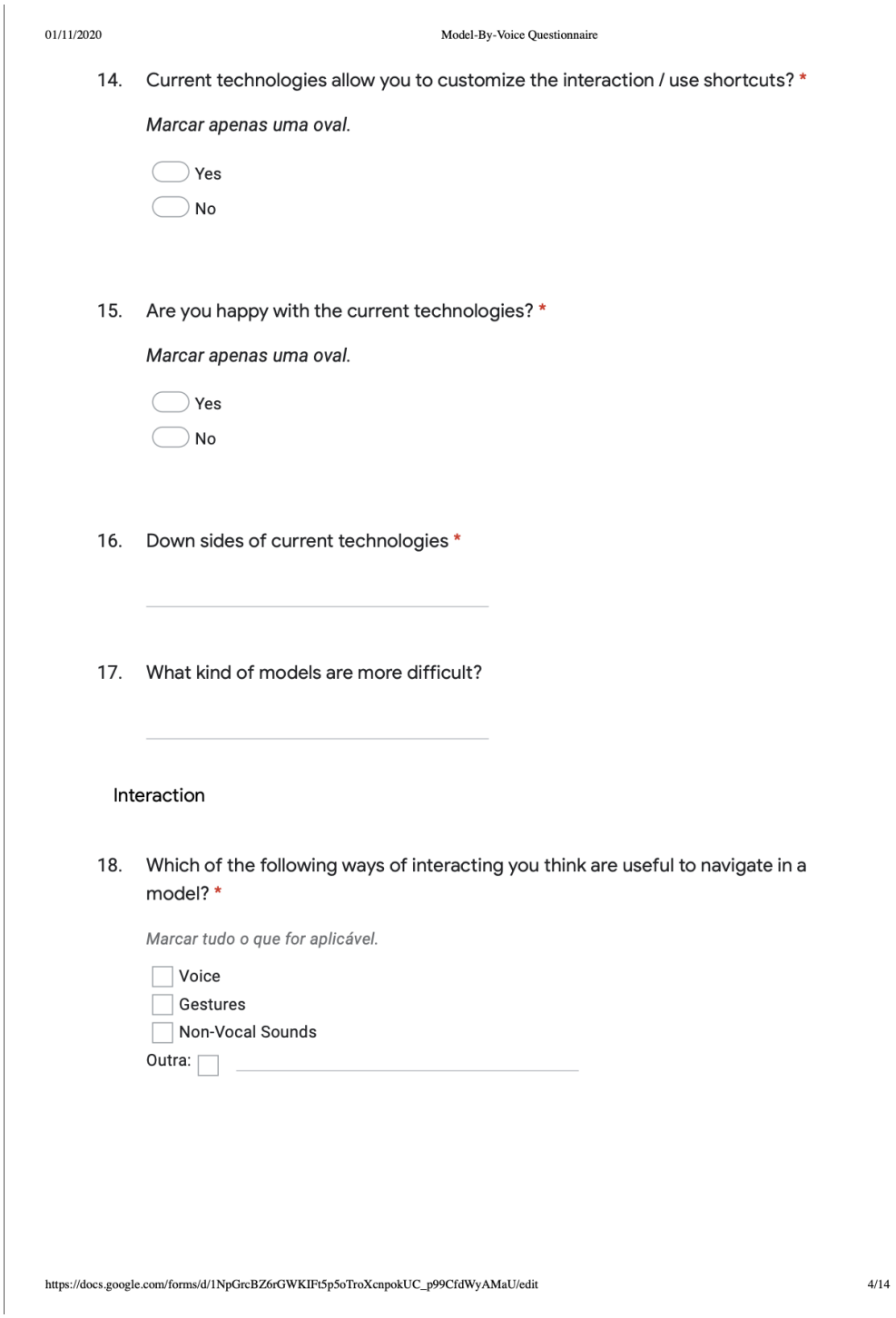
12. Which Technologies do you use to perform modelling activities? *

13. Current technologies allow you to be autonomous? *

Marcar apenas uma oval.

- Yes
- No

Figura I.3: Página 3 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice



01/11/2020 Model-By-Voice Questionnaire

14. Current technologies allow you to customize the interaction / use shortcuts? *

Marcar apenas uma oval.

Yes
No

15. Are you happy with the current technologies? *

Marcar apenas uma oval.

Yes
No

16. Down sides of current technologies *

17. What kind of models are more difficult?

Interaction

18. Which of the following ways of interacting you think are useful to navigate in a model? *

Marcar tudo o que for aplicável.

Voice
Gestures
Non-Vocal Sounds
Outra:

https://docs.google.com/forms/d/1NpGrcBZ6rGWKIFt5p5oTroXcnpokUC_p99CfdWyAMaU/edit 4/14

Figura I.4: Página 4 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

19. Do you find drawing geometrical shapes, such as squares and circles, using the computer mouse, an useful way to build diagrams? (Ex: rectangles representing classes in a class diagram) *

Marcar apenas uma oval.

- ◯ Yes
- ◯ No

20. On the context of a model editor, do you find useful the existence of a chatbot, a feature to give instructions of navigation and ask for help? *

Marcar apenas uma oval.

- ◯ Yes
- ◯ No

21. If you answered 'Yes' to the previous question, how would you like to interact with this chatbot?

Marcar tudo o que for aplicável.

- ☐ Voice
- ☐ Gestures (Hands)
- ☐ Gestures (Drawn)
- ☐ Non-Vocal Sounds

Outra: ☐ _____

22. Do you think that an interaction model based on 2D concepts (such as a table) to navigate on models would be useful even for blind people? *

Marcar apenas uma oval.

- ◯ Yes
- ◯ Yes, for people who had sight at some point and have a notion of 2D, No for the blind since birth
- ◯ Not for blind people, just for people who have sight
- ◯ No, for everyone

Figura I.5: Página 5 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

23. Do you use Excel tables as an information processor aid tool? *

Marcar apenas uma oval.

- Yes
- No

24. If answered 'Yes' to the previous question, do you think the use of the Excel tables would be a good feature to navigate on the models?

Marcar apenas uma oval.

- Yes
- No

25. Would you consider using an application for modeling that relies on external devices such as a sensor to track gestures or ReSpeakers? *

Marcar apenas uma oval.

- Yes
- No

Experience Questionnaire

Figura I.6: Página 6 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

26. Which of the following interaction protocols is WRONG? (Consider the user is using the platform for the first time) *

Marcar apenas uma oval.

1 2

27. Do you think the new flexible interaction model (with the possibility to custom the interaction, table paradigm, non-vocal sounds, gestures, shortcuts [like copy, cut and use of patterns]) is more useful for the User? *

Marcar apenas uma oval.

Yes
No

28. Do you think it's easy to add new sound/gestures to the interaction protocol? *

Marcar apenas uma oval.

Yes
No

Figura I.7: Página 7 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

29. Do you think the new interaction model offers the final users/sle enough expressiveness? *

Marcar apenas uma oval.

- Yes
- No

30. What was your favorite way of modeling? (Consider your experience in the Traffic Light exercise) *

Marcar apenas uma oval.

- Voice
- Non-Vocal Sound
- Gestures

31. Would you rather use the new interaction model or the old one? (Consider your experience in the Traffic Light exercise) *

Marcar apenas uma oval.

- New interaction model
- Old interaction model

32. If you answered "Old" to the previous question" please describe how would you improve the new interaction model.

Figura I.8: Página 8 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

33. Did you prefer modeling on new the table paradigm or the graph paradigm *

Marcar apenas uma oval.

- Table paradigm
- Graph paradigm

Consider the 'FileSystem' DSL and the given example

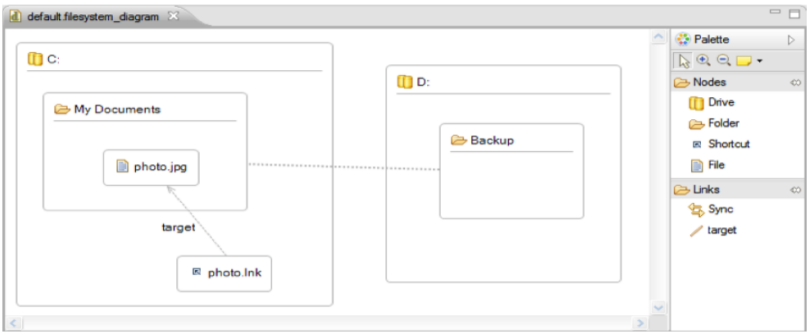
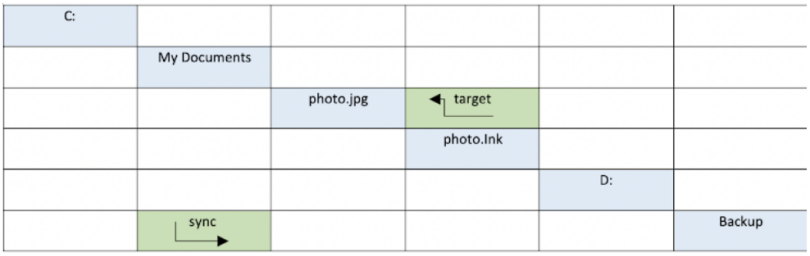


Figura I.9: Página 9 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

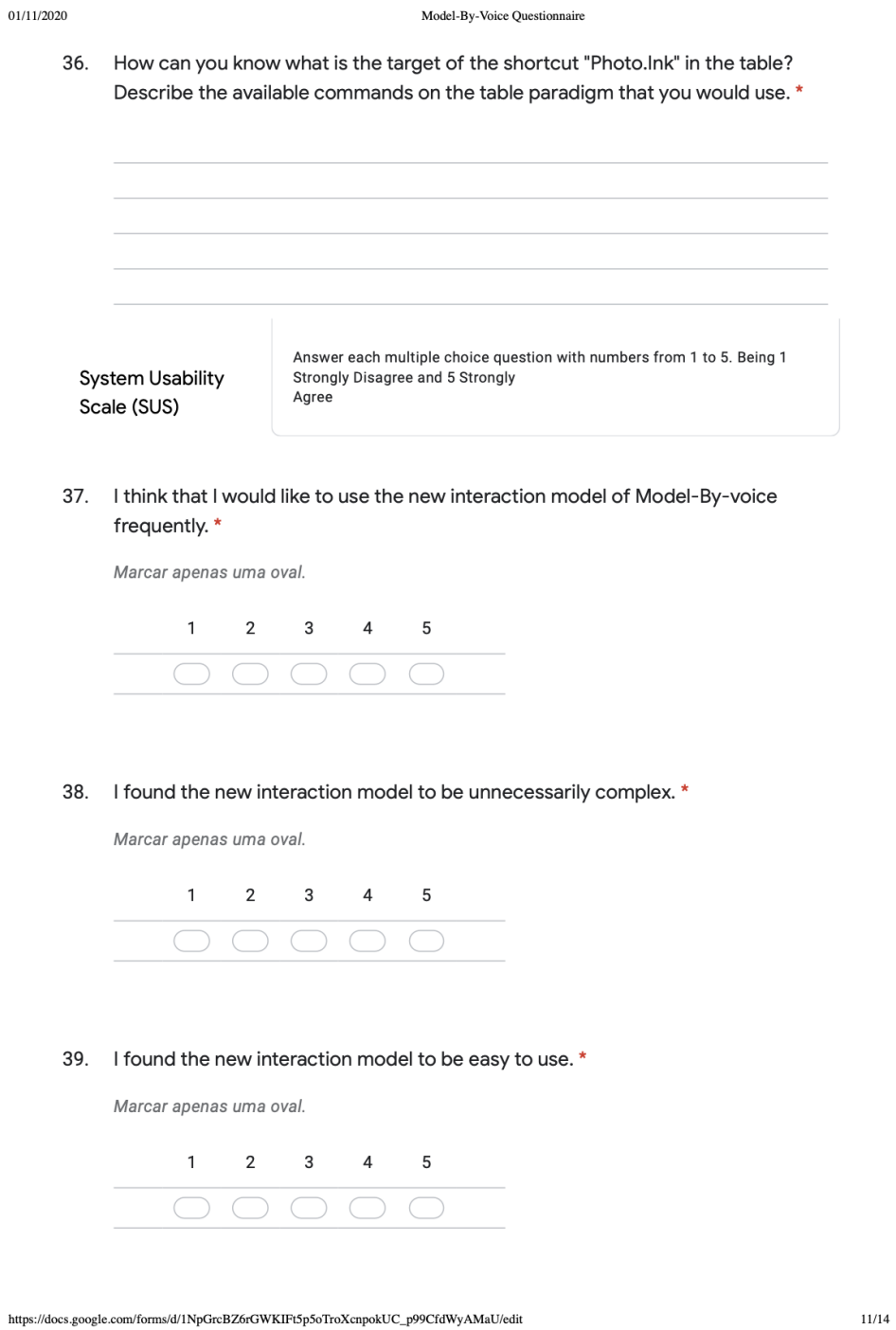
In he third task you were asked to perform, you used the new table paradigm. This following image represents the result of modeling in the table paradigm the above 'FileSystem' example, as you did in the task. In this paradigm the nodes are placed on the main diagonal of the table and the links are placed in the cell which coordinates are the number of the row of the target node and the column of the current node. In this paradigm you can navigate through cells with the keyboard arrows and navigate to a specific cell by naming its coordinates. If you're in a cell and go up, for example, and there is a link in that cell, you'll be transported to the exit node of that link. You can also order the nodes alphabetically and list all the elements of the diagram. Considering only the table image and the model you built , to answer the following 3 questions.



34. Please list all the links on the diagram (Ex: My Documents -> photo.jpg) *

35. Is it possible to know how many nodes are synced with a certain node? Describe How. *

Figura I.10: Página 10 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice



01/11/2020 Model-By-Voice Questionnaire

36. How can you know what is the target of the shortcut "Photo.lnk" in the table? Describe the available commands on the table paradigm that you would use. *

System Usability Scale (SUS)

Answer each multiple choice question with numbers from 1 to 5. Being 1 Strongly Disagree and 5 Strongly Agree

37. I think that I would like to use the new interaction model of Model-By-voice frequently. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

38. I found the new interaction model to be unnecessarily complex. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

39. I found the new interaction model to be easy to use. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

https://docs.google.com/forms/d/1NpGrcBZ6rGWKIFt5p5oTroXcnpokUC_p99CfdWyAMaU/edit 11/14

Figura I.11: Página 11 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

40. I think that I could not use the new interaction model without the support of a technical person. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

41. I found the new functions in the new interaction model were well integrated. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

42. I thought there was a lot of inconsistency in new interaction model. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

43. I would imagine that most people would learn to use the new interaction model very quickly. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

Figura I.12: Página 12 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

44. I found the new interaction model very cumbersome to use. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

45. I felt very confident using the new interaction model. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

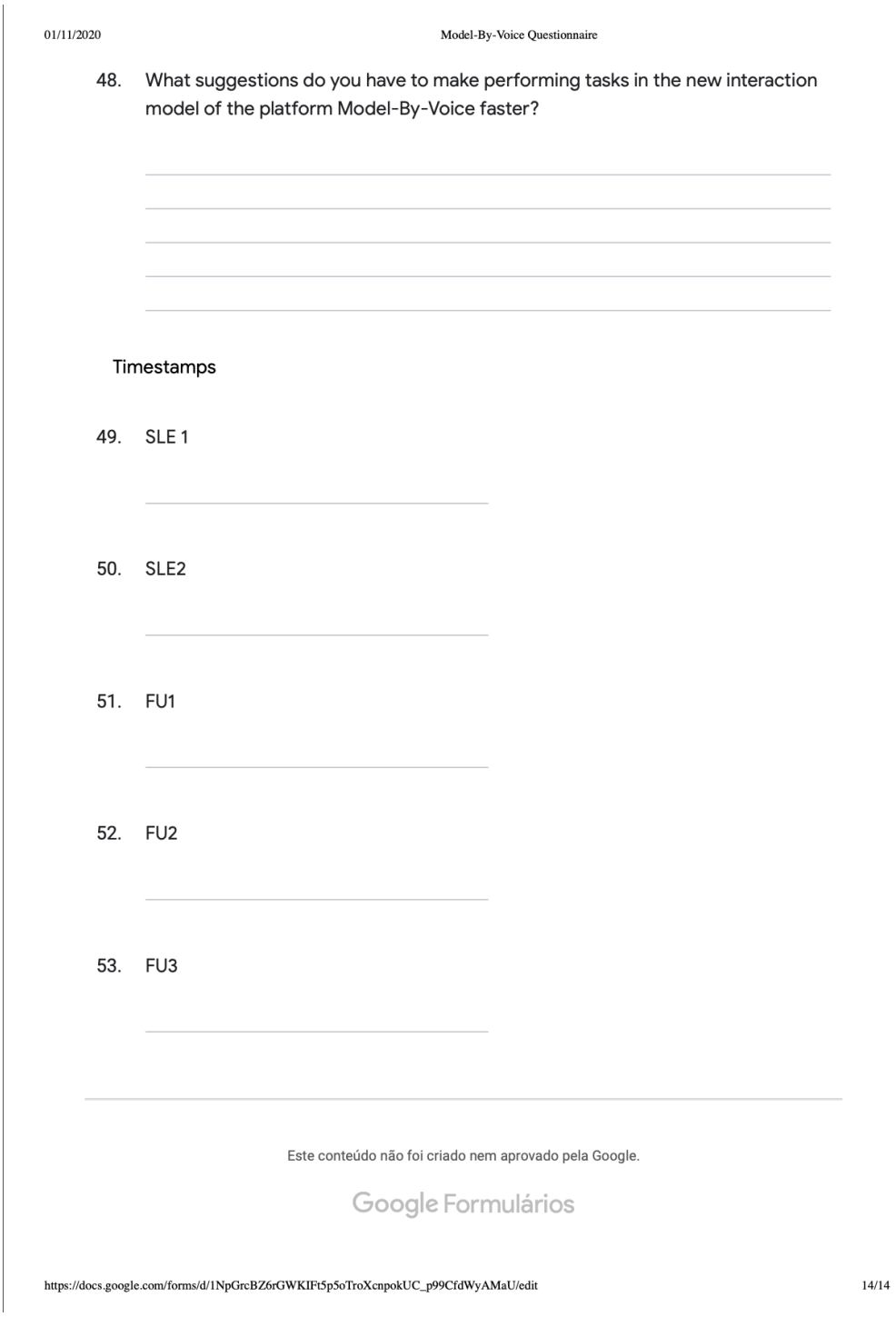
46. I could not use the new interaction model without having any knowledge of Modeling Engineering. *

Marcar apenas uma oval.

1	2	3	4	5
◯	◯	◯	◯	◯

47. What suggestions do you have to make the new interaction model of the platform Model-By-Voice more usable?

Figura I.13: Página 13 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice



48. What suggestions do you have to make performing tasks in the new interaction model of the platform Model-By-Voice faster?

Timestamps

49. SLE 1

50. SLE2

51. FU1

52. FU2

53. FU3

Figura I.14: Página 14 do questionário da experiência de teste do novo modelo de interação da plataforma Model-By-Voice

ANEXO II

GUIÃO DA EXPERIÊNCIA DE TESTE DA PLATAFORMA MODEL-BY-VOICE E MATERIAL DE APOIO

II.1 Engenheiro de Linguagens (SLE)

II.1.1 Desenhar protocolo de interação

Desenhe um protocolo de interação onde os utilizadores de nível de conhecimentos 0 e dificuldades visuais 0 apenas podem executar operações de ler e criar.

Passos:

- Criar um novo diagrama do tipo "protogocolgen19". (Na pasta protocolgen19 -> *model* -> *examples* -> por cima da pasta *examples "clickar"* no lado direito do rato -> *New* -> *Other...* -> procurar "protocolgen19 Diagram" -> dar nome ao diagrama: "nome.protocolgen19_diagram").
- Arrastar o Objeto Utilizador para a área de edição. Nas propriedades colocar 'Utilizador' no Nome; 0 no Nível de Conhecimento e 0 no Nível de Dificuldade Visual. (Figura II.1)
- Arrastar o objeto "Criar" para a área de edição. Nas propriedades colocar "Criar"no Nome; "o utilizador pode executar operações de criar na plataforma" na descrição.

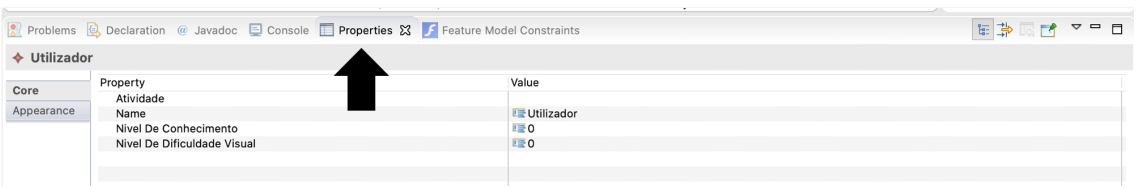


Figura II.1: Localização no editor das propriedades dos objetos.

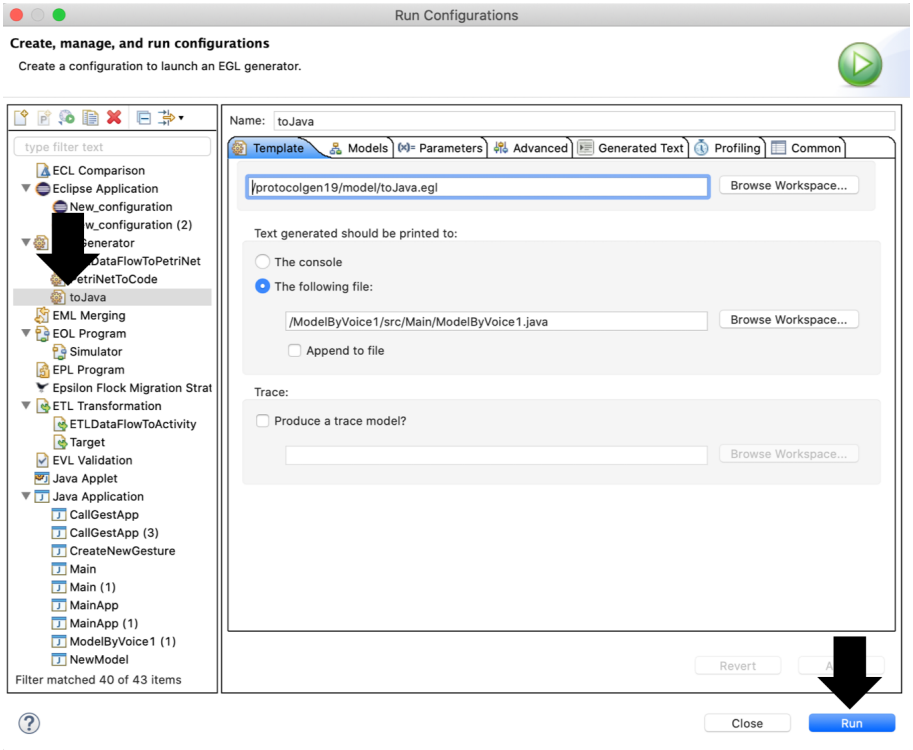


Figura II.2: Localização do ficheiro .toJava em *Run Configurations* e do botão *"Run"*.

- Arrastar o objeto "Ler" para a área de edição. Nas propriedades colocar "Ler" no Nome; "o utilizador pode executar operações de ler na plataforma" na descrição;

- Arrastar a Ligação "Atividade" ligando o objeto "Criar" com o objeto "Ler".

II.1.2 Completar protocolo de interação

Complete o protocolo de interação "completeTask.protocolgen19_diagram."

Passos:

- Adicionar "Gesto-> Arrastar objeto "Gesto" para área de edição. Nas propriedades dar o nome de *"create link"* e a descrição "downandleft".

- Adicionar "Som-> Arrastar objeto "Sons" para área de edição. Nas propriedades dar o nome de "nullname" e o *path* "/Users/joaomiguel/Desktop/SOUNDEXPERIENCE/nullName.wav".

- Gerar código Java (Na pasta protocolgen19 -> *model* -> toJava.egl -> por cima do ficheiro toJava.egl *"clickar"* no lado direito do rato -> *Run As* -> *Run Configurations...* -> no painel do lado esquerdo, escolher o ficheiro ".toJava-> *Run*) (Figura II.2)

II.2 Utilizador final

II.2.1 Traffic Lights (Modelo Antigo)

- Iniciar aplicação.
- Dizer o nome "America".
- responder *"No"* a *"Do you need Help?"*.
- Esperar que a plataforma crie parte do diagrama.
- Enunciar comando *"Create Node"*.
- Nome do novo nó: *"Yellow"*. Tipo do novo nó: "State".
- Enunciar comando *"Create Node"*.
- Nome do novo nó: *"Green"*. Tipo do novo nó: *"State"*.
- Enunciar Comando *"Create Link"*.
- target: *"Yellow"*. Tipo do *target*: *"State"*. Tipo da ligação: *"transition"*.
- Enunciar Comando *"Create Link"*.
- target: *"Red"*. Tipo do target: *"State"*. Tipo da ligação: *"transition"*.
- Executar o comando *"List Diagram"*. Responder *"yes"* a *"detail all nodes"* e *"yes"* a *"detail all links"*. - Executar comando *"exit program"*. Responder: *"Yes"* a *"Are you sure you want to stop executing?"*.

II.2.2 Traffic Lights (Modelo Novo)

- Iniciar aplicação.
- Dizer o nome "John".
- responder *"No"* a *"Do you need Help"*.
- Esperar que plataforma crie parte do diagrama.
- Executar o comando *"Copy"*.
- Responder *"node"* ao tipo de elemento a copiar.
- Responder *"Red"* ao nome do elemento a copiar.
- Responder *"Yellow"* ao nome a dar ao novo nó.
- Responder *"Yes"* a "guardar este nó como padrão".

Figura II.3: Ferramenta utilizada no reconhecimento de gestos na plataforma Model-By-Voice [55]

- Executar o comando *"use pattern"*.
- Responder *"Yellow"* ao nome do padrão a utilizar.
- Responder *"Green"* ao nome a dar ao novo nó.
- Enunciar Comando *"Gesture"*. Desenhar no painel de gestos uma linha para baixo e uma linha para a esquerda sem largar o rato(como ilustrado na Figura II.3). (*Create Link*)
- *target*: *"Yellow"*. Tipo do *target*: *"State"*. Tipo da ligação: *"transition"*.
- Enunciar Comando *"Sound"*. Reproduzir o ficheiro "SOM1". (*Create Link*)
- *target*: *"Red"*. Tipo do *target*: *"State"*. Tipo da ligação: *"transition"*.
- Executar o comando *"List Diagram"*. Responder *"yes"* a *"detail all nodes"* e *"yes"* a *"detail all links"*. - Executar comando *"exit program"*. Responder: *"Yes"* a *"Are you sure you want to stop executing?"*.

II.2.3 Modelar sistema de ficheiros

Considere a DSL de File Systems e o exemplo da Figura II.4:

- Iniciar aplicação.
- Fazer *Log-in* com o utilizador "Bond" previamente registado.
- Responder *"No"* a *"Do you need help"*.

- Esperar enquanto a plataforma cria os componentes relativos ao compartimento *drive C*. Posteriormente a tarefa irá consistir em criar os componentes relativos à *Drive D*.

- Executar o comando *"Create Compartment"*. Dar o Nome de *"Drive D"* ao compartimento. Dar o tipo *"Drive"* ao compartimento Executar o comando *"Create Compartment"*. Dar o Nome de "Backup" ao compartimento. Dar o tipo *"Folder"* ao compartimento Executar o comando *"Go To"*. *Row*: *two*|| *Column*: *two*.

- Executar o comando *"Create Link" Target*: *"backup"* | *Type of target*: *"folder"* | *type of link*: *"sync"* .

- Executar o comando *"List Diagram"*. Responder *"yes"* a *"detail all nodes"* e *"yes"* a *"detail all links"*.

- Executar comando *"exit program"*. Responder: *"Yes"* a *"Are you sure you want to stop executing?"*.

II.3 Material de apoio

Model-By-Voice é uma plataforma que permite modelar diagramas através de comandos de voz. A esta estão associados os conceitos de nós, compartimentos, atributos e ligações. Os nós são o conceito mais simples de modelação, representando qualquer entidade que se pretenda modelar. Os compartimentos são considerados como nós e agrupam um certo conjunto de nós. Os atributos também são considerados nós e referem-se a características de um determinado conjunto de nós. Por fim, as ligações estabelecem relações entre nós. O objetivo desta dissertação foi melhorar o processo de interação com a plataforma, criando um novo modelo de interação flexível. Foram expandidos os mecanismos de interação para sons não-vocais, gestos desenhados e a inclusão de um paradigma de tabela, para além de abordagem baseada em grafos. Foram acrescentadas novas funcionalidades como copiar/cortar nós, utilizar padrões para criar nós, customização do feedback da

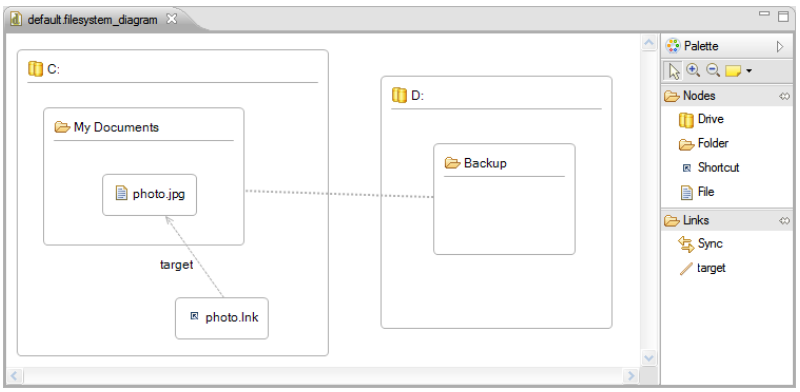


Figura II.4: Sistema de ficheiros proposto a ser modelado pelos utilizadores.

Figura II.5: Grafo ilustrativo de dois nós e uma ligação.

Tabela II.1: Exemplo de uma tabela de três nós e três ligações.

N1		
L1	N2	L3
L2		N3

ferramenta, customização do sintetizador de voz e disponibilização de apoio consoante os seus conhecimentos de modelação. O modelo de interação pode ser criado e customizado pelo Engenheiro de Linguagens, onde este pode definir o nível de conhecimentos e dificuldades visuais do futuro utilizador final, assim como os gestos e sons não vocais disponíveis de plataforma e as frases e som com que pretende que o *feedback* seja feito na plataforma. O modelo de interação pode também ser customizado pelo utilizador final à medida que este utiliza a plataforma. Este pode agora guardar as suas preferências nos campos referidos, e sempre que utilizar a ferramenta esta estará ajustada às suas necessidades e conhecimentos de modelação.

II.3.1 Paradigma de grafo

Na Figura II.5, os nós (N1 e N2) são ligados pela ligação (L1) num paradigma de grafo sem qualquer noção espacial.

II.3.2 Paradigma de tabela

No paradigma de tabela os nós são dispostos na diagonal principal da tabela (N1, N2, N3). As ligações são dispostas na coluna do nó atual mais acima e na linha do nó alvo. Como se pode observar na Tabela II.1 (L1 liga N1 a N2) (L2 liga N1 a N3) (L3 liga N3 a N2).

Na Tabela II.2 é apresentado o resultado produzido depois de modelar o sistema de ficheiros presente na fig. 4 no novo modelo de interação da plataforma Model-By-Voice utilizando o paradigma de tabela. Como enunciado, neste paradigma os nós sãos dispostos na diagonal principal da tabela e as ligações entre eles são colocadas na célula cujas coordenadas são o número da coluna do nó atual e o número da linha do nó alvo. Neste paradigma é possível navegar através das teclas do teclado e dirigir-se para qualquer célula

Tabela II.2: Tabela representativa do exemplo.

C:					
	My Documents				
		photo.jpg	target		
			photo.lnk		
				D:	
	sync				Backup

pretendida através de um sistema de coordenadas. Caso esteja numa célula e navegar para a célula acima, por exemplo, e se nessa célula estiver uma ligação, irá ser transportado para o nó de saída da ligação. Também é possível ordenar os nós da tabela na diagonal principal por ordem alfabética e listar todos os elementos do diagrama. Considerando apenas o resultado produzido na tabela, responda às seguintes perguntas.

II.3.3 Comandos disponíveis na plataforma:

- create diagram – Cria um diagrama na aplicação.
- create node – Cria um nó no diagrama criado.
- create attribute - Cria um atributo no diagrama criado.
- create compartment - Cria um compartimento no diagrama criado.
- create link - Cria uma ligação entre 2 nós no diagrama criado.
- remove navigation element – remove o atual element de navegação.
- remove attribute – remove o atributo pretendido.
- remove link – remove a ligação pretendida.
- list diagram – lista os elementos do diagrama.
- update navigation element – atualiza o elemento de navegação.
- detail navigation element – detalha o elemento de navegação.
- change navigation element – muda o elemento de navegação.
- copy – copia o nó pretendido e cola um novo nó com as mesmas características. (é possível guardar o o tipo do nó copiado como padrão)
- cut – corta o nó pretendido e cola um novo nó com as mesmas características. (é possível guardar o o tipo do nó copiado como padrão)

- use pattern – utiliza um padrão guardado para criar um novo nó com as características do padrão.
- speak neighbours – enuncia os vizinhos do elemento atual de navegação.
- save diagram – guarda o diagrama.
- open diagram – carrega um diagrama guardado.
- exit program – fecha o programa.
- undo – desfaz a última operação.
- activate help mode – ativa o modo de ajuda.
- record sound – grava um novo som.
- sound – permite utilizar sons para enunciar comandos.
- match gesture – configuração de um gesto para representar um comando.
- gesture – permite utilizar gestos para enunciar comandos.
- sleep – para a aplicação durante o tempo pretendido.
- deselect compartment – desselecciona o compartimento.

II.3.4 Comandos exclusivos do paradigma de tabela:

- go to – desloca-se para a célula pretendida na tabela. up - desloca-se para a célula acima na tabela.
- go down - desloca-se para a célula abaixo na tabela.
- go right - desloca-se para a célula à direita na tabela.
- go left - desloca-se para a célula à esquerda na tabela.
- go next node - desloca-se para o próximo nó na tabela.
- go previous node – desloca-se para o nó anterior na tabela.
- order alphabetically – ordena alfabeticamente os nós da tabela.